Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра вычислительных методов и программирования

И. Н. Коренская, И. В. Лущицкая

ОСНОВЫ ПРОГРАММИРОВАНИЯ В СРЕДЕ DELPHI. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Рекомендовано УМО по образованию в области приборостроения в качестве учебно-методического пособия для студентов специальностей 1-38 02 03, 1-27 01 01

Библиотека БГУИР

Минск БГУИР 2013

УДК 004.421(076)
ББК 32.973.26-0.18.2я73
К66

Рецензенты:

кафедра прикладной информатики
учреждения образования «Белорусский государственный аграрный технический университет» (протокол №8 от 22 марта 2012);

доцент кафедры информатики учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», кандидат технических наук Н. А. Волорова

Коренская, И. Н.

К66 Основы программирования в среде DELPHI. Лабораторный практикум : учеб.-метод. пособие / И. Н. Коренская, И. В. Лущицкая. – Минск : БГУИР, 2013. – 130 с.

ISBN 978-985-488-880-4.

Учебно-методическое пособие по дисциплине «Информатика» содержит краткие теоретические сведения по представлению чисел, системам счисления, основам алгоритмизации и программирования в среде Delphi. Предназначено для начального обучения студентов основам алгоритмизации и программирования в среде визуального программирования Delphi.

УДК 004.421(076)
ББК 32.973.26-0.18.2я73

ISBN 978-985-488-880-4

© Коренская И. Н.,
Лущицкая И. В., 2013
© УО «Белорусский государственный университет информатики и радиоэлектроники», 2013

Библиотека БГУИР

СОДЕРЖАНИЕ

Библиотека БГУИР

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №1

ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ В КОМПЬЮТЕРЕ. СИСТЕМЫ СЧИСЛЕНИЯ

Цель работы: научиться представлять числа в прямом, обратном, дополнительном кодах; осуществлять перевод чисел из десятичной системы счисления в двоичную, восьмеричную, шестнадцатеричную и обратно; из восьмеричной системы счисления в шестнадцатеричную и обратно; выполнять операции сложения и вычитания в двоичной, восьмеричной, шестнадцатеричной системах счисления.

1.1. Коды двоичных чисел

Любая информация (числа, команды, записи и т. п.) представляется в ЭВМ в виде двоичных кодов фиксированной или переменной длины. Отдельные элементы двоичного кода, имеющие значение 0 или 1, называют **разрядами** или **битами**. Двоичный код, состоящий из 8 разрядов, носит название **байта**. Для записи чисел также используют 32-разрядный формат (машинное слово), 16-разрядный формат (полуслово) и 64-разрядный формат (двойное слово).

Обычно старший разряд указывает на знак представляемых чисел, остальные разряды воспринимаются как цифровые. Значение **знакового разряда** для **положительных** чисел равно 0, а для **отрицательных** чисел – 1. Если количество разрядов кода не указано, предполагается, что под запись кода выделен **один байт**.

Применяют прямой, обратный и дополнительный коды чисел.

Прямой код используется при хранении чисел в памяти ЭВМ, а также при выполнении операций умножения и деления. Прямой код двоичного числа образуется из самого числа, к которому **дописываются нули** в зависимости от заданных разрядов кода. В знаковый разряд для положительных чисел записывается нуль, а для отрицательных – единица.

Например: +1101 → **0**0001101,
–1101 → **1**0001101.

Обратный и дополнительный коды применяют для замены операции вычитания операцией сложения, что упрощает устройство арифметического блока ЭВМ.

Обратный код для положительного числа совпадает с прямым кодом. Для отрицательного числа все цифры числа заменяются на противоположные (1 на 0, 0 на 1). В знаковый разряд заносится единица.

Например: +1101 → **0**0001101,
–1101 → **1**1110010.

Дополнительный код положительного числа совпадает с прямым кодом. Для отрицательного числа дополнительный код образуется путем получения обратного кода и добавлением к младшему разряду единицы.

Например: +1101 → **0**0001101,
–1101 → **1**1110011.

Библиотека БГУИР

1.2. Системы счисления

Системой счисления называется способ представления чисел с помощью заданного набора специальных знаков и соответствующие ему правила действия над ними.

Различают непозиционные и позиционные системы счисления.

В **непозиционных** системах счисления значение цифры не зависит от ее позиции в записи числа. *Примеры* непозиционных систем счисления: римская (XXIV), славянская.

В **позиционных** системах счисления значение цифры определяется ее позицией в записи числа. 2 1 0 – разряды

Например: $3\ 7\ 9 = 3 \cdot 10^2 + 7 \cdot 10^1 + 8 \cdot 10^0$.

Примеры позиционных систем счисления: десятичная (135), двоичная (1101_2), восьмеричная (777_8), шестнадцатеричная ($AE3_{16}$).

Алфавит системы счисления – это набор цифр в системе счисления.

Основание системы счисления – это количество цифр.

1.2.1. Двоичная система счисления

Двоичная система счисления лежит в основе работы компьютера, т. к. в нем существуют два устойчивых состояния: низкое или высокое напряжение, намагничено или не намагничено. Одному состоянию соответствует значение, равное 0, другому – 1.

Основание: 2.

Алфавит: 0, 1.

Примеры чисел: 0, 1, 10, 11, 100, 101, 110, 111, 1000… .

1.2.2. Восьмеричная система счисления

Основание: 8.

Алфавит: 0, 1, 2, 3, 4, 5, 6, 7.

Примеры чисел: 0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, 20… .

1.2.3. Шестнадцатеричная система счисления

Основание: 16.

Алфавит: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A (10), B (11) ,C (12), D (13), E (14), F (15).

Примеры чисел: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1A, 1B, 1C, 1D, 1E, 1F, 20,…, FF, 100… .

1.2.4. Перевод чисел из одной системы счисления в другую

1.2.4.1. Перевод чисел из десятичной системы счисления в другую

При переводе чисел из десятичной системы счисления в другую целую и дробную части переводят отдельно.

Для перевода **целой части** ее надо делить на основание системы счисления, в которую переводится число до тех пор, пока частное не станет меньше этого основания. В процессе деления следует фиксировать все остатки от деления. Число в новой системе счисления записывается, начиная с последнего частного, к которому дописываются остатки от деления в обратном порядке. В

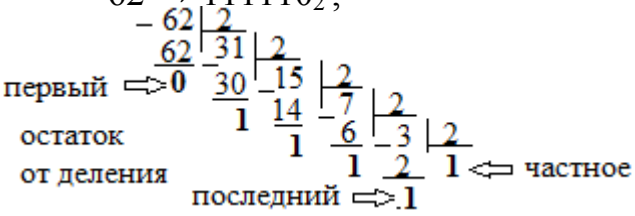
Библиотека БГУИР

результате получится: частное – старший разряд, а самый первый остаток – младший разряд.

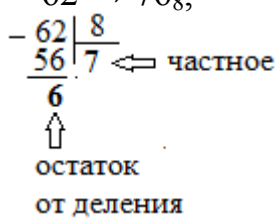
Например: переведем число 62,79 из десятичной в двоичную, восьмеричную, шестнадцатеричную системы счисления.

Сначала переведем целую часть числа, получим:

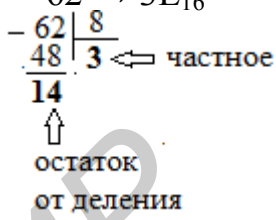
$62 \rightarrow 111110_2$,



$62 \rightarrow 76_8$,

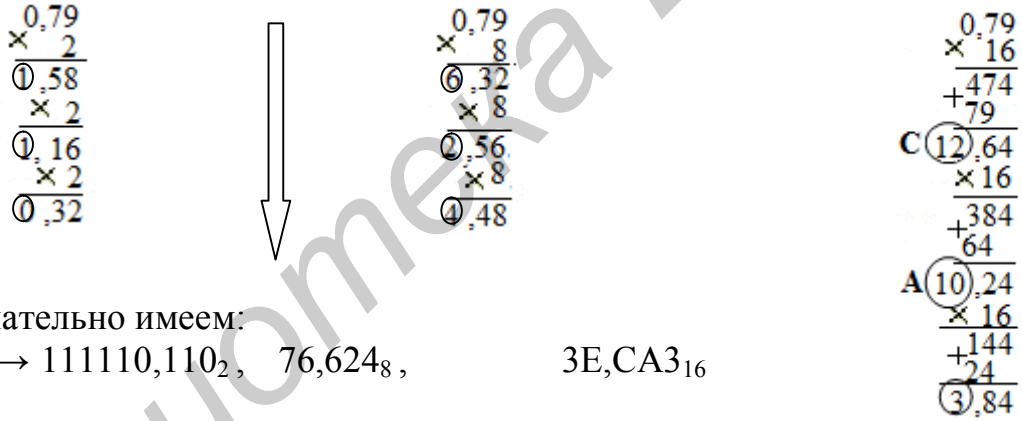


$62 \rightarrow 3E_{16}$



Для перевода **дробной части** ее надо умножать на основание системы счисления, в которую переводится число до тех пор, пока дробная часть не станет равной нулю или не будет достигнута заданная точность. При этом целая часть в процессе умножения не участвует, а используется при записи числа в новой системе счисления.

Переведем дробную часть числа с точностью 10^{-3} (3 знака в дробной части), получим: $0{,}79 \rightarrow 0{,}110_2$, $0{,}79 \rightarrow 0{,}624_8$, $0{,}79 \rightarrow 0{,}CA3_{16}$



Окончательно имеем:

$62{,}79 \rightarrow 111110{,}110_2$, $76{,}624_8$, $3E{,}CA3_{16}$

1.2.4.2. Перевод чисел из любой системы счисления в десятичную

При переводе чисел из любой системы счисления в десятичную надо число представить в виде суммы произведений цифры числа, умноженной на основание системы счисления, в которой записано число в степени разряда данной цифры.

Примеры:

$_{2\ 1\ 0}$ ← *разряды*

$1C5_{16} = 1 \cdot 16^2 + 12 \cdot 16^1 + 5 \cdot 16^0 = 256 + 192 + 5 = 453$;

$_{3\ 2\ 1\ 0\ -3\ -2}$ ← *разряды*

$1001{,}01_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 8 + 1 + 0{,}25 = 9{,}25$;

$_{1\ 0\ -1}$

$27{,}5_8 = 2 \cdot 8^1 + 7 \cdot 8^0 + 5 \cdot 8^{-1} = 16 + 7 + 0{,}125 = 23{,}125$.

Библиотека БГУИР

1.2.4.3. Перевод чисел из восьмеричной системы счисления в шестнадцатеричную и обратно

Правило перевода **из восьмеричной** системы счисления **в шестнадцатеричную**.

1. Каждую восьмеричную цифру заменяют тремя двоичными (**триадами**, т. к. $8 = 2^3$).

2. Полученное двоичное значение разбивают на **тетрады** (четыре двоичные цифры, т. к. $16 = 2^4$): целую часть числа – справа налево, дробную – слева направо.

3. Каждую тетраду заменяют шестнадцатеричной цифрой.

Например: $751{,}03_8 \rightarrow x_{16}$

7 5 1 0 3

1. Заменим каждую восьмеричную цифру триадой: $111\ 101\ 001{,}000\ 011_2$.
2. Разобьем полученное значение на тетрады: $0001\ 1110\ 1001{,}0000\ 1100_2$.

1 E 9 0 C

3. Заменим каждую тетраду шестнадцатеричным значением. Получим $1E9{,}0C_{16}$.

Рассмотрим правило перевода **из шестнадцатеричной в восьмеричную** систему счисления.

1. Каждую шестнадцатеричную цифру заменяют четырьмя двоичными цифрами (тетрадами, т. к. $16 = 2^4$).

2. Полученное двоичное значение разбивают на триады (три двоичные цифры, т. к. $8 = 2^3$): целую часть числа – справа налево, дробную – слева направо.

3. Каждую триаду заменяют восьмеричной цифрой.

Например: $FA{,}09_{16} \rightarrow x_8$

F A 0 9

1. Заменим каждую шестнадцатеричную цифру тетрадой: $1111\ 1010{,}0000\ 1001_2$.

2. Разобьем полученное значение на триады: $011\ 111\ 010{,}000\ 010\ 010_2$.

3 7 2 0 2 2

3. Заменим каждую триаду восьмеричным значением. Получим $372{,}022_8$.

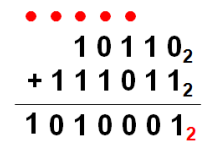
1.2.5. Операции сложения и вычитания в разных системах счисления

Рассмотрим операции сложения и вычитания **в двоичной** системе счисления.

Для выполнения операций сложения надо знать правила сложения в двоичной системе счисления: $0 + 0 = 0$ $0 + 1 = 1$ $1 + 0 = 1$

$1 + 1 = \mathbf{1}0_2$ $1 + 1 + 1 = \mathbf{1}1_2$

Например: $10110_2 + 111011_2 = 1010001_2$



Правила вычитания в двоичной системе счисления:

$0 - 0 = 0$ $1 - 1 = 0$ $1 - 0 = 1$ $\mathbf{1}0_2 - 1 = 1$

Например: $1000101_2 - 11011_2 = 101010_2$

В восьмеричной и шестнадцатеричной системах счисления сложение и вычитание происходит по тем же правилам, что и в десятичной, но надо помнить, что в восьмеричной после цифры 7 идет перенос в старший разряд, а в шестнадцатеричной – после F.

Библиотека БГУИР

Примеры: $765{,}71_8 + 475{,}6_8 = 1463{,}51_8$, $510{,}21_8 - 435{,}117_8 = 53{,}071_8$,

$$\begin{array}{r} 7\,6\,5,\,7\,1_8 \\ +\;4\,7\,5,\,6_8 \\ \hline 1\,4\,6\,3,\,5\,1_8 \end{array}$$

7+6=8+5
5+5+1=8+3
6+7+1=8+6
7+4+1=8+4
⇧ 1в перенос

$$\begin{array}{r} 5\,1\,0,\,2\,1_8 \\ -\;4\,3\,5,\,1\,1\,7_8 \\ \hline 5\,3,\,0\,7\,1_8 \end{array}$$

⇩ заем
8-7=1
(0+8)-1=7
(2-1)-1=0
(0+8)-5=3
(1-1+8)-3=5
(5-1)-4=0

$F1D{,}0A_{16} + E9{,}B7_{16} = 1006{,}C1_{16}$

$$\begin{array}{r} F\,1\,D,\,0\,A_{16} \\ +\;E\,9,\,B\,7_{16} \\ \hline 1\,0\,0\,6,\,C\,1_{16} \end{array}$$

A+7=10+7=17=16+1
0+B+1=0+11+1=12=C
D+9=13+9=22=16+6
1+E+1=1+14+1=16 ⇦ **1** в
F+1=15+1=16 ⇦ перенос

$1E0{,}03_{16} - 7F{,}5A_{16} = 160{,}A9_{16}$

$$\begin{array}{r} 1\,E\,0,\,0\,3_{16} \\ -\;7\,F,\,5\,A_{16} \\ \hline 1\,6\,0,\,A\,9_{16} \end{array}$$

⇩ заем
(3+16)−A=19−10=9
(0−1+16)−5=15−5=10 =A
(0−1+16)−F=15−15=0
(E−1)−7=(14−1)−7=6

1.3. Индивидуальные задания

1. Перевести числа из десятичной системы счисления в двоичную, восьмеричную и шестнадцатеричную (п. 2).

2. Перевести числа в десятичную систему счисления (п. 3).

3. Выполнить операции над числами (п. 4). Проверить правильность вычислений переводом исходных данных и результатов в десятичную систему счисления. Двоичные числа представить в прямом, обратном и дополнительном кодах.

4. Перевести число из восьмеричной системы счисления в шестнадцатеричную (п. 5).

5. Перевести число из шестнадцатеричной системы счисления в восьмеричную (п. 6).

6. Проверить полученные результаты при помощи программы Калькулятор (Стандартные).

Все переводы выполнять с точностью $\mathbf{10^{-3}}$. Данные взять из табл. 1.1.

Таблица 1.1

Индивидуальные задания по системам счисления

№ варианта	№ задания				
1	2	3	4	5	6
1	305_{10} $153{,}25_{10}$ $248{,}46_{10}$	$10110101{,}011_2$ $671{,}24_8$ $41A{,}6F_{16}$	$1101111011_2 \pm 1110010000_2$ $757{,}4_8 \pm 356{,}5_8$ $8EC{,}98_{16} \pm 59B{,}8_{16}$	$70{,}23_8$	$1E2{,}0C_{16}$
2	164_{10} $712{,}25_{10}$ $11{,}89_{10}$	$1111100111{,}01_2$ $413{,}41_8$ $11A{,}8C_{16}$	$11000110012 \pm 1101001100_2$ $7443{,}18 \pm 746{,}54_8$ $EA{,}416 \pm 2B4{,}C_{16}$	$1347{,}17_8$	$1B{,}D01_{16}$

Библиотека БГУИР

Продолжение табл. 1.1

1	2	3	4	5	6
3	273_{10} $156,25_{10}$ $797,5_{10}$	$1011110100,011_2$ $1017,2_8$ $1D1,B_{16}$	$1111000_2 \pm 1001001101_2$ $7577,2_8 \pm 573,74_8$ $7DB,9_{16} + \pm 3E8,8_{16}$	$112,04_8$	$41A,6_{16}$
4	358_{10} $377,5_{10}$ $87,27_{10}$	$11001100,011_2$ $112,04_8$ $3D4,A_{16}$	$100101100_2 \pm 101000011_2$ $7364,44_8 \pm 647,5_8$ $958,A_{16} \pm E4,C_{16}$	$1017,2_8$	$155,6C_{16}$
5	675_{10} $810,25_{10}$ $1017,25_{10}$	$110011000,1101_2$ $1347,17_8$ $15F,6C_{16}$	$111001111_2 \pm 1111010010_2$ $7345,74_8 \pm 465,24_8$ $E83,B_{16} \pm DF,8_{16}$	$1017,2_8$	$334,A_{16}$
6	808_{10} $176,25_{10}$ $284,25_{10}$	$110110011,01_2$ $1665,3_8$ $FA,7_{16}$	$101111110_2 \pm 110001101_2$ $6347,5_8 \pm 5742,3_8$ $9C6,8_{16} \pm BE,5_{16}$	$704,6_8$	$252,38_{16}$
7	306_{10} 467_{10} $218,5_{10}$	$1000001111,01_2$ $465,3_8$ $2AC,38_{16}$	$1000101000_2 \pm 1100001101_2$ $675,4_8 \pm 527,4_8$ $75E,8_{16} \pm 5DB,6_{16}$	$1665,3_8$	$1C7,68_{16}$
8	113_{10} $607,5_{10}$ $314,71_{10}$	$1110011100,111_2$ $704,6_8$ $36E,3B_{16}$	$110100101_2 \pm 1000000010_2$ $563,74_8 \pm 355,2_8$ $9E5,D8_{16} \pm 3BA,78_{16}$	$140,22_8$	$FA,7_{16}$
9	374_{10} $164,25_{10}$ $97,14_{10}$	$10000110,00101_2$ $777,16_8$ $1C7,E8_{16}$	$1001101010_2 \pm 1100000101_2$ $7157,5_8 \pm 742,6_8$ $97E,8_{16} \pm A4,8_{16}$	$465,3_8$	$1E9,4_{16}$
10	524_{10} $579,5_{10}$ $847,625_{10}$	$111001101,1001_2$ $140,22_8$ $1DE,54_{16}$	$11100100_2 \pm 1101010000_2$ $7457,6_8 \pm 476,3_8$ $D35,8_{16} \pm 3E9,A_{16}$	$1600,14_8$	$36A,38_{16}$
11	875_{10} $649,25_{10}$ $6,52_{10}$	$1101101000,01_2$ $1600,14_8$ $1E9,4_{16}$	$1001000101_2 \pm 1010111110_2$ $1453,3_8 \pm 620,2_8$ $D6A,7_{16} \pm 3F8,8_{16}$	$753,16_8$	$1DE,54_{16}$
12	294_{10} $976,625_{10}$ $282,73_{10}$	$1101100,01_2$ $1053,2_8$ $E00,6A_{16}$	$10111111_2 \pm 1000111110_2$ $7574,62_8 \pm 734,4_8$ $9BD,8_{16} \pm E7,C_{16}$	$1034,34_8$	$1E7,08_{16}$
13	617_{10} $545,25_{10}$ $84,82_{10}$	$111001000,01_2$ $1471,17_8$ $3EC,5_{16}$	$10101001112 \pm 1110100100_2$ $7365,748 \pm 754,16_8$ $B9D,816 \pm 7F8,4_{16}$	$1053,2_8$	$7E2,C6_{16}$
14	1047_{10} $518,625_{10}$ $198,91_{10}$	100001010_2 $452,63_8$ $1E7,08_{16}$	$1010111112 \pm 1101100101_2$ $7672,48 \pm 365,6_8$ $DE9,216 \pm 78B,A_{16}$	$605,02_8$	$B00,F6_{16}$
15	233_{10} $801,5_{10}$ $218,73_{10}$	$1000110001,101_2$ $1034,34_8$ $72B,6A_{16}$	$1011110012 \pm 1010110101_2$ $756,748 \pm 322,5_8$ $A5E,9816 \pm D3,2_{16}$	$1471,17_8$	$9C,0D_{16}$
16	969_{10} $508,5_{10}$ $281,09_{10}$	$110010010,101_2$ $605,02_8$ $3C8,8_{16}$	$10011111102 \pm 1100010100_2$ $7563,78 \pm 765,34_8$ $F7A,E816 \pm 9D,7_{16}$	$452,63_8$	$3EC,5_{16}$

Библиотека БГУИР

Окончание табл. 1.1

1	2	3	4	5	6
17	163_{10} $352{,}375_{10}$ $288{,}61_{10}$	$1000001101{,}01_2$ $247{,}1_8$ $8E1{,}4C_{16}$	$1101100101_2 \pm 1111101000_2$ $3174{,}6_8 \pm 675{,}34_8$ $38A{,}E_{16} + D9{,}98_{16}$	$1636{,}24_8$	$C7{,}78_{16}$
18	917_{10} $74{,}5_{10}$ $84{,}33_{10}$	$111110100{,}101_2$ $1446{,}62_8$ $9C{,}D1_6$	$1110111111_2 \pm 10001100101_2$ $7664{,}7_8 \pm 576{,}3_8$ $DF9{,}6_{16} \pm 8B{,}4_{16}$	$61{,}72_8$	$3C8{,}8_{16}$
19	182_{10} $863{,}25_{10}$ $75{,}2_{10}$	$11100011{,}11_2$ $1762{,}7_8$ $1B5{,}61_6$	$111011011_2 \pm 1110001001_2$ $742{,}4_8 \pm 456{,}7_8$ $A9E{,}9_{16} \pm D8{,}4_{16}$	$16{,}36_8$	$3D{,}5_{16}$
20	804_{10} $435{,}75_{10}$ $30{,}43_{10}$	$110101100{,}1011_2$ $1164{,}36_8$ $1D5{,}C8_{16}$	$11110100_2 \pm 110000100_2$ $1774{,}6_8 \pm 735{,}4_8$ $E78{,}C_{16} \pm F7{,}4_{16}$	$247{,}1_8$	$3C1{,}6_{16}$
21	753_{10} $90{,}065_{10}$ $62{,}88_{10}$	$1001011101{,}011_2$ $615{,}72_8$ $3DA{,}5_{16}$	$1101111010_2 \pm 1111010011_2$ $647{,}6_8 \pm 477{,}5_8$ $8F9{,}B_{16} \pm 3D2{,}5_{16}$	$16{,}62_8$	$8C{,}4A_{16}$
22	571_{10} $580{,}375_{10}$ $106{,}67_{10}$	$11010110{,}00001_2$ $1343{,}66_8$ $3C3{,}6_{16}$	$11111101_2 \pm 101110110_2$ $1276{,}34_8 \pm 712{,}57_8$ $7FE{,}58_6 \pm 939{,}7_{16}$	$171{,}3_8$	$1B5{,}6_{16}$
23	244_{10} $1027{,}375_{10}$ $151{,}44_{10}$	$1001011{,}0101_2$ $171{,}3_8$ $3A3{,}4F_{16}$	$110010111_2 \pm 1011101111_2$ $7443{,}7_8 \pm 556{,}24_8$ $1BE{,}9_{16} \pm D9A{,}38_{16}$	$1031{,}5_8$	$18F{,}8_{16}$
24	388_{10} $674{,}25_{10}$ $159{,}05_{10}$	$100101011{,}101_2$ $750{,}51_8$ $9C0{,}F8_{16}$	$1011101112 \pm 1100001011_2$ $5367{,}248 \pm 3775{,}64_8$ $B25{,}816 \pm 2FD{,}4_{16}$	$134{,}66_8$	1DD,216
25	386_{10} $270{,}25_{10}$ $317{,}32_{10}$	$1001011001{,}011_2$ $1335{,}2_8$ $18F{,}A8_{16}$	$1011112 \pm 1100010_2$ $1643{,}28 \pm 162{,}44_8$ $DA9{,}416 \pm E4{,}B_{16}$	$1234{,}2_8$	$1D8{,}E4_{16}$
26	279_{10} $572{,}25_{10}$ $184{,}97_{10}$	$1110100{,}0011_2$ $1234{,}2_8$ $1DF{,}2_{16}$	$100101112 \pm 10101010_2$ $765{,}348 \pm 343{,}7_8$ $9FE{,}416 \pm B3{,}78_{16}$	$750{,}51_8$	$3A3{,}4_{16}$
27	1003_{10} $204{,}25_{10}$ $241{,}39_{10}$	$110011001{,}01_2$ $1031{,}5_8$ $1E8{,}2A_{16}$	$1001110112 \pm 101000001_2$ $1636{,}548 \pm 756{,}44_8$ $B8E{,}916 \pm A2{,}6_{16}$	$1335{,}2_8$	$9B0{,}8E_{16}$
28	414_{10} $330{,}5_{10}$ $115{,}41_{10}$	$111100011{,}1_2$ $150{,}44_8$ $3F7{,}7_{16}$	$11011011_2 \pm 100000001_2$ $246{,}7_8 \pm 75{,}2_8$ $A9D{,}8_{16} \pm 1B9{,}7_{16}$	$413{,}2_8$	$1F8{,}D6_{16}$
29	775_{10} $158{,}3125_{10}$ $1{,}09_{10}$	$100100011{,}0011_2$ $236{,}63_8$ $14A{,}6C_{16}$	$101110111_2 \pm 110010110_2$ $1635{,}4_8 \pm 576{,}2_8$ $34C{,}D_{16} \pm CB{,}4_{16}$	$2015{,}5_8$	$B0{,}8D_{16}$
30	149_{10} $184{,}75_{10}$ $61{,}52_{10}$	$111111001{,}1011_2$ $1636{,}24_8$ $C7{,}7F_{16}$	$10110111_2 \pm 100001000_2$ $1542{,}3_8 \pm 670{,}1_8$ $2F3{,}4_{16} \pm 1EA{,}8_{16}$	$150{,}44_8$	$2A5{,}C2_{16}$

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №2

ПРОГРАММИРОВАНИЕ ЛИНЕЙНЫХ АЛГОРИТМОВ

Цель работы: научиться составлять простейшие программы в среде Delphi. Написать и отладить программу линейного алгоритма.

2.1. Интегрированная среда разработчика Delphi

Среда Delphi визуально реализуется в виде нескольких одновременно раскрытых на экране монитора окон. Их количество, расположение, размер и вид можно изменять, что значительно повышает производительность работы. При запуске Delphi на экране появляются пять окон (рис. 2.1):

- главное – Delphi;
- стартовая форма – Form1;
- редактор свойств объектов – Object Inspector;
- просмотр списка объектов – Object TreeView;
- редактор кода – Unitl.pas.

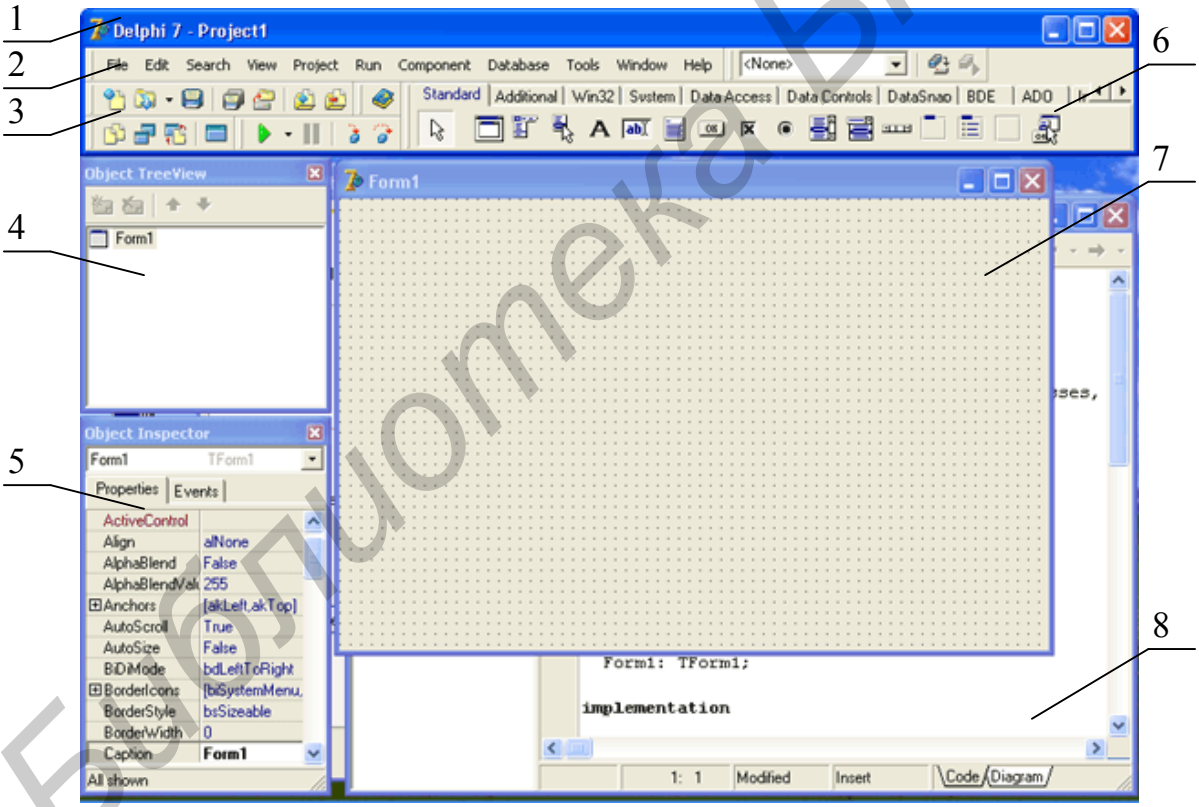


Рис. 2.1. Вид экрана после запуска DELPHI:

1 – главное окно;
2 – основное меню;
3 – значки основного меню;
4 – окно просмотра дерева объектов;
5 – окно инспектора объектов;
6 – палитра компонентов;
7 – окно пустой формы;
8 – окно кода программы

Библиотека БГУИР

Главное окно всегда присутствует на экране и предназначено для управления процессом создания программы. Основное меню содержит все необходимые средства для управления проектом. Пиктограммы облегчают доступ к наиболее часто применяемым командам основного меню. Через меню компонентов осуществляется доступ к набору стандартных сервисных программ среды DELPHI, которые описывают некоторый визуальный элемент (компонент), помещенный в окно формы. Каждый компонент имеет определенный набор свойств (параметров), которые можно задавать. *Например*: цвет, заголовок окна, надпись на кнопке, размер и тип шрифта и др.

Окно инспектора объектов (вызывается клавишей F11) предназначено для изменения свойств выбранных компонентов и состоит из двух страниц. Страница Properties (**Свойства**) предназначена для изменения свойств компонента, страница Events (**События**) – для определения реакции компонента на то или иное событие (*например* нажатие клавиши или щелчок мышью по кнопке).

Окно формы представляет собой проект Windows-окна программы. В него в процессе написания программы помещаются необходимые компоненты. Причем при выполнении программы помещенные компоненты будут иметь тот же вид, что и на этапе проектирования.

Окно программного кода предназначено для просмотра, написания и редактирования кода программы. В Delphi используется язык программирования Object Pascal. При первоначальной загрузке в окне кода программы находится код, содержащий минимальный набор операторов для нормального функционирования пустой формы в качестве Windows-окна. При помещении компонента в окно формы код программы автоматически дополняется описанием необходимых для его работы библиотек стандартных программ (раздел `Uses`) и типов переменных (раздел `Type`).

Программа в среде Delphi составляется как описание алгоритмов, которые надо выполнить при возникновении определенного события, связанного с формой (*например* щелчок мышью по кнопке – событие OnClick, создание формы – OnCreate). Для каждого обрабатываемого на форме события с помощью страницы Events инспектора объектов в коде программы организуется процедура (`Procedure`), между ключевыми словами `Begin` и `End` которой записывается на языке Object Pascal требуемый программный код.

Переключение между окнами формы и кода программы осуществляется нажатием клавиши F12.

2.2. Структура программ DELPHI

Для каждого приложения среда создает файлы со следующими расширениями:

– *.dpr – файл описания проекта, в котором описываются все формы проекта (*например* Project1.dpr) и запускается само приложение (Application.run); для просмотра его кода надо выполнить: Project – View Source;

– *.pas – файл модуля Unit, который является кодом программы для формы (*например* Unit1.pas для Form1);

Библиотека БГУИР

– *.dfm – файл описания формы и ее компонентов (*например* Unit1.dfm), который может храниться в виде бинарного или текстового файла;

– *.res – файл ресурсов, в котором хранятся значки, картинки, меню, константы, помещаемые на форму (*например* Project1.res);

– *.dof – файл настроек проекта (*например* Project1.dof);

– *.dcu – результат трансляции модуля с расширением *.pas, т. е. программный код модуля, представленный в машинных кодах;

– *.exe – результат редактирования программы, т. е. объединение всех модулей *.dcu в одну готовую к выполнению программу.

При выполнении лабораторных работ следует сохранять файлы с расширениями *.dpr, *.pas, *.dfm и *.res. Остальные являются рабочими, и их можно не сохранять.

Модуль Unit представляет собой отдельную программную единицу. В результате трансляции создается машинный код, записываемый в файл с расширением *.dcu.

Структура модуля `Unit` может иметь следующий вид:

```
Unit Имя модуля;
Interface              // Интерфейсная часть модуля – секция описания
Uses …;                // Имена подключаемых модулей
// Объявления глобальных типов, констант, переменных, заголовков процедур и
// функций, доступных в других модулях, подключивших данный модуль
Implementation                  // Секция реализации модуля
Uses …;                         // Имена подключаемых модулей
// Объявления локальных (внутренних) констант, типов, переменных, процедур и
// функций, доступных только внутри данного модуля; внутри секции записы-
// ваются коды процедур и функций, объявленных в интерфейсной части модуля
Initialization  // Секция инициализации модуля (может отсутствовать)
// Помещаются операторы, выполняемые сразу после загрузки программы в
// память ЭВМ; секция выполняется в том порядке, в котором модули Unit
// описаны в основной программе в разделе Uses
Finalization    // Секция завершения (может отсутствовать)
// Выполнение операторов происходит после окончания работы программы перед
// выгрузкой ее из оперативной памяти ЭВМ; выполняется в обратном порядке
// по сравнению с порядком выполнения секции инициализации
End.                    // Конец модуля Unit
```

2.3. Порядок выполнения задания

Составить программу вычисления арифметического выражения для заданных значений x, y, z: $u = \mathrm{tg}^2(x+y) - e^{y-z}\sqrt{\cos x^2 + \sin^2|z-x|}$.

Панель диалога программы организовать в виде, представленном на рис. 2.2.

Библиотека БГУИР

2.3.1. Настройка формы

Пустая форма в правом верхнем углу имеет кнопки управления, предназначенные для:

– свертывания формы в пиктограмму  ;
– развертывания формы на весь экран  ;
– возвращения к исходному размеру  ;
– закрытия формы  .

С помощью мыши, «захватывая» одну из кромок формы или выделенную строку заголовка, можно отрегулировать нужные размеры формы и ее положение на экране.

2.3.2. Изменение заголовка формы

Новая форма (*например* Form1) имеет одинаковые имя (Name) и заголовок (Caption). Имя формы менять не рекомендуется, т. к. оно входит в код программы.

Для изменения заголовка вызовите окно инспектора объектов (F11) и щелкните левой кнопкой мыши (ЛКМ) в свободном от компонентов месте формы. В инспекторе объектов Object Inspector щелкните ЛКМ на свойстве Caption (вкладка Properties). В выделенной строке наберите свои данные, *например* ‘Лаб. раб. N 1. Ст. гр. 112501 Иванов А.А.’.

2.3.3. Размещение строки ввода (TEdit)

Для ввода из формы в программу или вывода на форму информации, которая вмещается в одну строку, используют **строку ввода**, представляемую компонентом TEdit. В программе (п. 2.3.10) с помощью такого компонента вводятся значения переменных x, y, z.

Для размещения компонента TEdit на форме щелкните ЛКМ на пиктограмме  (вкладка Standard в меню компонентов) и далее в том месте формы, где он появится. Аналогично добавьте на форму еще два компонента TEdit. Захватывая их мышью, отрегулируйте размеры и положение. Обратите внимание на то, что в коде программы в разделе описания переменных Var появились 3 новые однотипные переменные Edit1, Edit2, Edit3. Для каждой из них в свойстве Text будет содержаться строка символов (тип String) и отображаться в соответствующем окне Edit.

Так как численные значения переменных x, y, z имеют действительный тип `Extended`, то для **преобразования строковой записи числа**, находящегося в `Edit1.Text`, в **действительное значение** используется стандартная функция `StrToFloat()`, *например* `x:=StrToFloat(Edit1.Text);`

Если данные имеют целочисленный тип, *например* `Integer`, то применяется стандартная функция преобразования **из строки в целое число**, *например* `x:=StrToInt(Edit1.Text);`

В записи числа не должно быть пробелов. Действительное число пишется с десятичной точкой или запятой, что определяется настройкой ОС Windows.

С помощью инспектора объектов установите шрифт и размер символов, отображаемых в строке TEdit (Object Inspector – Properties – Font).

Библиотека БГУИР

2.3.4. Размещение надписей (TLabel)

На проектируемой форме (рис. 2.2) создаются 4 пояснительные надписи (компоненты TLabel). Для их размещения щелкните ЛКМ сначала на пиктограмме A меню компонентов Standard и далее в том месте формы, где появятся надписи. Отрегулируйте их размер, предварительно выделив надпись на форме. В свойство Caption инспектора объектов введите строку поясняющего текста, *например* 'Введите значение X:', и установите шрифт и размер символов (свойство Font).

Обратите внимание на то, что в коде программы автоматически появились 4 новых переменных типа TLabel. В их свойствах Caption хранятся пояснительные строки.

2.3.5. Размещение многострочного окна вывода (TMemo)

Для вывода результатов работы программы обычно используется многострочное окно вывода (компонент с типом TMemo). Выберите в меню компонентов пиктограмму и поместите его на форму. С помощью мыши отрегулируйте его местоположение и размер. Установите, предварительно выделив компонент, вертикальную и горизонтальную полосы прокрутки (свойства ScrollBars – ssBoth).

В результате в коде программы появится переменная Memo1 типа TMemo. Информация, отображаемая построчно в окне типа TMemo, находится в массиве строк Memo1.Lines (свойство Lines). Каждая строка имеет тип String.

Для очистки окна используется метод `Memo1.Clear;` Для добавления новой строки в окно применяется метод `Memo1.Lines.Add`(переменная типа String)`;`

Для вывода значения переменной действительного или целого типа его надо преобразовать к типу `String` и добавить в массив Memo1.Lines. *Например* если переменная `u:=100;` – целого типа, то при использовании метода `Memo1.Lines.Add('Значение u = '+IntToStr(u));` в окне Memo1 появится строка: Значение u = 100. Если переменная `u:=-256.38666;` – действительного типа, то в результате применения метода `Memo1.Lines.Add('Значение u ='+FloatToStrF(u,ffFixed,8,2));` будет выведена строка: 'Значение u = –256,39'. В соответствии с заданным форматом `ffFixed,8,2` под все число отводится восемь позиций, две из которых занимает его дробная часть.

Если число строк в массиве типа TMemo превышает размер окна, то для их просмотра применяется вертикальная полоса прокрутки (свойства ScrollBars – ssVertical). Если длина выводимой строки превосходит количество символов в строке окна, то для просмотра всей строки используется горизонтальная полоса прокрутки (свойства ScrollBars – ssHorizontal).

Библиотека БГУИР

2.3.6. Размещение стандартной кнопки закрытия проекта (TBitBtn)

Для создания кнопки, при нажатии которой программа завершит работу, щелкните на пиктограмме  , расположенной в меню компонентов вкладки Additional и поместите компонент на форму. Отрегулируйте положение и размер кнопки. Установите шрифт, начертание, размер и цвет символов надписи кнопки (свойство Font). Разместите на ней стандартную надпись Close. Для этого в инспекторе объектов установите свойства Kind – bkClose. Кнопка bkClose закрывает главное окно и завершает работу программы.

Отличительная особенность компонента TBitBtn – наличие растрового изображения на поверхности кнопки, определяемое свойством Glyph. При помощи свойства Kind можно задать одну из 11 стандартных разновидностей кнопок. Нажатие любой из них, кроме bkCustom и bkHelp, закрывает модальное окно и возвращает в программу результат mr*** (*например* bkOk – mrOk).

2.3.7. Написание процедуры обработки события создания формы (FormCreate)

Создадим программу – обработчик события «создание формы» OnCreate, которое будет выполняться при запуске программы. В результате действия процедуры FormCreate начальные значения переменных x, y, z помещаются в соответствующие строки TEdit и в окно TMemo выводится строка с указанием номера группы и фамилии студента.

Для этого дважды щелкните мышью (2ЛКМ) в любом свободном от компонентов месте формы. На экране появится программный код, в котором автоматически прописывается заголовок процедуры – обработчика события создания формы `Procedure TForm1.FormCreate(Sender:TObject);` Между операторами `Begin` и `End` записываются операторы, необходимые для начальной инициализации формы (п. 2.3.10).

2.3.8. Написание процедуры обработки события нажатия кнопки (ButtonClick)

Кнопка, *например* Button1, описывается компонентом TButton. Для ее создания выберите в меню компонентов вкладки Standart пиктограм  и поместите его на форму. С помощью инспектора объектов измените заголовок Caption компонента Button1 на Выполнить. Отрегулируйте положение и размер кнопки. Установите шрифт, начертание и размер символов надписи кнопки (свойство Font).

Щелкните 2ЛКМ внутри кнопки. В результате появится код программы с заголовком процедуры обработчика события – нажатия кнопки

```
Procedure TForm1.ButtonClick(Sender:TObject);
```

В появившемся окне редактора кода программы наберите код процедуры, приведенный в примере ниже (п. 2.3.10).

Библиотека БГУИР

2.3.9. Запуск и работа с программой

Запустить программу можно тремя способами, нажав:

– пиктограмму ▶ ;
– клавишу F9;
– Run в главном меню Run.

Сначала происходит трансляция и, если нет ошибок, компоновка проекта и создание единого загружаемого файла с расширением .exe. На экране появится активная форма проекта (рис. 2.2).

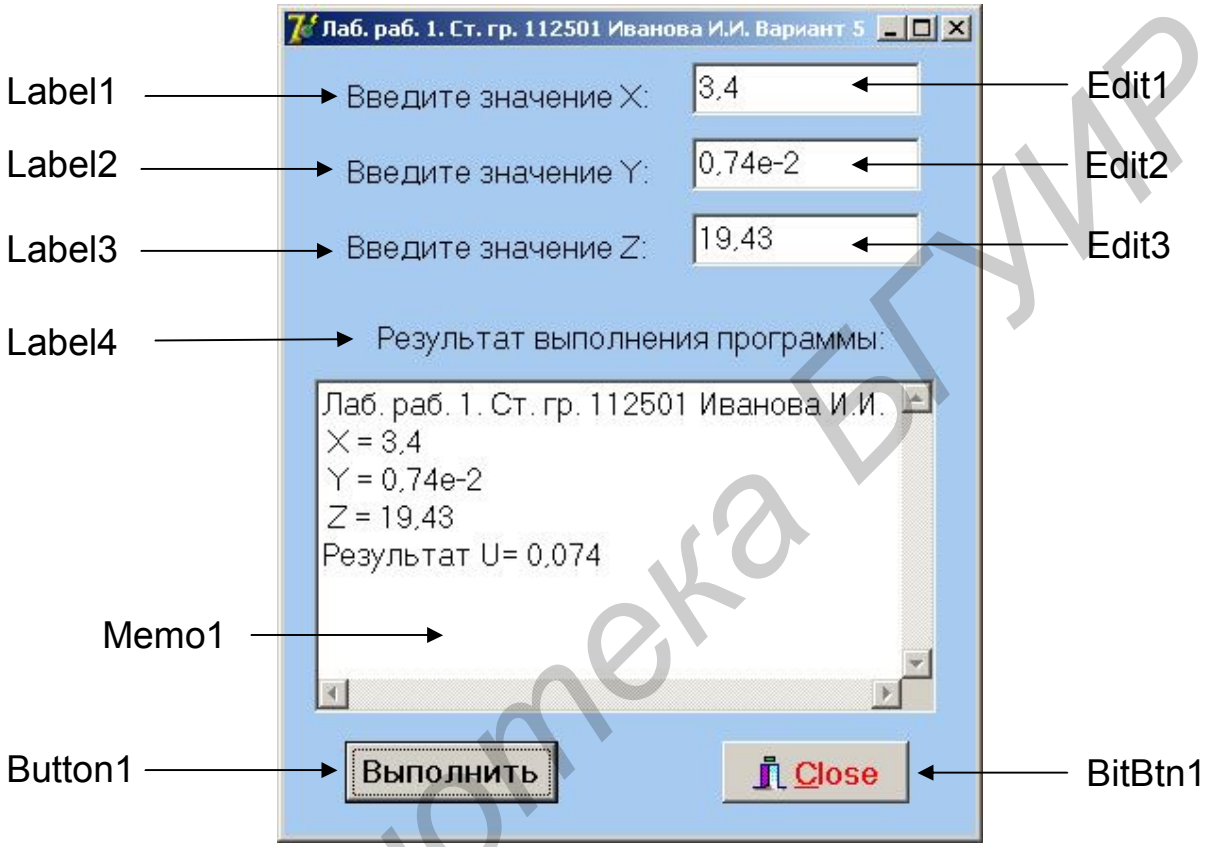


Рис. 2.2. Активная форма проекта

Для получения результатов вычисления с заданными на форме значениями после запуска программы нажмите кнопку Выполнить. В окне Memo1 появится результат. При изменении исходных значений x, y, z в строках TEdit и нажатии кнопки Выполнить выведутся новые результаты. Завершить работу программы можно, выбрав меню Run – Program Reset, кнопку ✕ или Close.

Ниже представлена блок-схема алгоритма (рис. 2.3).

Библиотека БГУИР

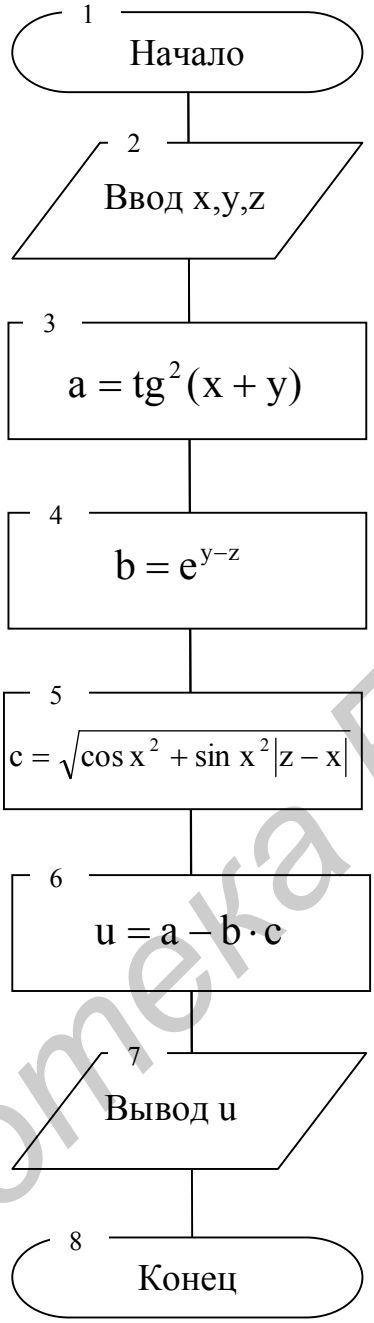


Рис. 2.3. Блок-схема алгоритма

2.3.10. Код программы

```
unit Unit1;
interface
uses                    //Подключение модуля математических функций Math
 Windows, Messages, SysUtils, Variants, Classes, Graph-
ics, Controls, Forms, Dialogs, Buttons, StdCtrls, Math;
Type
 TForm1 = class(TForm)
  Label1: TLabel;       //Описание помещенных на форму компонентов
  Label2: TLabel;
```

Библиотека БГУИР

```
  Label3: TLabel;
  Edit1: TEdit;
  Edit2: TEdit;
  Edit3: TEdit;
  Label4: TLabel;
  Memo1: TMemo;
  Button1: TButton;
  BitBtn1: TBitBtn;
  procedure FormCreate(Sender: TObject);  //Описание заголовков
  procedure Button1Click(Sender: TObject);//созданных процедур
private
  { Private declarations }
public
  { Public declarations }
end;
var
 Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.FormCreate(Sender: TObject);   // Процедура
begin              // создания формы, выполняемая при запуске программы
 Edit1.Text:='3,4';                  // Начальное значение X
 Edit2.Text:='0,74e–2';              // Начальное значение Y
 Edit3.Text:='19,43';                // Начальное значение Z
 Memo1.Clear;                        // Очистка окна редактора Memo1
// Вывод строки в многострочный редактор Memo1
 Memo1.Lines.Add('Лаб. раб. 1. Ст. гр. 112501 Иванова И.И. ');
end;
procedure TForm1.Button1Click(Sender: TObject);
Var                                  // Описание всех переменных,
 x,y,z,a,b,c,u:Extended;             // используемых в программе
 begin
 x:=StrToFloat(Edit1.Text);          // Считывание значения X
 Memo1.Lines.Add('X = '+Edit1.Text); // Вывод X в окно Memo1
 y:=StrToFloat(Edit2.Text);          // Считывание значения Y
 Memo1.Lines.Add('Y = '+Edit2.Text); // Вывод Y в окно Memo1
 z:=StrToFloat(Edit3.Text);          // Считывание значения Z
 Memo1.Lines.Add('Z = '+Edit3.Text); // Вывод Z в окно Memo1
 a:=Sqr(Tan(x+y));          // Вычисление арифметического выражения
 b:=Exp(y-z);
 c:=Sqrt(Cos(Sqr(x))+Sqr(Sin(Abs(z-x))));//c = √(cos x² + sin²|z − x|)
 u:=a-b*c;
```

Библиотека БГУИР

```
 Memo1.Lines.Add('Результат U='+FloatToStrF(u,ffFixed,8,3));
end; // Вывод результата в окно Memo1  с точностью 3 цифры в дробной части
end.
```

2.4. Индивидуальные задания

Составить программу вычисления выражения согласно указанному преподавателем варианту. Уточнить условие задания, количество, наименование, типы исходных данных. В соответствии с этим установить необходимое количество компонентов TEdit, тексты надписей на форме, размеры шрифтов, а также типы переменных и функции преобразования при вводе и выводе результатов. С помощью инспектора объектов изменить цвет формы, шрифт выводимых символов.

1. $t=\dfrac{2\cos(x-\frac{2}{3})}{0{,}5+\sin^2 y}\left(1+\dfrac{z^2}{3-\frac{z^2}{5}}\right)$

При **x** = 14,26, **y** = –1,22, **z** = $3{,}5\cdot10^{-2}$. **Ответ**: **t** = 0,7492

2. $u=\dfrac{\sqrt[3]{9+|x-y^2|}}{x^2+y^2+2}-e^{|x-y|}tg^3 z$

При **x** = –4,5, **y** = $0{,}75\cdot10^{-4}$, **z** = $0{,}845\cdot10^{2}$. **Ответ**: **u** = 3,51460

3. $v=\dfrac{1+\sin^2(x+y)}{\left|x-\dfrac{2y}{1+x^2y^2}\right|}x^{|y|}+\cos^2(\operatorname{arctg}\dfrac{1}{z})$

При **x** = $3{,}74\cdot10^{-2}$, **y** = –0,825, **z** = $0{,}16\cdot10^{2}$. **Ответ**: **v** = 1,055

4. $w=|\cos x-\cos y|^{1+2\sin^2 y}(1+z+\frac{z^2}{2}+\frac{z^3}{3}+\frac{z^4}{4})$

При **x** = $0{,}4\cdot10^{4}$, **y** = –0,875, **z** = $-0{,}475\cdot10^{-3}$. **Ответ**: **w** = 1,9873

5. $\alpha=\ln(y^{-\sqrt{|x|}})(x-\dfrac{y}{2})+\sin^2(\operatorname{arctg} z)$

При **x** = –15,246, **y** = $4{,}642\cdot10^{-2}$, **z** = 21. **Ответ**: **α** = –182,038

6. $\beta=\sqrt{10(\sqrt[3]{x}+x^{y+2})}(\arcsin^2 z-|x-y|)$

При **x** = $16{,}55\cdot10^{-3}$, **y** = –2,75, **z** = 0,15. **Ответ**: **β** = –40,6307

Библиотека БГУИР

7. $\gamma = 5\text{arctgx} - \frac{1}{4}\text{arccosx}\,\frac{x + 3|x - y| + x^2}{|x - y|z + x^2}$

При **x** = 0,1722, **y** = 6,33, **z** = $3{,}25 \cdot 10^{-4}$. **Ответ**: **γ** = –205,306

8. $j = \frac{e^{|x-y|}\,|x-y|^{|x+y|}}{arctgx - arctgz} + \sqrt[3]{x^6 + \ln^2 y}$

При **x** = $-2{,}235 \cdot 10^{-2}$, **y** = 2,23, **z** = 15,221. **Ответ**: **j** = 39,3741

9. $\psi = \left| x^{\frac{y}{x}} - \sqrt[3]{\frac{y}{x}} \right| + (y - x)\frac{\cos y - \frac{z}{y - x}}{1 + (y - x)^2}$

При **x** = $1{,}825 \cdot 10^{2}$, **y** = 18,225, **z** = $-3{,}298 \cdot 10^{-2}$. **Ответ**: **ψ** = 1,2131

10. $a = 2^{-x}\sqrt{x + \sqrt[4]{|y|}}\,\sqrt[3]{e^{x - \frac{1}{\sin z}}}$

При **x** = $3{,}981 \cdot 10^{-2}$, **y** = $-1{,}625 \cdot 10^{3}$, **z** = 0,512. **Ответ**: **a** = 1,26185

11. $b = y^{\sqrt[3]{|x|}} + \frac{\cos^3 y}{e^{|x-y|} + \frac{x}{2}}|x - y|\left(1 + \frac{\sin^2 z}{\sqrt{x + y}}\right)$

При **x** = 6,251, **y** = 0,827, **z** = 25,001. **Ответ**: **b** = 0,7121

12. $c = 2^{(y^x)} + (3^x)^y \frac{y\left(\text{arctgz} - \frac{1}{3}\right)}{|x| + \frac{1}{y^2 + 1}}$

При **x** = 3,251, **y** = 0,325, **z** = $0{,}466 \cdot 10^{-4}$. **Ответ**: **c** = 4,23655

13. $f = \frac{\sqrt[4]{y + \sqrt[3]{x - 1}}}{|x - y|(\sin^2 z + tgz)}$

При **x** = 17,421, **y** = $10{,}365 \cdot 10^{-3}$, **z** = $0{,}828 \cdot 10^{5}$. **Ответ**: **f** = 0,33056

Библиотека БГУИР

14. $g = \dfrac{y^{x+1}}{\sqrt[3]{|y-2|+3}} + \dfrac{x+\dfrac{y}{2}}{2|x+y|}(x+1)^{-\frac{1}{\sin z}}$

При **x** = $12{,}3\cdot10^{-1}$, **y** = 15,4, **z** = $0{,}252\cdot10^{3}$. **Ответ**: **g** = 82,8256

15. $h = \dfrac{x^{y+1}+e^{y-1}}{1+x|y-tgz|}(1+|y-x|)+\dfrac{|y-x|^2}{2}-\dfrac{|y-x|^3}{3}$

При **x** = 2,444, **y** = $0{,}869\cdot10^{-2}$, **z** = $-0{,}13\cdot10^{3}$. **Ответ**: **h** = –0,4987

16. $s = \dfrac{\left|x-\dfrac{3y}{1+xy^2}\right|}{1+\cos^2(x+y)}x^{|y|}+\sin^2\left(arctg\dfrac{1}{z}\right)$

При **x** = $3{,}999\cdot10^{-2}$, **y** = –6,011, **z** = $0{,}245\cdot10^{3}$. **Ответ**: **S** = 2,61678

17. $u = \dfrac{\left|x^2+y^2+2\right|}{\sqrt[3]{x+|x-y|^2+1}} - e^{|x-y|}+\left(tg^2z+1\right)$

При **x** = 1,78, **y** = 3,83, **z** = $13{,}57\cdot10^{2}$. **Ответ**: **u** = –1,7389

18. $r = e^{|y-x|}arctgx - \dfrac{1}{4}arctgx\cdot\ln(y-\sqrt{x})\left(x-\dfrac{y}{2}\right)$

При **x** = 1,471, **y** = 4,62. **Ответ**: **r** = 22,9512

19. $t = \sin^2(arctg(z)\cdot x^3)+\ln\dfrac{\left|\sqrt{x-y}\right|}{y-\dfrac{x}{2}}$

При **x** = $2{,}126\cdot10^{-2}$, **y** = 4,62, **z** = $18{,}52\cdot10^{5}$. **Ответ**: **t** = –0,0298

20. $c = (3x)^y - \dfrac{y\left(arctgz-\dfrac{\pi}{6}\right)}{\sin|x|+\dfrac{1}{y^2+1}}$

При **x** = 7,5, **y** = $23{,}8\cdot10^{-4}$, **z** = 1,4. **Ответ**: **c** = 29,3869

Библиотека БГУИР

21. $u = \ln(y-\sqrt{|x|})\left(x+\frac{y}{2}\right)^3 + \frac{1-\sin z}{1+\cos z}$

При **x** = $3{,}466\cdot10^{-3}$, **y** = 2,743, **z** = 1,43. **Ответ**: **u** = 2,5752

22. $w = \ln\frac{1+\sqrt{\sin z}}{1-\sqrt{\cos z}} + 2arctg\sqrt[3]{x+y}\,\frac{1+2y}{x^4}$

При **x** = $1{,}499\cdot10^{2}$, **y** = 0,1174, **z** = 0,43. **Ответ**: **w** = 3,5642

23. $v = 2^{\arcsin 3z} + (1-\arccos 3z)^2\,\frac{x}{e^y+1}$

При **x** = 2,85, **y** = $6{,}23\cdot10^{-3}$, **z** = 0,14. **Ответ**: **v** = 1,3773

24. $m = \frac{\cos z - \frac{z}{y-x}}{1+(y-x)^2} + \cos\frac{x}{\sqrt{2}} + 5^{\sqrt{x}}\,\frac{x}{\sqrt{y}}$

При **x** = 0,49722, **y** = $1{,}3343\cdot10^{5}$, **z** = 0,234. **Ответ**: **m** = 0,9431

25. $p = \sqrt[3]{x^2}\,\frac{1-x}{1+y^2}\sin^3\cos^3 z + \frac{x}{\sqrt{1-x^4}}$

При **x** = 0,1231, **y** = $2{,}14\cdot10^{-3}$, **z** = 1,784. **Ответ**: **p** = 0,1212

26. $n = \frac{e^{arctgz} + y\ln(1+x)^2 + 1}{1+y^2} + \frac{y^2}{\sqrt[3]{x^2}}$

При **x** = $3{,}791\cdot10^{4}$, **y** = $8{,}412\cdot10^{-2}$, **z** = 1,683. **Ответ**: **n** = 5,5487

27. $q = \frac{\sin\cos y}{\sqrt{2-\sin^4 z}} + \frac{y^2}{x^2-2}\cdot e^x\sqrt{x-y^x}$

При **x** = 8,634, **y** = $1{,}52\cdot10^{-1}$, **z** = 0,437. **Ответ**: **q** = 5,5569

Библиотека БГУИР

28. $k = \ln\dfrac{1+\sqrt{\sin z}}{1-\sqrt{x+y}} + 2arctgz - \ln\dfrac{|x-2|}{(y+2)^3}$

При **x** = $6{,}2\cdot10^4$, **y** = $8{,}234\cdot10^{-3}$, **z** = 1,6211. **Ответ**: **k**= –11,7275

29. $d = \sqrt[3]{\sin^2 z} + \dfrac{1}{\cos^2 z}\cdot\dfrac{\sqrt{x^3+1}}{y^4+2x+|y-5|}$

При **x** = $31{,}14\cdot10^{-2}$, **y** = 2,673, **z** = 0,245. **Ответ**: **d** = 0,4089

30. $b = \dfrac{1}{\sqrt{3}}\ln\dfrac{\left|tg\dfrac{z}{2}+2-y^3\right|}{x^3+2+tg\dfrac{y}{2}}\cdot\dfrac{x}{e^x+1}+\dfrac{y}{y+3}$

При **x** = 42,26, **y** = $17{,}78\cdot10^3$, **z** = 4,45. **Ответ**: **b** = 0,9998

2.5. Задания повышенной сложности

1. Вычислить значение многочлена $7x^6 - 3x^5 + 8x^3 + 2x^2 + 6$ за минимально возможное количество операций.
2. Найти сумму и произведение цифр заданного трехзначного числа.
3. Определить среднее арифметическое цифр данного четырехзначного числа
4. Получить число, обратное заданному трехзначному числу (*например* 123 → 321).
5. Присвоить переменной k вторую с конца цифру в записи целого числа n.
6. Поменять местами первую и последнюю цифры в трехзначном числе.
7. Присвоить переменной k первую цифру дробной части вещественного числа.
8. Поменять местами значения переменных x и y, не используя дополнительных.
9. Из заданного четырехзначного числа получить двузначное число, удалив из исходного четырехзначного числа цифры сотен и единиц (*например* 2783 → 28).
10. Дана сторона равностороннего треугольника. Найти его площадь. Длину стороны задать с помощью генератора случайных чисел.
11. Из круга радиусом r вырезан прямоугольник, большая сторона которого равна a. Найти максимальный радиус круга, который можно вырезать из полученного прямоугольника.
12. Определить номер недели, если с начала года прошло k дней (1 января было в понедельник).

Библиотека БГУИР

13. Определить количество полных часов (h) и минут (m), если идет k-я секунда суток.

14. Определить угол в градусах (f) между положением часовой стрелки в начале суток и ее положением в h часов, m минут, s секунд.

15. Определить полное количество часов (h) и минут (m), прошедших от начала суток до того времени (в первой половине дня), когда часовая стрелка повернулась на f градусов ($0 \le f < 360$, f – вещественное число).

ЛАБОРАТОРНАЯ РАБОТА №3
ПРОГРАММИРОВАНИЕ РАЗВЕТВЛЯЮЩИХСЯ АЛГОРИТМОВ

Цель работы: научиться пользоваться простейшими компонентами организации переключений (TCheckBox, TRadioGroup). Составить блок-схему, написать и отладить программу разветвляющегося алгоритма.

3.1. Операции сравнения и логические операции

Операции сравнения используются при работе с **двумя** операндами и возвращают значение True (1), если результат сравнения – истина, и False (0), если результат сравнения – ложь. В языке Pascal определены следующие **операции сравнения**: **<** (меньше**), <=** (меньше или равно), **>** (больше), **>=** (больше или равно), **=** (равно), **<>** (не равно).

Логические операции применяются только к данным логического типа (**`Boolean`**). Существуют следующие **логические операции**:

And – **логическое** «**и**» (пересечение),
Or – **логическое** «**или**» (объединение),
Xor – **исключающее** «**или**»,
Not – **отрицание** или **логическое** «**не**».

Примеры:

$0 \le y \le 20$	=>	`(0<=y)And(y<=20)`
$x \ne 0$ или $y = 0$	=>	`(x<>y)Or(y = 0)`
x – четное	=>	`Not Odd(x)`
y кратно 3	=>	`y Mod 3 = 0`

3.2. Оператор условной передачи управления If

Оператор `If` проверяет результат логического выражения или значение переменной типа `Boolean` и организует разветвление вычислений.

Форматы оператора `If`.

1. Полная форма:

```
If условие Then оператор_1
           Else оператор_2;
```

Если условие (логическое выражение*)* истинно, то выполняется оператор_1, иначе – оператор_2.

Пример: найти наибольшее из значений a и b, т. е. Max (a, b):

```
If a>b Then Max:=a
       Else Max:=b;
```

Библиотека БГУИР

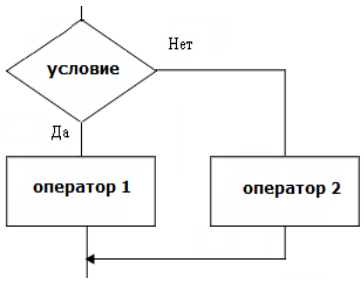
2. Сокращенная форма:

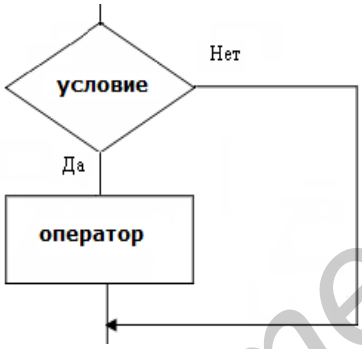
`If` условие `Then` оператор`;`

Если условие (логическое выражение*)* истинно, то выполняется оператор.

Пример: найти |a|

```
If a<0 Then a:=-a;
```





Библиотека БГУИР

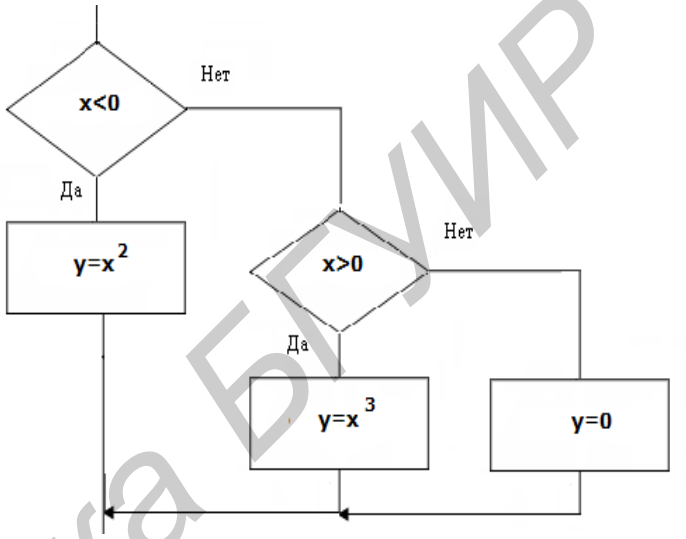
3. Вложенная форма:

```
If условие_1 Then оператор_1
   Else If условие_2 Then оператор_2
                        Else оператор_3;
```

Если условие_1 истинно, то выполняется оператор_1, иначе, если условие_2 истинно, то выполняется оператор_2, иначе выполняется оператор_3.

Например вычислить выражения:

1. $$y=\begin{cases} x^2, & x<0 \\ x^3, & x>0 \\ 0, & x=0 \end{cases}$$




```
If x<0 Then y:=Sqr(x)
   Else If x>0 Then
                y:=Power(x,3)
                  Else y:=0;
```

2. $$s=\begin{cases} \ln x+\sqrt[3]{|y|}, & x/y>0 \\ \ln|x/y|\cdot(x+y)^3, & x/y\le 10 \\ (x^2+y)^3, & \text{иначе} \end{cases}$$

Поскольку никаких ограничений на ввод значения переменной y по условию примера не предусмотрено, то при вводе значения y = 0 может возникнуть ситуация **деление на** 0. Поэтому следует записывать оператор If, начиная с ветви **иначе**:

```
If (x = 0)Or(y = 0) Then s:=Power(Sqr(x)+y,3)    // x = 0 или y = 0
  Else If x/y>0 Then s:=Ln(x)+Power(Abs(y),1/3) //иначе x / y > 0
                  Else s:=Ln(Abs(x/y))*Power(x+y,3);  // x / y < 0
```

3.3. Оператор выбора Case

Оператор Case применяется для выбора одного из возможных вариантов.

Формат оператора:

```
Case переменная выбора Of
   значение 1: оператор 1;
   значение 2: оператор 2;
   ...
   значение n: оператор n;
   Else  оператор n+1
End;
```

переменная выбора, значение 1, ..., значение n – константа, переменная или выражение **порядкового** типа: целого, символьного, логического, интервально-

Библиотека БГУИР

го или перечисляемого. Значения переменной выбора и значение 1, ..., значение n должны быть **одинакового типа**.

При выполнении оператора `Case` сначала вычисляется значение переменной выбора, которое последовательно сравнивается со значением 1, ..., значением n, и при совпадении значений выполняется соответствующий оператор, иначе выполняется оператор n+1. Конструкция `Else` может отсутствовать.

Пример: выбор вида функции `f(x)`: x^2, $|x|$, $\sqrt{x}$

```
Case k Of
  0: f:=Sqr(x);
  1: f:=Abs(x);
  2: f:=Sqrt(x);
  Else Begin
         Memo1.Lines.Add('F(x) не задана!');
         Exit;
       End;
End;
```

3.4. Оператор безусловной передачи управления GoTo

Оператор безусловной передачи управления имеет следующий вид:

`GoTo` метка*;*

Метка – это идентификатор, описанный в разделе меток `Label` следующим образом: `Label` метка1, метка2, …;

Например:

```
Label M1;
…
GoTo M1;
…
M1: y:=Sin(x);
…
```

Оператор `GoTo` передает управление оператору с указанной меткой, в данном случае – `M1` (все метки, используемые в программе, должны быть описаны). Оператор, следующий за `GoTo`, обязательно должен иметь метку, иначе он никогда не получит управление. По возможности следует избегать использования оператора `GoTo`, т. к. это приводит к созданию неэффективных программ. Переход внутри программы приводит к тому, что надо заново обновлять очередь команд, готовых к выполнению в процессоре, и перенастраивать управляющие регистры. Чем меньше в программе операторов `GoTo`, тем выше квалификация программиста.

3.5. Кнопки-переключатели в Delphi

При написании программ для организации разветвлений часто используются компоненты в виде кнопок-переключателей. Состояние такой кнопки (включено – выключено) визуально отражается на форме. На форме (рис. 3.1) представлены кнопки-переключатели двух типов: TCheckBox и TRadioGroup.

Компонент TCheckBox организует кнопку **независимого** переключателя, с помощью которой можно задать свой вариант решения (типа **да**/**нет**). В про-

Библиотека БГУИР

грамме состояние кнопки связано со значением логической переменной, проверяемой оператором `If`.

Компонент TRadiogroup организует группу кнопок – **зависимых** переключателей. При нажатии одной из них все остальные кнопки отключаются. В программу передается номер включенной кнопки (0, 1, 2, ...), анализируемый оператором `Case`.

3.6. Порядок выполнения задания

Вычислить значение выражения

$$s = \begin{cases} |f(x)\cos x| + \ln y, & |x \cdot y| > 10 \\ e^{2f(x)+y}, & 3 < |x \cdot y| \le 10 \\ \sqrt{|f(x)|} + 2tgy & \text{иначе} \end{cases}$$

При выполнении задания предусмотреть:
– выбор вида функции f(x): sh(x), x^2 или e^x;
– вывод информации о выбранной ветви вычислений;
– возможность округления полученного результата.
Составить блок-схему алгоритма.

3.6.1. Создание формы проекта

Разместите на форме необходимые компоненты в соответствии с рис. 3.1.

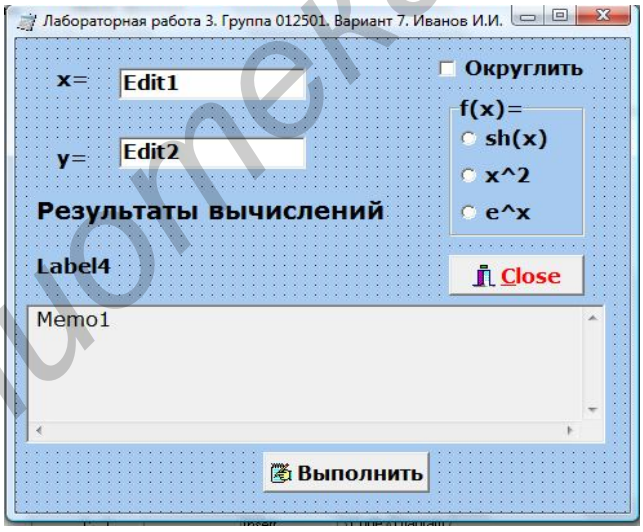


Рис. 3.1. Форма проекта

Для этого:

1) подпишите форму (свойство Caption);

2) установите на форму компоненты TLabel для вывода пояснений и информации о том, по какой ветви происходило вычисление;

3) установите на форму компоненты TEdit для ввода значений переменных x и y;

4) установите на форму компонент TMemo для вывода результата, добавьте в него обе линейки прокрутки (свойства ScrollBars – ssBoth);

5) установите на форму две кнопки TBitBtn: для одной из них задайте свойства Kind – bkClose (стандартная кнопка, закрывающая проект), другую

Библиотека БГУИР

кнопку подпишите Выполнить (свойство Caption), установите на нее значок (свойство Glyph – кнопка Load – выберите и загрузите файл с расширением *.bmp);

6) измените для компонентов TEdit, TBitBtn, TMemo вид курсора на crHandPoint (свойство DragCursor);

7) измените размеры и цвет формы (свойство Color), задайте шрифт для компонентов (свойство Font).

3.6.2. Работа с компонентом TCheckBox

Выберите в меню компонентов Standard пиктограмму  и поместите ее в нужное место формы. С помощью инспектора объектов измените заголовок (Caption) на Округлить. В коде программы появится переменная CheckBox1 типа TCheckBox. В зависимости от того, нажата кнопка или нет, логическая переменная CheckBox1.Checked будет принимать значения True или False.

3.6.3. Работа с компонентом TRadioGroup

Выберите в меню компонент Standard пиктограмму  и поместите ее в нужное место формы. На форме появится окаймленный линией чистый прямоугольник с заголовком RadioGroup1. Замените заголовок (Caption) на F(x). Для размещения на компоненте кнопок установите свойство Columns равным единице (кнопки размещаются в одном столбце). 2ЛКМ по правой части свойства Items – появится строчный редактор списка заголовков кнопок. Наберите три строки с именами: в первой строке – sh(*x*), во второй – x^2, в третьей – e^x, нажмите OK. На форме внутри окаймления появятся три кнопки-переключателя с введенными надписями.

Обратите внимание на то, что в коде программы появилась переменная RadioGroup1 типа TRadioGroup. При нажатии одной из кнопок группы в переменной целого типа RadioGroup1.ItemIndex будет находиться ее номер (отсчитывается от нуля), что используется в коде приведенной ниже программы.

3.6.4. Создание обработчиков событий FormCreate и ButtonClick

Создайте обработчик события FormCreate (2ЛКМ в свободном месте формы). На экране появится код, в котором автоматически создастся заголовок процедуры – обработчика события создания формы

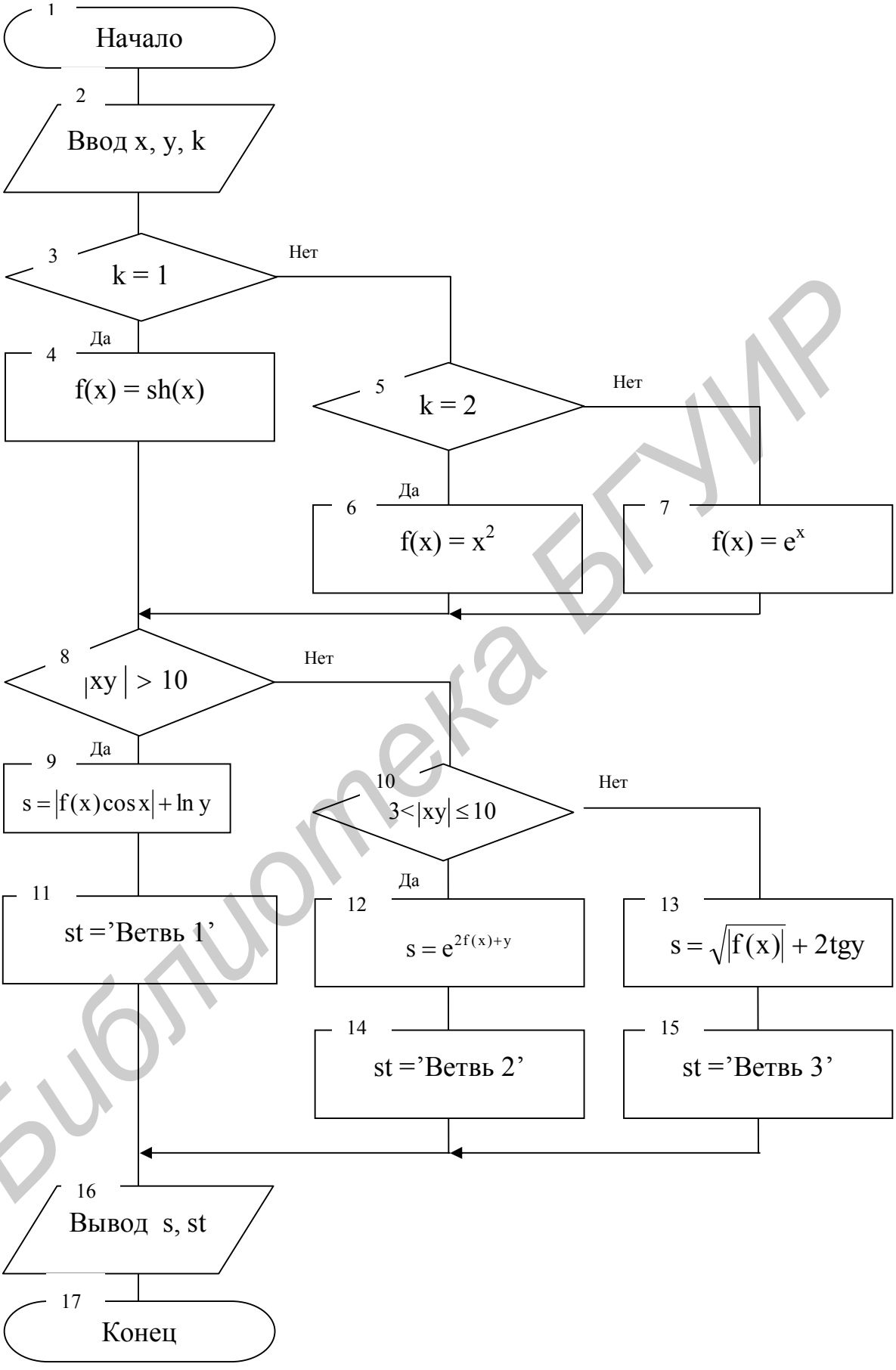
```
Procedure TForm1.FormCreate(Sender:TObject);
```

Между `begin...end` вставьте код программы, соответствующий вводу начальных значений переменных x и y из соответствующих строк ввода TEdit, очистке от надписи компонента Label4, выводу в окно TMemo строки с указанием темы лабораторной работы, номера группы и фамилии студента.

Обработчик события нажатия кнопки ButtonClick создается аналогично (`Procedure TForm1.ButtonClick(Sender:TObject);`).

Ниже представлена блок-схема алгоритма (рис. 3.2).

Библиотека БГУИР



Библиотека БГУИР

Рис. 3.2. Блок-схема алгоритма

3.6.5. Код программы

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics,
  Controls, Forms, Dialogs, StdCtrls, ExtCtrls;
type
 TForm1 = class(TForm)
  CheckBox1: TCheckBox;
  RadioGroup1: TRadioGroup;
  Memo1: TMemo;
  Button1: TButton;
  Edit1: TEdit;
  Label1: TLabel;
  Label2: TLabel;
  Edit2: TEdit;
  Label3: TLabel;
  Edit3: TEdit;
  procedure FormCreate(Sender: TObject);
  procedure Button1Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;
var
 Form1: TForm1;
implementation
{$R *.DFM}
procedure TForm1.FormCreate(Sender: TObject);
begin
 Edit1.Text:='0,1';     // Начальное значение X
 Edit2.Text:='0,793'; // Начальное значение Y
 Label4.Caption:=' ';//Очистка компонента для вывода ветви вычисления
 Memo1.Clear;           // Очистка окна редактора Memo1
 // Вывод строк текста в многострочный редактор Memo1
 Memo1.Lines.Add ('Программирование разветвляющихся алгоритмов');
 Memo1.Lines.Add('Выполнил студент группы 012501 Иванов И.И.');
 RadioGroup1.ItemIndex:=1;          // Выбор функции f(x): x2
end;
procedure TForm1.Button1Click(Sender: TObject);
Var                // Описание всех переменных, используемых в программе
 x, y, xy, s, f: Extended;
 st: String;    // Строка для вывода сообщения номера ветви вычисления
begin
 x:=StrToFloat(Edit1.Text);              // Ввод исходных данных и
```

Библиотека БГУИР

```
Memo1.Lines.Add('x = '+Edit1.Text);      // их вывод в окно Memo1
y:=StrToFloat(Edit2.Text);
Memo1.Lines.Add('y = '+Edit2.Text);
Case RadioGroup1.ItemIndex Of // Проверка номера нажатой кнопки
    0: f:=sinh (x);           // и выбор соответствующей ей функции
    1: f:=sqr(x);
    2: f:=exp(x);
End;
xy:=x*y; // Часто встречающиеся вычисления обозначаются отдельно
If Abs(xy)>10 Then Begin  // В зависимости от условия вычисляется
                    s:=Abs(f*Cos(x))+Ln(y);//соответствующее
                    st:='Ветвь 1';//ему выражение и запоминается
                  End          // номер  ветви вычисления
       Else If (3>xy)And(xy<=10) Then Begin
                                        s:=Exp(2*f+y);
                                        st:='Ветвь 2';
                                      End
                                Else Begin
                                       s:=Sqrt(Abs(f))+2*Tan(y);
                                       st:='Ветвь 3';
                                     End;
Label4.Caption:=st;   // Вывод сообщения о ветви вычисления в Label4
If CheckBox1.Checked Then     // Вывод результата в окно Memo1
          Memo1.Lines.Add('Rezult s = '+IntToStr(Round(s)))
 Else Memo1.Lines.Add('Rezult s ='+FloatToStrF(s,ffGeneral,8,2));
end;  // Если CheckBox1 включен, то результат округляется до целой части,
end.  // иначе выводится с точностью до двух знаков в дробной части.
```

3.7. Индивидуальные задания

1. Составить программу вычисления выражения согласно указанному варианту.
2. При выполнении задания предусмотреть:
   – выбор вида функции f(x): x^2, e^x, sh(x);
   – вывод информации о выбранной ветви вычислений в компонент TLabel;
   – возможность округления результата.

Составить блок-схему алгоритма.

1. $$s=\begin{cases}(f(x)+y)^2-\sqrt[3]{|f(x)|}, & x\cdot y>0\\ (f(x)+y)^2+\sin(x), & x\cdot y<0\\ (f(x)+y)^2+y^3, & x\cdot y=0\end{cases}$$

2. $$s=\begin{cases}\ln(f(x))+\sqrt[3]{|f(x)|}, & x/y>0\\ \ln|f(x)/y|\cdot(x+y)^3, & x/y<0\\ (f(x)^2+y)^3 & \text{иначе}\end{cases}$$

Библиотека БГУИР

3. $s=\begin{cases} f(x)^2+\sqrt[3]{y}+\sin y, & x-y=0 \\ (f(x)-y)^2+\ln|x|, & x-y>0 \\ (y-f(x))^2+tgy & \text{иначе} \end{cases}$

4. $s=\begin{cases} \sqrt[3]{|f(x)-y|}+tg(f(x)), & x>y \\ (y-f(x))^3+\cos(f(x)), & x<y \\ (y+f(x))^2+x^3 & \text{иначе} \end{cases}$

5. $s=\begin{cases} \ln(f(x))+(f^2(x)+y)^3, & x/y>0 \\ \ln|f(x)/y|+(f(x)+y)^3, & x/y<0 \\ (f^2(x)+y)^3, & x=0 \\ 0 & \text{иначе} \end{cases}$

6. $s=\begin{cases} e^{f(x)}, & 1<x\cdot b<10 \\ \sqrt[3]{|f(x)+4y|}, & 12<x\cdot b<40 \\ y\cdot f(x)^2 & \text{иначе} \end{cases}$

7. $s=\begin{cases} e^{f(x)-|y|}, & 0{,}5<x\cdot y<10 \\ \sqrt[3]{|f(x)+y|}, & 0{,}1<x\cdot y<0{,}5 \\ 2f(x)^2 & \text{иначе} \end{cases}$

8. $s=\begin{cases} (f(x)^2+y)^3, & x/y<0 \\ \ln|f(x)/y|+x/y, & x/y>0 \\ \sqrt[3]{|\sin y|} & \text{иначе} \end{cases}$

9. $s=\begin{cases} 2f(x)^3+3y^2, & x>|y| \\ |f(x)-y|, & 3<x<|y| \\ \sqrt[3]{|f(x)-y|} & \text{иначе} \end{cases}$

10. $s=\begin{cases} \ln(|f(x)|+|y|), & |x\cdot y|>10 \\ e^{f(x)+y}, & |x\cdot y|<10 \\ \sqrt[3]{|f(x)|}+y & \text{иначе} \end{cases}$

11. $s=\begin{cases} \ln|f^2(x)\cdot y|, & x\cdot y>0 \\ f^3(x)+\dfrac{x}{y^2}, & x\cdot y<0 \\ tg|f(x)|+\sqrt[3]{y^2} & \text{иначе} \end{cases}$

12. $s=\begin{cases} tg(x)+f(x)^2, & y>2x \\ |f(x)+y|^3, & y<2x \\ \sqrt[3]{x}\cdot\sin(x), & \text{иначе} \end{cases}$

13. $s=\begin{cases} (f(x)+\ln(|y|))^3, & x/y>0 \\ 2/3+\ln(|\sin(y)|), & x/y<0 \\ \sqrt[3]{f(x)^2}+y & \text{иначе} \end{cases}$

14. $s=\begin{cases} \ln(f(x))^3, & x^3>0 \\ tg(x^3)+f(x), & x^3<0 \\ \sqrt[3]{|y^3-x^2|} & \text{иначе} \end{cases}$

15. $s=\begin{cases} \left(f^2(x)+y^3\right)/x, & f(x)>0 \\ \ln|f^3(x)|+\cos y, & f(x)<0 \\ \sqrt[3]{\sin^2 y} & \text{иначе} \end{cases}$

16. $s=\begin{cases} y\sqrt{f(x)}, & y-\text{четное}, \quad x>0 \\ y/2\sqrt{|f(x)|}, & y-\text{нечетное}, x<0 \\ \sqrt{|yf(x)|} & \text{иначе} \end{cases}$

17. $s=\begin{cases} y\sqrt{|f(x)|}+3\sin(x), & x>y \\ x\sqrt{|f(x)|}, & x<y \\ \sqrt[3]{|f(x)|}+x^3/y & \text{иначе} \end{cases}$

18. $s=\begin{cases} \dfrac{5}{6}f^2(x)+2y^2\sqrt{|f(x)|}, & x\cdot y<0 \\ \cos\left(\dfrac{3f(x)}{2y}+5\sqrt[3]{y^2}\right), & x\cdot y>0 \\ (f^4(x)+3y)^2 & \text{иначе} \end{cases}$

Библиотека БГУИР

19. $s=\begin{cases}(f(x)-2\cos^2 y^3)e^{f(x)+y}, & -1<y<1\\ \sqrt{\left(\left|\dfrac{3-f(x)}{y^2f^4(x)}\right|+5\right)^3}-e, & x\cdot y>0\\ \ln^3 f^2(x)+1 & \text{иначе}\end{cases}$

20. $s=\begin{cases}\ln^3\left|y^2+f(x)\right|, & x>y\\ \sqrt{\left(\sqrt[3]{|f(x)|}+\dfrac{2y+1}{y^3+2}\right)^3}-1, & y>x\\ (f^3(x)-tgy)^2 & \text{иначе}\end{cases}$

21. $s=\begin{cases}\dfrac{1}{3}f^2(x)-\sqrt{2y}, & 10<y<2x\\ e^{xy^2}-\left|x^3+\sqrt[3]{3f(x)}\right|, & y>0 \text{ и } x>0\\ \cos(f(x)+\sqrt[3]{y^2}) & \text{иначе}\end{cases}$

22. $s=\begin{cases}ctg^2(y^3+1), & x\cdot y=0\\ \dfrac{1}{2}e^{f^2(x)}\dfrac{\sqrt{|f(x)|}}{3y^3}, & y\cdot x>0\\ \ln(f(x)+3)^2 & \text{иначе}\end{cases}$

23. $s=\begin{cases}\sin\left(5f(x)+3y|f(x)|\right), & -1<y<x\\ \cos\left(3f(x)+5y|f(x)|\right), & x>y\\ \left(f(x)+y\right)^2, & x=y.\end{cases}$

24. $s=\begin{cases}\sin^3\left(y^2|f(x)|\right), & -2<y<x\\ e^{\cos(2y|f(x)|)}-\ln(x^2y^2), & x>y\\ \sqrt{|f(x)+y|}+2 & \text{иначе}\end{cases}$

25. $s=\begin{cases}\sqrt{f^2(x)+y^3}-\dfrac{1+x}{2y}, & 2y>0\\ e^{|f^3(x)+1|}+\cos|x-y|, & 2y<0\\ \sqrt[3]{f^2(x)}-2|x-y| & \text{иначе}\end{cases}$

26. $s=\begin{cases}2\ln\left|f(x)+3y^2\right|, & y\cdot f(x)<0\\ \sin\left(\dfrac{2f(x)}{3y|f^3(x)|}+1\right), & y\cdot f(x)>0\\ e^{(f(x)+y)^2} & \text{иначе}\end{cases}$

27. $s=\begin{cases}\dfrac{1-(\sqrt{f(x)}+y^3)^2}{2f(x)}, & f(x)>0\\ \ln\left|f^3(x)+\cos^2 y\right|, & f(x)<0\\ \sqrt[3]{x^2y^2+1}+e^{-2y} & \text{иначе}\end{cases}$

28. $s=\begin{cases}\ln(\left|y\cdot f^2(x)\right|+2), & -2<y<2\\ e^{\left(\frac{3}{2}f(x)-y\right)^2}, & x>y\\ \sqrt[3]{f^2(x)+2|y|} & \text{иначе}\end{cases}$

29. $s=\begin{cases}\ln|1+\cos y|, & y>0\\ \dfrac{x^2-e^{y^4}}{\sqrt{|f(x)+\sqrt[3]{y^2}|}}, & 1<y\cdot f(x)<10\\ tg(f^3(x)) & \text{иначе}\end{cases}$

30. $s=\begin{cases}\dfrac{f^2(x)+1}{2}+\dfrac{|y^3|}{x+1}, & x>0\\ \ln\left|f^3(x)+\dfrac{2}{3}\right|+e^{2\cos y}, & x<0\\ \sqrt[3]{f^2(x)}-\left(\dfrac{y}{2}\right)^2, & \text{иначе}\end{cases}$

Библиотека БГУИР

3.8. Задания повышенной сложности

1. Вычислить $r = \max(\min(f(x), y), z)$.

2. Если сумма трех попарно различных действительных чисел x, y, z меньше единицы, то наименьшее из этих трех чисел заменить полусуммой двух других; в противном случае заменить меньшее из x и y полусуммой двух оставшихся значений.

3. Удвоить числа x, y, z, если они упорядочены по возрастанию, утроить, если – по убыванию, а иначе четные возвести в степень наименьшего из значений.

4. Проверить, могут ли числа a, b, c быть датой (*например* 3, 7, 1972 – дата). Если могут, то вывести полученную дату.

5. Значения переменных a, b и c поменять местами так, чтобы оказалось $a \le b \le c$.

6. Определить для заданного номера года, является он високосным или нет (високосным является год, если его номер делится на 4, за исключением тех, которые делятся на 100 и не делятся на 400).

7. Вывести значения D (день) и M (месяц) для правильной даты невисокосного года, следующей за заданной.

8. Дано натуральное число. Проверить, является ли оно четырехзначным палиндромом (первая цифра равна четвертой, а вторая – третьей).

9. Проверить, является ли натуральное шестизначное число «счастливым», в котором сумма первых трех его цифр равна сумме трех последних.

10. Действительные числа x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4 являются координатами вершин прямоугольника со сторонами, параллельными осям координат. Определить, принадлежит ли начало координат этому прямоугольнику.

11. Выяснить, пройдет ли кирпич с ребрами a, b, c в прямоугольное отверстие со сторонами x и y. Кирпич в отверстие надо просовывать так, чтобы каждое из его ребер было параллельно или перпендикулярно каждой из сторон отверстия.

12. Определить, какая цифра находится в k-й ($1 \le k \le 180$) позиции последовательности 10111213…9899, в которой выписаны подряд все двузначные числа.

13. Дано натуральное k. Определить k-ю цифру в последовательности 11010010001000010000…, в которой выписаны подряд степени 10.

14. Вывести фразу «*мне* k *лет*» ($1 \le k \le 99$), учитывая при этом, что при некоторых значениях k слово *«лет»* надо заменить на слово «*год*» или «*года*».

15. Вывести строку – словесное описание данного числа x ($100 \le x \le 999$), *например* 256 – «двести пятьдесят шесть», 814 – «восемьсот четырнадцать».

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №4

ПРОГРАММИРОВАНИЕ ЦИКЛИЧЕСКИХ АЛГОРИТМОВ. ОТЛАДКА ПРОГРАММ

Цель работы: изучить простейшие средства отладки программ в среде Delphi. Составить блок-схему, написать и отладить программу циклического алгоритма.

4.1. Операторы организации циклов

Под **циклом** понимают многократное выполнение одних и тех же операторов при различных значениях промежуточных данных. Число повторений цикла может быть задано в **явной** или **неявной форме**.

Для организации повторений в Delphi предусмотрены три оператора цикла.

4.1.1.Оператор цикла – For

Оператор применяется, если **известно количество повторений**, и переменная цикла имеет **порядковый тип**.

Существует **два** варианта оператора `For`:

С шагом **1**:

`For` переменная цикла:= нач. значение `To` кон. значение `Do`
оператор;

переменная цикла должна быть **порядкового** типа;

нач. значение и кон. значение – начальное и конечное значения переменной цикла **по типу совместимые** с ней, иначе при неявном приведении типов возможны ошибки; могут быть константами, переменными, выражениями.

Для выполнения цикла должно выполняться условие:

нач. значение ≤ кон. значение

Выполнение цикла `For`:

1. Вычисляется нач. значение и присваивается переменной цикла.
2. Проверяется условие выполнения цикла: нач. значение ≤ кон. значение.
3. Если условие истинно (выполняется), то выполняется оператор и пп. 4–5, иначе (условие ложно) цикл завершается, и выполняется оператор, стоящий после цикла.
4. Значение переменной цикла увеличивается на единицу.
5. Происходит переход к п. 2.

С шагом **–1**: переменная цикла изменяется от большего значения к меньшему:

`For` переменная цикла:=нач. значение `DownTo` кон. значение `Do`
оператор;

`DownTo` – «вниз к»;

нач. значение, кон. значение – так же, как и с шагом 1.

Для выполнения цикла надо, чтобы нач. значение ≥ кон. значение.

Попытки использовать переменную цикла сразу после завершения цикла могут привести к непредсказуемому поведению программы при отладке, т. к. переменная цикла может потерять свое значение.

Библиотека БГУИР

Пример вычисления значения факториала n! = 1 · 2 · 3 · … · n

```
F:=1;
For i:=1 To n Do
  F:=F*i;
```

```
F:=1;
For i:=n DownTo 1 Do
     F:=F*i;
```

Фрагменты блок-схемы вычисления факториала приведены на рис. 4.1–4.2.

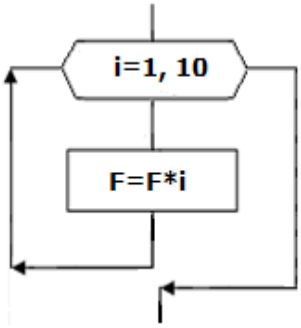


Рис. 4.1. Цикл `For` с шагом 1

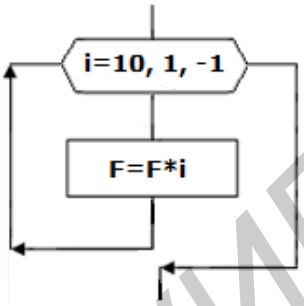


Рис. 4.2. Цикл `For` с шагом –1

Правила использования цикла `For`.

1. Внутри цикла нельзя изменять значения переменной цикла и переменных, входящих в начальное и конечное значения цикла.

2. Циклы можно вкладывать друг в друга, но нельзя их пересекать.

3. Можно передавать управление из цикла, но нельзя передавать управление внутрь цикла.

Если заранее неизвестно, сколько раз необходимо выполнить операторы цикла, то удобнее применять **циклы с предусловием** `While` или **с постусловием** `Repeat-Until`.

С помощью этих циклов можно запрограммировать любые повторяющиеся фрагменты алгоритмов, которые на практике чаще всего используют, когда число повторений заранее:

1) неизвестно (*например* цикл до достижения требуемой точности результата и т. п.);

2) известно, но шаг переменной цикла не равен 1 или переменная цикла не является переменной порядкового типа.

4.1.2. Оператор цикла с предусловием While

Формат оператора: `While` логическое выражение `Do` оператор;

`While` – «пока, в то время как»;

логическое выражение – условие выполнения цикла; может быть переменной, константой или выражением, имеющим логический тип.

Выполнение цикла `While`.

1. Вычисляется значение условия выполнения цикла.

2. Если оно истинно (True), то выполняется оператор цикла и происходит переход на п. 1, иначе (False) цикл завершается.

Цикл `While` обеспечивает выполнение оператора цикла до тех пор, пока условие имеет значение True (истина).

Библиотека БГУИР

Обычно в операторе цикла изменяется значение переменной, которая входит в логическое выражение.

Условие проверяется перед началом каждого выполнения цикла. Поэтому, если до первого выполнения цикла условие имеет значение False (ложь), оператор цикла **не выполнится ни одного раза**.

Если в цикле содержится не один, а **несколько операторов**, то они заключаются в «операторные скобки» `Begin … End`.

Примеры вычисления (рис. 4.3):

1) суммы $S=\sum_{i=1}^{n} i^2 = 1^2+2^2+3^2+...+9^2+10^2$

```
S:=0;
i:=0;
While i<n Do
Begin
   Inc(i);
   S:=S+i*i;
End;
```

2) факториала $n!=1\cdot 2\cdot 3\cdot ...\cdot n$

```
F:=1;
i:=0;
While i<n Do
    Begin
       Inc(i);
       F:=F*i;
    End;
```

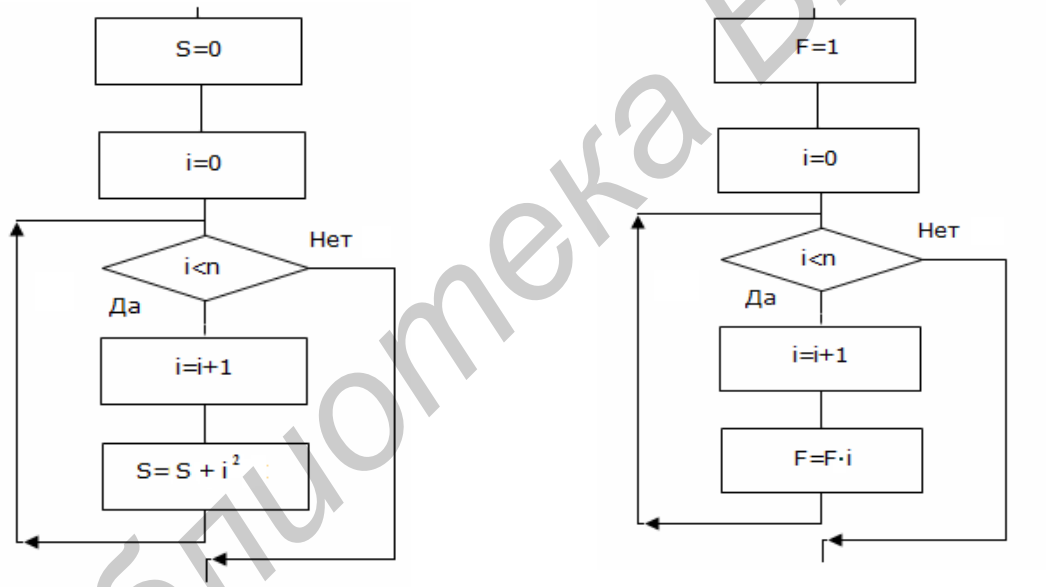


Рис. 4.3. Фрагменты блок-схем вычисления суммы и факториала

4.1.3. Оператор цикла с постусловием Repeat … Until

Формат оператора:

```
Repeat
      оператор1;
      ...
Until логическое выражение;
```

логическое выражение – условие окончания цикла.

`Repeat…Until` – «повторять до тех пор, пока»

Выполнение оператора цикла `Repeat … Until`.

1. Выполняются операторы цикла.

Библиотека БГУИР

2. Вычисляется значение условия: если оно ложно (False), то происходит переход на пункт 1, иначе (True) цикл завершается.

Примеры вычисления значений (рис. 4.4):

1) суммы $S = \sum_{i=1}^{n} i^2 = 1^2 + 2^2 + \ldots + 10^2$

```
S:=0;
i:=0;
Repeat
     Inc(i);
     S:=S+i*i;
Until i=n;
```

2) факториала n! = 1 · 2 · 3 · … · n

```
F:=1;
i:=0;
Repeat
     Inc(i);
     F:=F*i;
Until i = n;
```

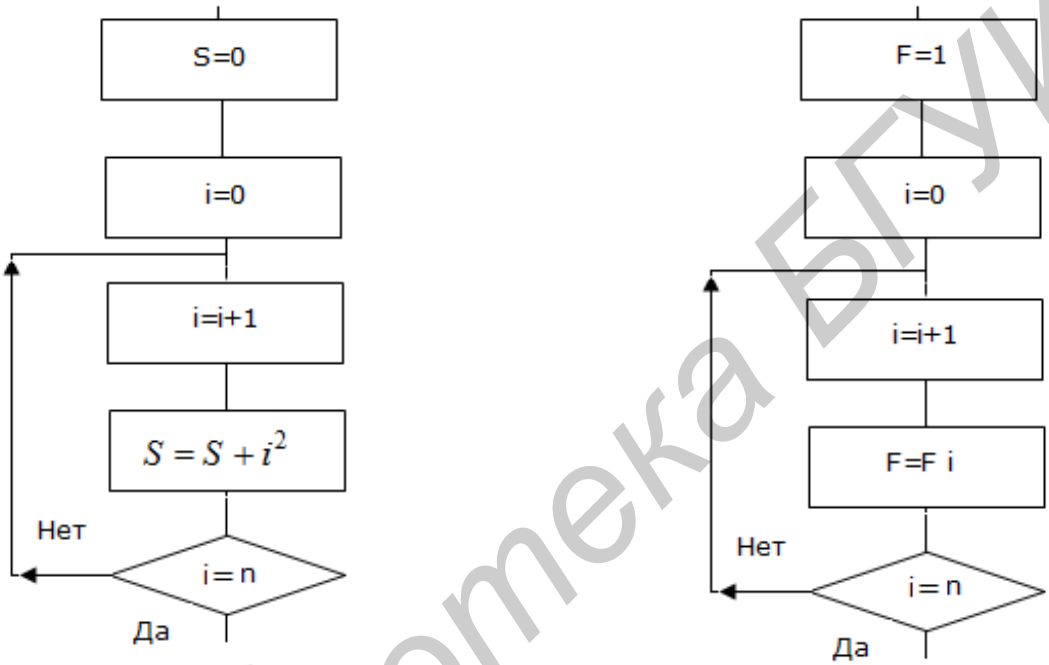


Рис. 4.4. Фрагменты блок-схем вычисления суммы и факториала

Отличия циклов `While` от `Repeat...Until` приведены в табл. 4.1.

Таблица 4.1

Отличия циклов `While` от `Repeat...Until`

Признак	`While`	`Repeat...Until`
Минимальное количество раз выполнения цикла	0	1
Значение условия при выполнении цикла	True	False
Местоположение условия	В начале цикла	В конце цикла
Применение конструкции `Begin…End`	Применяется, если в цикле более одного оператора	Не применяется

Библиотека БГУИР

4.2. Операторы управления

`Exit` и `Return` – прерывают выполнение подпрограммы и осуществляют принудительный выход из подпрограммы в вызывающую программу.

`Halt` – останавливает выполнение программы и возвращает управление операционной системе.

Для управления работой циклов используются операторы `Continue` и `Break`, которые можно вызывать только внутри цикла.

4.2.1. Оператор Break

Прерывает выполнение цикла и передает управление первому оператору, расположенному после цикла.

```
Repeat                          While логическое выражение Do
    оператор;                       Begin
        …                            оператор;
    Break;                           …
        …                            Break;
    оператор;                        …
Until логическое выражение;          оператор;
оператор;                       End;
                                оператор;
```

При прерывании работы цикла `For` с помощью `Break` переменная цикла сохраняет свое текущее значение.

4.2.2. Оператор Continue

Прерывает работу текущей итерации цикла и передает управление оператору проверки условия, пропуская оставшуюся часть цикла.

```
Repeat                          While логическое выражение Do
    оператор;                       Begin
        …                            оператор;
    Continue;                        …
        …                            Continue;
    оператор;                        …
Until логическое выражение;          оператор;
                                End;
```

4.3. Средства отладки программ в Delphi

Практически в каждой написанной программе обнаруживаются ошибки.

Ошибки первого уровня (**ошибки компиляции**) связаны с неправильной записью операторов. При ее обнаружении компилятор Delphi останавливается напротив первого оператора, в котором обнаружена ошибка. В нижней части экрана появляется текстовое окно, содержащее сведения обо всех ошибках, найденных в проекте. Каждая строка окна содержит имя файла, в котором найдена ошибка, номер строки с ошибкой и характер ошибки. Для быстрого перехода к

Библиотека БГУИР

ошибке – 2ЛКМ на строке с ее описанием. Для получения более полной информации о характере ошибки надо вызвать справку клавишей F1. Следует учитывать, что одна ошибка может повлечь за собой другие, которые исчезнут при ее исправлении. Поэтому рекомендуется исправлять ошибки последовательно сверху вниз и после исправления каждой ошибки компилировать программу снова.

Ошибки второго уровня (**ошибки выполнения**) связаны с ошибками алгоритма или с неправильной его программной реализацией. Эти ошибки приводят к неверному результату, переполнению, делению на ноль и т. д. Поэтому программу надо протестировать, т. е. выполнить расчеты при таких значениях исходных данных, для которых заранее известен результат. Если тестовые расчеты указывают на ошибку, то для ее поиска надо использовать встроенные средства отладки среды.

В простейшем случае для локализации места ошибки рекомендуется поступать следующим образом. В окне редактора программы установить курсор в строке перед подозрительным участком и нажать клавишу F4 (выполнение до курсора). Выполнение программы будет остановлено на строке с курсором. Для просмотра значения переменной нужно навести на нее курсор (на экране будет высвечено ее значение) или нажать одновременно Ctrl + F7 и в появившемся окне ввести ее имя (с помощью данного окна можно также изменить значение переменной во время выполнения программы). Нажимая клавишу F7 (пошаговое выполнение), можно построчно выполнять программу, контролируя значения переменных. При нахождении курсора внутри цикла после нажатия F4 вычисления останавливаются после каждого выполнения тела цикла. Для продолжения расчетов надо выбрать меню Run – Run.

Способы установки точек прерывания.

1. ЛКМ на левом краю окна редактирования: выбранная для остановки строка выделяется красной полосой, на ее левом краю появляется маленький значок.

2. При выполнении команд Run – Add Breakpoint – Source Breakpoint... появится диалоговая панель редактирования точек прерывания Edit Breakpoint. Можно задать параметр Condition, где ввести выражение, при истинности которого точка прерывания «сработает», иначе выполнение приложения не будет прервано при прохождении через эту строку, или количество проходов, после которых точка прерывания переходит в активное состояние.

4.4. Порядок выполнения задания

Написать и отладить программу, которая выводит таблицу значений функции $y(x) = e^{-x}$ и ее разложение в ряд $s(x) = \sum_{k=0}^{\infty} a^k = \sum_{k=0}^{\infty} (-1)^k \frac{x^k}{k!}$ для x, изменяющегося в интервале от x_n до x_k с шагом h. Функцию S(x) вычислить с точностью до 0,001. Вывести число итераций, необходимое для достижения заданной точности.

Библиотека БГУИР

При составлении алгоритма удобно использовать рекуррентную последовательность (каждое новое слагаемое зависит от одного или нескольких предыдущих). Для получения расчетной формулы рассмотрим значения слагаемого при разных значениях k:

при $k=0;\ a_0=1\frac{1}{1};$ при $k=2;\ a_2=1\frac{x\cdot x}{1\cdot 2};$

при $k=1;\ a_1=-1\frac{x}{1};$ при $k=3;\ a_3=-1\frac{x\cdot x\cdot x}{1\cdot 2\cdot 3}$ и т. д.

На каждом шаге слагаемое дополнительно умножается на $-\frac{x}{k}$. Исходя из этого формула рекуррентной последовательности будет иметь вид $a_k=-a_{k-1}\frac{x}{k}$.

Полученная формула позволяет избавиться от многократного вычисления факториала и возведения в степень. Если в выражении имеется нерекуррентая часть, то ее следует рассчитывать отдельно.

Панель диалога представлена на рис. 4.5.

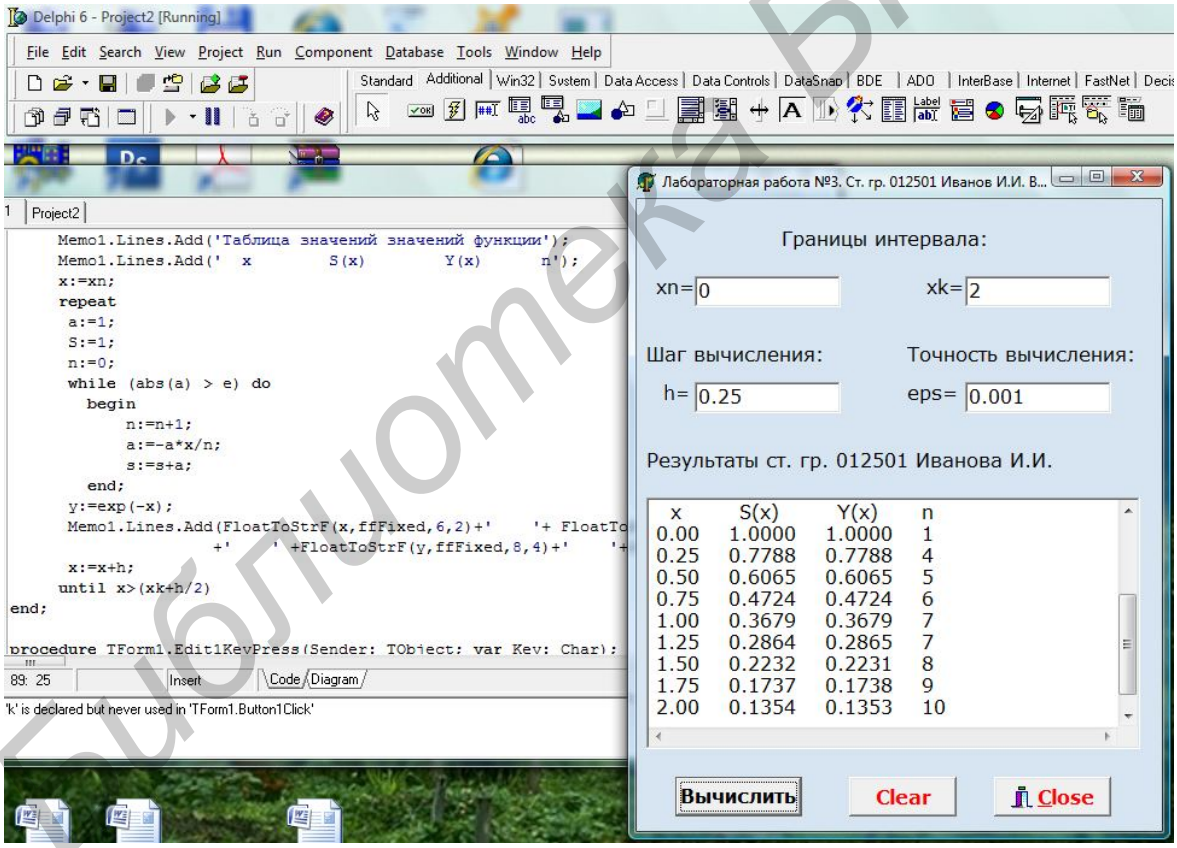
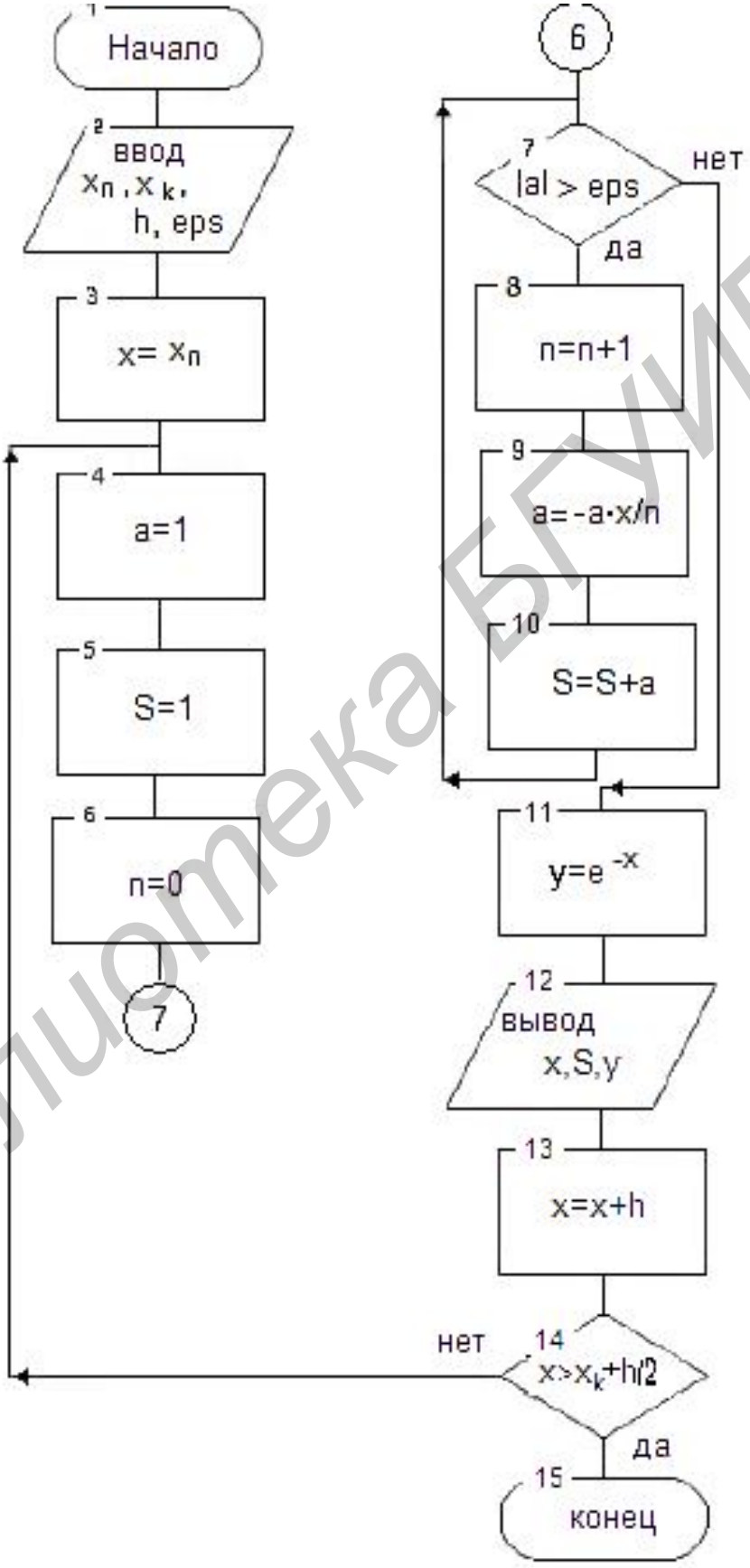


Рис. 4.5. Панель диалога

Библиотека БГУИР

4.4.1. Блок-схема алгоритма



Библиотека БГУИР

4.4.2. Код программы

```
unit Unit1;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes,
 Graphics, Controls, Forms, Dialogs, Buttons, StdCtrls;
type
 TForm1 = class(TForm)
  Edit1: TEdit;
  Edit2: TEdit;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  Edit3: TEdit;
  Label5: TLabel;
  Edit4: TEdit;
  Label6: TLabel;
  Memo1: TMemo;
  Label7: TLabel;
  Label8: TLabel;
  Button1: TButton;
  BitBtn1: TBitBtn;
  BitBtn2: TBitBtn;
  procedure FormCreate(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure BitBtn2Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;
var
 Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.FormCreate(Sender: TObject);
begin
  Edit1.Text:='0';                    // Начальное значение xn
  Edit2.Text:='2';                    // Конечное значение xk
  Edit3.Text:='0,25';                 // Значение шага h
  Edit4.Text:='0,001';                // Значение точности eps
  Memo1.Clear;
```

Библиотека БГУИР

```
  Label6.Caption:='Результаты ст. гр. 012501 Иванова И.И.';
end;
procedure TForm1.Button1Click(Sender: TObject);
var
 xn,xk,x,h,eps,a,s,y:Extended;
 n,k:Integer;
begin
 Memo1.Lines.Add('Вычисление таблицы значений функции');
 Memo1.Lines.Add('Исходные данные:');
 xn:=StrToFloat(Edit1.Text);
 Memo1.Lines.Add('xn = '+FloatToStrF(xn,ffFixed,6,2));
 xk:=StrToFloat(Edit2.Text);
 Memo1.Lines.Add('xk = '+FloatToStrF(xk,ffFixed,6,2));
 h:=StrToFloat(Edit3.Text);
 Memo1.Lines.Add('h = '+FloatToStrF(h,ffFixed,8,3));
 eps:=StrToFloat(Edit4.Text);
 Memo1.Lines.Add('eps = '+FloatToStrF(e,ffFixed,8,5));
 Memo1.Lines.Add('Таблица значений функции');
 Memo1.Lines.Add(' x      S(x)      Y(x)    n');
 x:=xn; // Начальное значение x  (левая граница интервала)
 Repeat
  a:=1;  // Начальное значение рекуррентной части
  S:=1; // Начальное значение суммы ряда, т. к. по условию задания k = 0
  n:=0; // Начальное значение числа итераций (приближений)
  While Abs(a)> eps Do //Вычисление суммы ряда с заданной точностью
   Begin
    Inc(n);     //  Вычисление числа итераций (приближений)  n:=n+1;
    a:=-a*x/n;//Вычисление рекуррентной части: знака, степени, факториала
    s:=s+a;
   End;
  y:=Exp(-x);       // Вычисление значения функции y(x)
  Memo1.Lines.Add(FloatToStrF(x,ffFixed,6,2)+' '+
                  FloatToStrF (s,ffFixed,8,4)+' '+
                  FloatToStrF(y,ffFixed,8,4)+ ' '+IntToStr(n));
  x:=x+h;      // Вычисление следующего значения x  (добавление  шага  h)
 Until x>xk+h/2  // xk+h/2 применяется для исключения потери последнего x
end;
procedure TForm1.BitBtn2Click(Sender: TObject);
begin              //  Очистка полей ввода и вывода для нового расчета
  Edit1.text:=' ';
  Edit2.text:=' ';
  Edit3.text:=' ';
```

Библиотека БГУИР

```
  Edit4.text:=' ';
  Memo1.Clear;
  Label6.Caption:=' ';
end;
end.
```

После отладки программы введите тест (n = 2, x_n = 0, x_k = 1, h = 3), установите курсор на оператор `Inc(n);` и нажмите клавишу F4. После этого, нажимая клавишу F7, пошагово выполните программу и проследите, как меняются все переменные в процессе выполнения.

4.5. Индивидуальные задания

В соответствии с индивидуальным заданием (табл. 4.2) составить блок-схему алгоритма и программу вывода на экран таблицы значений функции Y(x) и ее разложения в ряд S(x) для x, изменяющегося от x_n до x_k, с заданным количеством шагов m ($h = \frac{x_k - x_n}{m}$) и количеством слагаемых для вычисления S(x) или точностью eps. Близость значений S(x) и Y(x) во всем диапазоне значений x указывает на правильность вычисления S(x) и Y(x).

Спроектировать панель диалога. Создать вместо обработчика Button1.Click обработчик Memo1Click или Label1DblClick.

Изучить средства отладки программ.

Таблица 4.2

Индивидуальные задания

Номер варианта	x_n	x_k	S(x)	n или ε	Y(x)
1	2	3	4	5	6
1	0,1	1	$x-\frac{x^3}{3!}+\ldots+(-1)^n\frac{x^{2n+1}}{(2n+1)!}$	16	$\sin x$
2	0,1	1	$1+\frac{x^2}{2!}+\ldots+\frac{x^{2n}}{(2n)!}$	10	$\frac{e^x+e^{-x}}{2}$
3	0,1	1	$1+\frac{\cos\frac{\pi}{4}}{1!}x+\ldots+\frac{\cos n\frac{\pi}{4}}{n!}x^n$	12	$e^{x\cos\frac{\pi}{4}}\cos(x\sin\frac{\pi}{4})$
4	0,1	1	$1-\frac{x^2}{2!}+\ldots+(-1)^n\frac{x^{2n}}{(2n)!}$	8	$\cos x$
5	0,1	1	$1+3x^2+\ldots+\frac{2n+1}{n!}x^{2n}$	14	$(1+2x^2)e^{x^2}$
6	0,1	1	$x+\frac{x^3}{3!}+\ldots+\frac{x^{2n+1}}{(2n+1)!}$	8	$\frac{e^x-e^{-x}}{2}$
7	0,1	0,9	$\frac{x^3}{3}-\frac{x^5}{15}+\ldots+(-1)^{n+1}\frac{x^{2n+1}}{4n^2-1}$	13	$\frac{1+x^2}{2}\operatorname{arctg}x-\frac{x}{2}$

Библиотека БГУИР

Продолжение табл. 4.2

1	2	3	4	5	6
8	0,1	1	$1+\frac{2x}{1!}+\ldots+\frac{(2x)^n}{n!}$	12	e^{2x}
9	0,1	1	$1+2\frac{x}{2}+\ldots+\frac{n^2+1}{n!}\left(\frac{x}{2}\right)^n$	7	$\left(\frac{x^2}{4}+\frac{x}{2}+1\right)e^{\frac{x}{2}}$
10	0,1	0,7	$x-\frac{x^3}{3}+..+(-1)^n\frac{x^{2n+1}}{2n+1}$	9	arctgx
11	0,1	1	$1-\frac{3}{2}x^2+..+(-1)^n\frac{2n^2+1}{(2n)!}x^{2n}$	4	$\left(1-\frac{x^2}{2}\right)\cos x-\frac{x}{2}\sin x$
12	0,1	1	$-\frac{(2x)^2}{2}+\frac{(2x)^4}{24}-\ldots+(-1)^n\frac{(2x)^{2n}}{(2n)!}$	7	$2(\cos^2 x-1)$
13	–1,5	–0,2	$-(1+x)^2+\frac{(1+x)^4}{2}+\ldots+(-1)^n\frac{(1+x)^{2n}}{n}$	15	$\ln\frac{1}{2+2x+x^2}$
14	0,2	0,8	$\frac{x}{3!}+\frac{4x^2}{5!}+\ldots+\frac{n^2}{(2n+1)!}x^n$	3	$\frac{1}{4}\left(\frac{x+1}{\sqrt{x}}\mathrm{sh}\sqrt{x}-\mathrm{ch}\sqrt{x}\right)$
15	0,1	0,9	$\frac{x^2}{2}-\frac{x^4}{12}+\ldots+(-1)^{n+1}\frac{x^{2n}}{2n(2n-1)}$	15	$x\cdot\mathrm{arctg}x-\ln\sqrt{1+x^2}$
16	0,1	0,9	$\sum_{i=1}^{\infty}\frac{x^{2^{i-1}}}{1-x^{2^i}}$	0,001	$\frac{x}{1-x}$
17	0,1	3	$\sum_{i=1}^{\infty}(-1)^{i-1}\frac{\cos((2i-1)x)}{2i-1}$	0,001	$\pi/4$
18	0,2	1,2	$\sum_{i=1}^{\infty}\frac{(-1)^i 2^i\cos(ix)}{i}$	0,0001	$-\frac{1}{2}\ln(5+4\cos(x))$
19	1	3	$\sum_{i=0}^{\infty}\frac{(x\ln(2))^i}{i!}$	0,0001	2^x
20	0,1	1,5	$\sum_{i=1}^{\infty}(-1)^{i+1}\frac{2^{2i-1}x^{2i}}{(2i)!}$	0,0001	$\mathrm{Sin}^2(x)$
21	0,1	1,5	$\frac{1}{4}\sum_{i=0}^{\infty}(-1)^{i+1}\frac{3^{2i+1}-3}{(2i+1)!}x^{2i+1}$	0,0001	$\mathrm{Sin}^3(x)$
22	0,1	2	$\mathrm{cosec}(x)\sum_{i=1}^{\infty}(-1)^{i+1}\frac{2^{2i}x^{4i-2}}{(4i-2)!}$	0,001	Sh(x)
23	0,1	1	$\sec(x)+\sec(x)\sum_{i=1}^{\infty}(-1)^i\frac{2^{2i}x^{4i}}{(4i)!}$	0,001	Ch(x)
24	0,1	1,5	$x\prod_{i=1}^{\infty}\left(1-\frac{x^2}{i^2\pi^2}\right)$	0,001	Sin(x)

Библиотека БГУИР

Окончание табл. 4.2

1	2	3	4	5	6
25	0,1	1,5	$\prod_{i=1}^{\infty}\left(1-\frac{4x^2}{(2i+1)^2\pi^2}\right)$	0,001	Cos(x)
26	0,1	0,9	$x\prod_{i=1}^{\infty}\cos\left(\frac{x}{2^i}\right)$	0,001	Sin(x)
27	0,1	1,5	$\prod_{i=1}^{\infty}\left[1-\frac{4}{3}\sin^2\left(\frac{x}{3^i}\right)\right]$	0,001	$\frac{\sin(x)}{x}$
28	0,1	0,9	$\sum_{i=0}^{\infty}\frac{(2i)!}{2^{2i}(i!)^2(2i+1)}x^{2i+1}$	0,001	Arcsin(x)
29	0,1	1	$\sum_{i=0}^{\infty}\frac{2^{2i}(i!)^2x^{2i+2}}{(2i+1)!(i+1)}x^{2i+1}$	0,001	$\left(\arcsin(x)\right)^2$
30	0,1	2	$\ln(2)-\sum_{i=1}^{\infty}(-1)^i\frac{(2i-1)!}{2^{2i}(i!)^2}x^{2i}$	0,001	$\ln\left(1+\sqrt{1+x^2}\right)$

4.6. Задания повышенной сложности

1. Определить, является ли натуральное число k **палиндромом** (читается одинаково как слева направо, так и справа налево). *Например*: 1221, 35753 – палиндромы; 7664, 112 – не палиндромы.

2. Удалить из записи натурального числа n нечетные цифры, сохранив порядок следования остальных цифр. *Например* 5843 → 84.

3. Найти все двузначные числа, сумма цифр которых не меняется при умножении их на 2, 3, 4, 5, 6, 7, 8, 9.

4. Вывести трехзначные числа, квадраты которых оканчиваются тремя одинаковыми цифрами, отличными от нуля.

5. Поменять местами наибольшую и наименьшую цифры заданного натурального числа n.

6. Числа **Фибоначчи** (F_i) определяются по формулам: $F_0 = F_1 = 1$; $F_i = F_{i-1} + F_{i-2}$ при i = 2, 3, ... Найти первое из чисел Фибоначчи, превосходящих заданное число M (M > 0).

7. **Пифагоровыми** называются тройки натуральных чисел a, b, c, удовлетворяющие условию $a^2 + b^2 = c^2$. *Например,* пифагоровой является тройка чисел 6, 8, 10. Найдите все тройки пифагоровых чисел, не превышающих 25.

8. **Совершенным** называется число, равное сумме своих делителей, кроме себя самого. *Например,* совершенным является число 28 = 1 + 2 + 4 + 7 + 14. Найдите все совершенные числа на отрезке [a, b].

9. Число называется **автоморфным**, если после возведения в квадрат оно совпадает с младшими разрядами числа, *например* 52 = 25, 252 = 625. Определить, является ли заданное число n автоморфным.

Библиотека БГУИР

10. Число, состоящее из n цифр, называется числом **Армстронга**, если оно равно сумме n-х степеней своих цифр, *например* $153 = 1^3 + 5^3 + 3^3$. Определить, является ли заданное число p числом Армстронга.

11. Дана непустая последовательность различных натуральных чисел, за которой следует 0. Найти величину наибольшего среди отрицательных чисел этой последовательности и его порядковый номер.

12. Жители островов Чунга и Чанга один раз в год по праздникам обмениваются драгоценностями. Жители острова Чунга привозят половину своих драгоценностей на остров Чанга, а жители острова Чанга привозят треть своих драгоценностей на остров Чунга. Какая часть драгоценностей будет находиться на острове Чунга через M лет.

13. На заданном интервале (a, b) получить все трехзначные числа, в записи которых нет одинаковых цифр.

14. На заданном интервале (a, b) определить НОД чисел.

15. Найти k-ю цифру в последовательности, состоящей из квадратов всех натуральных чисел (149162536…).

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №5

ОБРАБОТКА ИСКЛЮЧИТЕЛЬНЫХ СИТУАЦИЙ. ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ОДНОМЕРНЫХ МАССИВОВ

Цель работы: изучить свойства компонента `TStringGrid`. Составить блок-схему, написать и отладить программу с использованием одномерных массивов, обработать возможные исключительные ситуации.

5.1. Обработка исключительных ситуаций

Под исключительной ситуацией понимается ошибочное состояние, возникающее при выполнении программы и требующее выполнения определенных действий для продолжения работы или корректного ее завершения.

Стандартный обработчик (метод TApplication.HandleException), вызываемый по умолчанию, информирует пользователя о возникновении ошибки и завершает выполнение программы. Для обработки исключительных ситуаций внутри программы используется оператор `Try`, который перехватывает исключительную ситуацию и дает возможность разработчику предусмотреть определенные действия при ее возникновении.

Конструкция блока `Try`… `Finally`:

```
Try
      операторы, выполнение которых может привести
      к возникновению исключительной ситуации
   Finally
      операторы, выполняемые всегда, вне зависимости
      от возникновения исключительной ситуации
End;
```

При возникновении исключительной ситуации в одном из операторов управление сразу передается первому оператору блока `Finally`.

Данная конструкция позволяет корректно завершить выполнение программы вне зависимости от возникающей исключительной ситуации. Обычно в блок `Finally` помещают операторы, закрывающие открытые файлы, освобождающие выделенную динамическую память. Недостатком такой конструкции является то, что программа не информирует о том, возникала ли исключительная ситуация, и, следовательно, не позволяет пользователю ее скорректировать.

Конструкция блока `Try`…`Except`:

```
Try
      операторы, выполнение которых может привести
      к возникновению исключительной ситуации
   Except
      операторы, выполняемые только в случае
      возникновения определенных исключительных ситуаций
End;
```

При возникновении исключительной ситуации управление передается в блок `Except`, иначе блок `Except` пропускается. Такая конструкция позволяет

Библиотека БГУИР

определить причину возникшей проблемы и рекомендовать пользователю определенные действия для ее исправления. В простейшем случае в разделе `Except` пишутся операторы, выполняемые при возникновении любой исключительной ситуации. Для определения типа возникшей ошибки в разделе `Except` используется конструкция, работающая по схеме оператора `Case`:

```
On тип исключительной ситуации 1 Do оператор 1;
On тип исключительной ситуации 2 Do оператор 2;
...
Else операторы, выполняемые, если не определен тип
        исключительной ситуации;
```

Выполняется только оператор, стоящий после `Do` для реализуемой исключительной ситуации. Некоторые из возможных типов исключительных ситуаций представлены в табл. 5.1.

Для отладки программы, содержащей обработку исключительных ситуаций, надо отключить опцию Stop on Delphi Exceptions, находящуюся в Tools – Debbuger Options… – закладка Language Exceptions.

Для преднамеренного инициирования исключительной ситуации применяются процедуры `Abort`, `Assert (`b:Boolean`)` и ключевое слово `Raise`:

```
Raise(тип исключения). Create(текст сообщения);
```

Таблица 5.1

Типы исключительных ситуаций

Тип исключительной ситуации	Причина
1	2
EAbort	Намеренное прерывание программы, генерируемое процедурой `Abort`
EArrayError	Ошибка при операциях с массивами: использование ошибочного индекса массива, добавление слишком большого числа элементов в массив фиксированной длины (для использования требует подключения модуля `MxArrays`)
EConvertError	Ошибка преобразования строки в другие типы данных
EDivByZero	Попытка целочисленного деления на нуль
ERangeError	Целочисленное значение или индекс вне допустимого диапазона (при включенной директиве проверки границ массива {$R+})
EIntOverflow	Переполнение при операции с целыми числами (при включенной директиве {$Q+ })
EInvalidArgument	Недопустимое значение параметра при обращении к математической функции
EZeroDivide	Деление на нуль числа с плавающей точкой
EOutOfMemory	Недостаточно памяти

Библиотека БГУИР

Окончание табл. 5.1

1	2
EFileNotFound	Файл не найден
EInvalidFileName	Неверное имя файла
EInvalidOp	Неправильная операция с плавающей точкой
EOverFlow	Переполнение при выполнении операции с плавающей точкой
EAssertionFailed	Возникает при намеренной генерации исключительной ситуации с помощью процедуры `Assert` (при включенной директиве {$C+})

5.2. Функции ShowMessage и MessageDlg

Для вывода сообщений применяются функции `ShowMessage` и `MessageDlg`. Функция `ShowMessage` (Msg:String) отображает диалоговое окно с заданным в Msg сообщением и кнопкой OK для закрытия окна. В заголовке окна отображается имя выполняемой программы.

Функция `MessageDlg` (Const Msg:WideString; DlgType:TMsgDlgType; Buttons: TMsgDlgButtons; HelpCtx: Longint): Word отображает диалоговое окно с заданными кнопками. Параметр Msg содержит текст сообщения. Параметр DlgType определяет вид отображаемого окна (табл. 5.2).

Например:

```
If MessageDlg ('Вы действительно хотите завершить работу?',
mtConfirmation,[mbOk,mbNo],0) = mrNo Then CanClose:= False;
```

Функция **MessageDlg** отображает диалоговое окно в центре экрана. Первый ее параметр-сообщение, выводимое в окне. Второй параметр – тип окна – может принимать следующие значения: mtWarning (предупреждение), mtError (ошибка), mtInformation (информация), mtConfirmation (подтверждение). Третий параметр задает кнопки в диалоговом окне, его значения: mbYes, mbNo, mbOK, **mbCancel**, mbAbort, mbRetry, mbIgnore, mbAll, mbHelp. Четвертый параметр определяет темы помощи, появляющиеся при щелчке на кнопке помощи или нажатии F1 во время отображения диалогового окна.

Таблица 5.2

Вид отображаемого окна, определяемого параметром DlgType

Значения типа окна	Вид отображаемого окна
1	2
mtWarning	Заголовок «Warning». Значок: желтый треугольник с восклицательным знаком внутри
mtError	Заголовок «Error». Значок: красный круг с перечеркиванием внутри
mtInformation	Заголовок «Information». Значок: символ “i” на голубом поле

Библиотека БГУИР

Окончание табл. 5.2

1	2
mtConfirmation	Заголовок «Confirmation». Значок: символ “?” на зеленом поле
mtCustom	Заголовок соответствует имени выполняемого файла. Без значка

Параметр Buttons указывает кнопки, которые будут находиться в окне (табл. 5.3). Список необходимых кнопок заключается в квадратные скобки.

Параметр HelpCtx определяет номер контекстной справки для данного окна.

Результатом выполнения функции является значение, соответствующее нажатой кнопке. Возвращаемое значение имеет имя, состоящее из букв mr и имени кнопки, *например*: mrYes, mrOK, mrHelp.

Таблица 5.3

Значения параметра Buttons

Значение параметра	Тип кнопки	Значение параметра	Тип кнопки
mbYes	Кнопка «Yes»	mbRetry	Кнопка «Retry»
mbNo	Кнопка «No»	mbIgnore	Кнопка «Ignore»
mbOK	Кнопка «OK»	mbAll	Кнопка «All»
mbCancel	Кнопка «Cancel»	mbHelp	Кнопка «Help»
mbAbort	Кнопка «Abort»		

5.3. Работа с массивами

Массив – пронумерованная структура данных, элементы которой обладают одним и тем же типом.

Например: 1 5 6 0 7 9 2 – *массив целых чисел*

a_1 a_2 a_3 a_4 a_5 a_6 a_7

1,2 5,1 7,9 2,1 5,5 9,3 4,8 – *массив вещественных чисел*

Каждый элемент массива обозначается именем, за которым в квадратных скобках указывается один или несколько индексов, разделенных запятыми, *например* a[1], bb[i], c12[i, j * 2], q[1, 1, i * j – 1]. В качестве индекса можно использовать любые порядковые типы.

5.3.1. Объявление массива

Тип массива или сам массив определяются соответственно в разделе типов (`Type`) или переменных (`Var`) с помощью следующей конструкции:

`Array` [описание индексов] `Of` тип элемента массива;

Например:

```
Const    N_Max = 100;
Type
  Mas = Array[1..N_Max] Of тип элементов массива;
Var
```

Библиотека БГУИР

```
  A: Mas;
  i, n: Byte; {i – порядковый номер (индекс) элемента в массиве A;
               n – количество элементов массива A}
```

Примеры описания массивов.

```
Const
 N_Max = 20;     // Задание максимального значения размера массива
Type
 Mas = Array[1..N] Of Byte;  // Описание типа одномерного массива
Var
 a: Mas;                          // a – массив типа Mas;
 b:Array[1..10] Of Integer;  // b – массив из десяти целых чисел
 y:Array[1..5,1..10] Of Char;//y – двумерный массив символьного типа
```

Элементы массивов могут использоваться в выражениях так же, как и обычные переменные, *например*:

```
F:=2*a[3]+a[b[4]+1]*3;
a[n]:=1+sqrt(abs(a[n-1]));
```

Для доступа к элементу массива необходимо указать его имя и индекс (порядковый номер элемента в массиве): имя_массива [индекс]

5.3.2. Примеры программ

Пример 1. Подсчитать сумму и количество отрицательных и положительных элементов массива.

```
Const    N_Max = 100;
Type
  Mas = Array[1.. N_Max] Of Extended;
Var
 a:Mas;
 i,n,K_Pol,K_Otr:Byte;//K_Pol – количество положительных элементов
 Sum_P,Sum_O:Extended;//K_Otr – количество отрицательных элементов
                      //Sum_P – сумма положительных элементов
                      //Sum_O – сумма отрицательных элементов
 Begin
  ... // Ввод элементов массива
  Sum_P:=0;                          // Инициализация переменных
  K_Pol:=0;
  Sum_O:=0;
  K_Otr:=0;
  For i:=1 To n Do Begin             // Подсчет сумм и количества
     If A[i]>0 Then Begin            // Если элемент положительный, то
                     Inc(K_Pol);             // подсчитать количество
                     Sum_P:=Sum_P+A[i]; // и сумму
                    End;
     If A[i]<0 Then Begin            // Если элемент отрицательный, то
                     Inc(K_Otr);             // подсчитать количество
```

Библиотека БГУИР

```
                    Sum_O:=Sum_O+A[i]; // и сумму
                  End;
    End;// For i
  ... // Вывод результатов
End;
```

Пример 2. Изменить значения элементов массива по заданному правилу:

– если значение элемента массива четное и положительное, то увеличить его в 2 раза,

– если кратно 3 – уменьшить на 10.

```
Const    N_Max = 100;
Type
  Mas = Array[1..N_Max] Of Integer ;
Var
  A:Mas;
  i,n:Byte;
Begin
... // Ввод элементов массива
  For i:=1 To n Do Begin       // Изменение значений элементов
    If (A[i]>0)And(Not Odd(A[i])) Then A[i]:=A[i]*2
            Else If A[i] Mod 3 = 0 Then A[i]:=A[i]-10;
  End;  // For i
... // Вывод результатов
End;
```

Пример 3. Записать положительные элементы массива A в массив B, а отрицательные элементы – в массив **C**, не меняя порядка следования элементов.

```
i_b:=0;                    // i_b – количество элементов массива B,
i_c:=0;                    // i_c – количество элементов массива C
For i:=1 to n Do Begin     // Формирование новых массивов
  If A[i]>0 Then Begin
                  Inc(i_b);          // Увеличение индекса
                  B[i_b]:=A[i];      // Запись в массив B
                End;
  If A[i]<0 Then Begin
                  Inc(i_c);          // Увеличение индекса
                  C[i_c]:=A[i];      // Запись в массив C
                End;
End;  // For i
```

Пример 4. Упорядочить элементы массива по возрастанию их значений, т. е. для всех элементов массива должно выполняться условие: $a_i < a_{i+1}$.

```
For k:=1 To n-1 Do                 // k – номер просмотра массива
  For i:=1 To n-k Do               // Просмотр элементов массива
    If A[i]>A[i+1] Then Begin      // Сравнение элементов массива
```

Библиотека БГУИР

```
                    Tmp:=A[i];      // Перестановка элементов
                    A[i]:=A[i+1]; // ai и ai+1, если они
                    A[i+1]:=Tmp;  // стоят неправильно
                  End;
```

Пример 5. Удалить из одномерного массива все отрицательные элементы.

Для решения данной задачи необходимо выполнить следующие действия:

– найти индекс удаляемого элемента;

– сдвинуть элементы, стоящие после него, в начало на один индекс;

– уменьшить размер массива на 1.

```
i:=1;
While i<=n Do Begin
 If B[i]<0 Then Begin   {1} // Если найден отрицательный элемент, то
    For j:=i DownTo n-1 Do {2} // сдвинуть элементы, стоящие после
        B[j]:=B[j+1];          // удаляемого, на одну позицию
    Dec(n);                {3} // Уменьшить размер массива
    Dec(i);                    // Возврат к предыдущему индексу
   End; // If
  Inc(i);
End;              // While
```

Пример 6. Циклический сдвиг элементов массива на k позиций **влево**:

```
k:=k Mod n;
For j:=1 To k Do Begin
                    Tmp:=A[1];
                    For i:=1 To n-1 Do
                       A[i]:=A[i+1];
                    A[n]:=Tmp;
                  End;
```

5.4. Компонент TStringGrid

Компонент TStringGrid предназначен для отображения информации в виде двумерной таблицы, каждая ячейка которой представляет собой окно однострочного редактора, как и строка ввода TEdit. Доступ к информации осуществляется с помощью свойства Cells [ACol:Integer; ARow:Integer] : String, где ACol, ARow – соответственно номер столбца и строки элемента двумерного массива. Нумерация осуществляется с **нуля**.

Свойства ColCount и RowCount устанавливают соответственно количество столбцов и строк в таблице, а свойства FixedCols и FixedRows задают количество столбцов и строк фиксированной зоны. Фиксированная зона выделена другим цветом, и в нее запрещен ввод информации с клавиатуры.

5.5. Порядок выполнения задания

В массиве A из n элементов поменять местами максимальный и минимальный элементы. Значение n вводить в компонент TEdit, A – в компонент TStringGrid. Результат после нажатия кнопки типа TButton вывести в компонент TStringGrid.

Панель диалога приведена на рис. 5.1.

Библиотека БГУИР

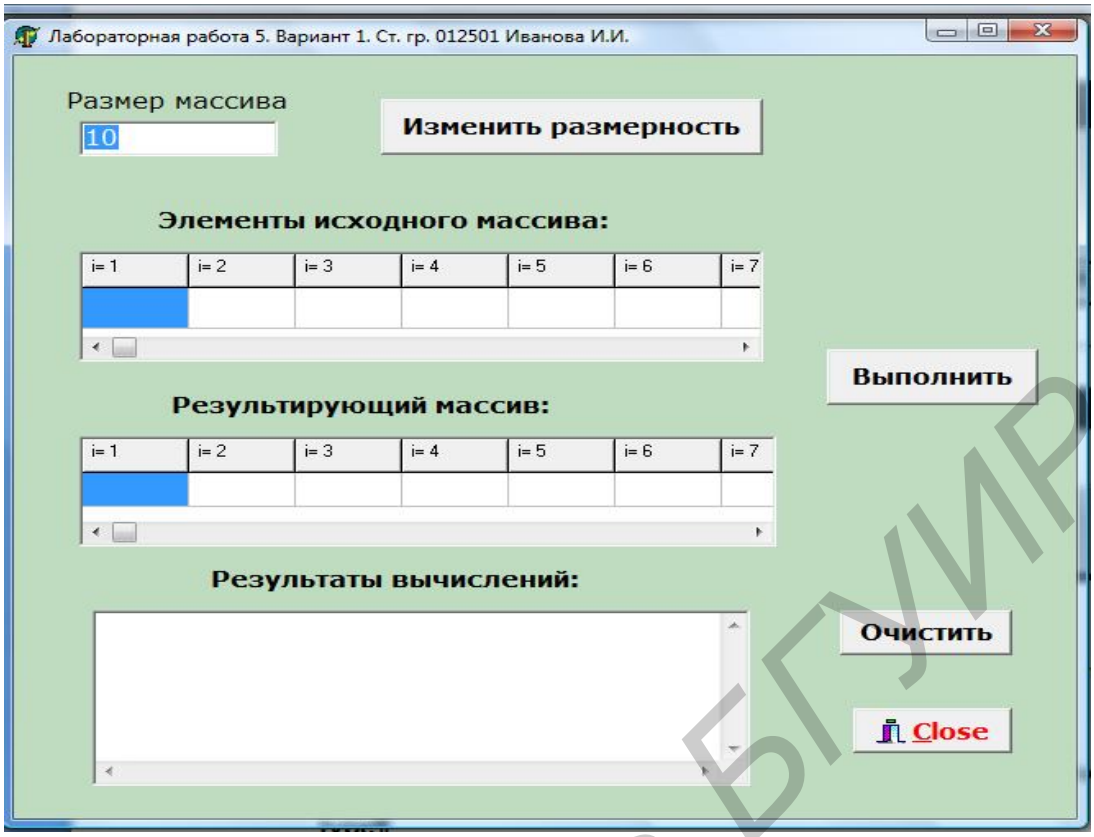


Рис. 5.1. Панель диалога

5.5.1. Настройка компонента TStringGrid

1. На вкладке Additional в меню компонент щелкните мышью по пиктограмме  для установки на форму компонентов StringGrid1 и StringGrid2.

2. Установите компонент в нужном месте формы и отрегулируйте его размер.

3. В инспекторе объектов установите значения свойств TStringGrid:

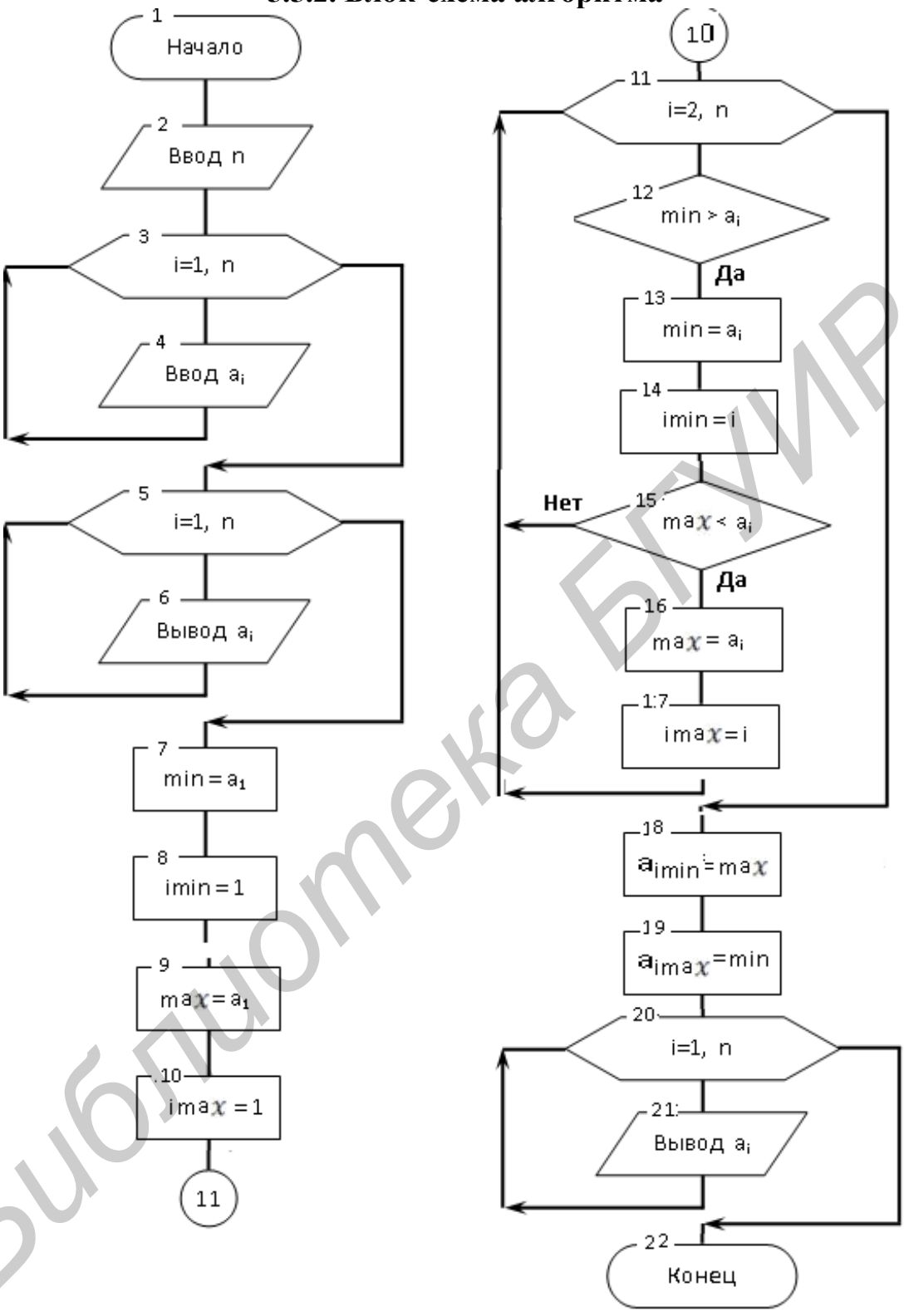
ColCount равным 7 (семь столбцов), RowCount равным 2 (две строки),

FixedCols – 0 (нет столбцов с фиксированной зоной), FixedRows – 1 (одна строка с фиксированной зоной для подписи индексов элементов массива).

4. В свойстве Options (ЛКМ на знаке «+», стоящем слева от Options) установите свойство goEditing в положение True для компонентов StringGrid1 и StringGrid2 (по умолчанию в компонент TStringGrid запрещен ввод информации с клавиатуры).

Библиотека БГУИР

5.5.2. Блок-схема алгоритма



Библиотека БГУИР

5.5.3. Код программы

```
unit Unit1;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms, Dialogs, StdCtrls, Buttons, Grids, MxArrays;
Type
 TForm1 = class(TForm)
  Label1: TLabel;
  Edit1: TEdit;
  Button1: TButton;
  StringGrid1: TStringGrid;
  Label2: TLabel;
  Button2: TButton;
  BitBtn1: TBitBtn;
  StringGrid2: TStringGrid;
  Label3: TLabel;
  Button3: TButton;
  Memo1: TMemo;
  Label4: TLabel;
  procedure FormCreate(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure Button3Click(Sender: TObject);
  procedure Button2Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;
Const
  Nmax = 100;      // Максимальный размер массива
Type               // Объявление типа одномерного массива размером Nmax
  Mas  =  Array[1..Nmax] Of Extended;
Var
  Form1: TForm1;
  A:Mas;                        // Объявление одномерного массива
  i, n: Byte;
Implementation
{$R *.dfm}
procedure TForm1.FormCreate(Sender: TObject);
Begin
  Edit1.Text:='10';             // Ввод размера массива
  n:=StrToInt (Edit1.Text);
```

Библиотека БГУИР

```
  Memo1.Clear;
  StringGrid1.ColCount:=n;      // Задание числа столбцов в таблицах
  StringGrid2.ColCount:=n;
  For i:=0 To n-1 Do Begin //Заполнение верхней строки поясняющими
   StringGrid1.Cells[i,0]:='i = '+IntToStr(i+1);  // подписями
   StringGrid2.Cells[i,0]:='i = '+IntToStr(i+1);
  End;
End;
procedure TForm1.Button1Click(Sender:TObject); // Изменить
Begin                                          // размер таблицы
  n:=StrToInt(Edit1.Text);      // Количество элементов в массиве
  StringGrid1.ColCount:=n;
  StringGrid2.ColCount:=n;
  For i:=0 To n-1 Do Begin //Заполнение верхней строки поясняющими
   StringGrid1.Cells[i,0]:='i = '+IntToStr(i+1); // подписями
   StringGrid2.Cells[i,0]:='i = '+IntToStr(i+1);
  End;
End;
procedure TForm1.Button3Click(Sender:TObject); // Очистить
begin
 Memo1.Clear;
 Edit1.Text:=' ';
 For i:=0 To n-1 Do Begin
  StringGrid1.Cells[i,1]:=' ';
  StringGrid2.Cells[i,1]:=' ';
 End;
End;
procedure TForm1.Button2Click(Sender: TObject);
Var
  iMax,iMin:Byte;
  Max,Min,t:Extended;
Begin
{$R+}
 Try     // Заполнение массива A элементами из таблицы StringGrid1
  Memo1.Lines.Add('Исходный массив:');
  For i:=0 To n-1 Do
   A[i+1]:=StrToFloat(StringGrid1.Cells[i,1]);
  For i:=1 To n Do
   Memo1.Lines.Add('A['+IntToStr(i)+']='+FloatToStrF(A[i],ffFixed,6,0));
 Except
 On ERangeError Do Begin
  ShowMessage('Выход за пределы массива. Уменьшите размер
  массива');
```

Библиотека БГУИР

```
 Exit;
End;
On EConvertError Do Begin
 ShowMessage('В ячейке отсутствует значение или число введено неправильно.');
 Exit;
End
Else Begin
  ShowMessage('Возникла неизвестная исключительная ситуация!');
  Exit;
 End;
End;
Try  //Поиск минимального и максимального элементов массива и их индексов
  Max:=A[1];                  // Инициализация значений
  iMax:=1;
  Min:=A[1];
  iMin:=1;
  For i:=2 To n Do Begin
    If Max<A[i] Then Begin // Поиск
      Max:=A[i];             // максимального элемента
      iMax:=i;               // и его индекса
    End;
    If Min>A[i] Then Begin // Поиск
      Min:=A[i];             // минимального элемента
      iMin:=i;               // и его индекса
    End;
  End;
  A[iMax]:=Min;
  A[iMin]:=Max;
  Except
    On EInvalidOp Do Begin
      MessageDlg('Неправильная операция с плавающей точкой.',
                 mtError, [mbCancel],0);
      Exit;
    End;
    On EOverFlow Do Begin
     MessageDlg('Переполнение при выполнении операции'+
                'с плавающей точкой.', mtError,[mbCancel],0);
      Exit;
    End
    Else Begin
     MessageDlg('Возникла неизвестная исключительная ситуация!',
                  mtError,[mbCancel],0);
      Exit;
    End;
```

Библиотека БГУИР

```
   End;
   For i:=0 To n-1 Do // Вывод результата в таблицу StringGrid2
  StringGrid2.Cells[i,1]:=FloatToStrF(A[i+1],ffFixed,6,0);
   Memo1.Lines.Add('Min = '+FloatToStrF(Min,ffFixed,6,0)+
                   ',iMin = '+IntToStr(iMin));
   Memo1.Lines.Add('Max = '+FloatToStrF(Max,ffFixed,6,0)+
                   ', iMax = '+IntToStr(iMax));
   Memo1.Lines.Add('Результирующий массив:');
   For i:=0 To n-1 Do
    Memo1.Lines.Add('A['+IntToStr(i)+'] = '+
                    FloatToStrF(A[i+1],ffFixed,6,0));
End;
End.
```

5.6. Индивидуальные задания

В соответствии с индивидуальным вариантом составить блок-схему алгоритма и программу. Во всех заданиях переменные вводить и выводить с помощью компонента TEdit, массивы – с помощью компонента TStringGrid. Вычисления выполнять после нажатия кнопки типа TButton. Вывести исходные данные и полученный результат. В местах возможного возникновения ошибок использовать конструкции для обработки исключительных ситуаций.

В одномерном массиве, состоящем из n вещественных элементов, определить:

1. Индексы совпадающих по величине соседних элементов.
2. Произведение между максимальным и минимальным элементами.
3. Сумму между первым и последним отрицательными элементами.
4. Среднее арифметическое элементов, расположенных до последнего положительного значения.
5. Сумму после последнего положительного элемента.
6. Произведение и количество кратных трем элементов, расположенных до первого отрицательного элемента.
7. Минимальный элемент, расположенный между первым и последним нулевыми элементами.
8. Произведение нечетных положительных элементов, расположенных после максимального по модулю элемента.
9. Среднее арифметическое положительных кратных пяти элементов массива, расположенных до минимального элемента.
10. Произведение четных отрицательных элементов, расположенных между первым и вторым положительными элементами.
11. Сумму и количество нечетных элементов, расположенных после последнего отрицательного элемента.
12. Среднее арифметическое модулей, кратных семи, элементов, расположенных после максимального элемента.

Библиотека БГУИР

13. Произведение модулей элементов, расположенных между первым и вторым минимальными элементами.

14. Сумму модулей элементов, расположенных между первым и последним нулевыми элементами.

15. Среднее арифметическое модулей четных элементов, расположенных между первым отрицательным и последним положительным элементами.

16. Второй по счету максимальный элемент и его индекс.

17. Максимальное значение между суммами четных и нечетных элементов массива, расположенных до минимального элемента.

18. Максимальное значение среди отрицательных элементов.

19. Минимальное значение среди четных элементов.

20. Произведение элементов, индексы которых совпадают с их значениями.

21. Индекс первого элемента, совпадающего по модулю с заданным числом t, или вывести сообщение, что такого нет.

22. Индекс последнего элемента, меньшего заданного числа t, или вывести сообщение, что такого нет.

23. Среднее арифметическое элементов, больших по модулю заданному числу t, или вывести сообщение, что таких нет.

24. Произведение элементов между первым и последним элементами, равными заданному числу t, или вывести сообщение, что таких нет.

25. Максимальное и минимальное значения элементов, меньших заданного числа t, или вывести сообщение, что таких нет.

26. Произведение неповторяющихся элементов.

27. Наиболее часто встречающийся элемент и его индекс.

28. Среднее арифметическое четных неповторяющихся элементов.

29. Значения и индексы двух элементов, произведение которых максимально.

30. Количество инверсий, т. е. таких пар элементов, в которых большее число находится слева от меньшего.

5.7. Задания повышенной сложности

1. Даны массивы A размером n и B размером m, в которых числа не повторяются. Записать в новый массив элементы, встречающиеся как в массиве A, так и в B.

2. Удалить из массива A размером n элементы, не входящие в массив B размером m.

3. Заданы два массива из n и m вещественных чисел. Найти наименьшее среди тех элементов первого массива, которые не входят во второй.

4. Преобразовать элементы массива следующим образом: все четные элементы перенести в начало, а остальные – в конец, сохраняя исходное взаимное расположение как среди четных, так и среди остальных элементов.

5. Массив из n действительных чисел оставить без изменения, если он упорядочен по неубыванию или по невозрастанию; в противном случае –

Библиотека БГУИР

получить положительные элементы исходного массива, упорядоченные по возрастанию.

6. В массиве B вставить после каждого отрицательного кратного трем элемента его модуль.

7. Удалить из массива X элементы, входящие в него более одного раза.

8. Определить в массиве наиболее часто встречающееся значение элемента и записать в новый массив те элементы, значения которых меньше найденного.

9. Заменить второй элемент массива $\max(x_1,x_2)$, третий – $\max(x_1, x_2, x_3)$ и т. д.

10. Упорядочить элементы целочисленного массива A по возрастанию количества цифр в числах, входящих в него (сначала однозначные, затем двузначные, трехзначные и т. д.). Числа с одинаковой разрядностью должны располагаться по убыванию.

11. Создать новый массив из наиболее длинной подпоследовательности подряд идущих убывающих значений.

12. Объединить элементы упорядоченных по неубыванию массивов, не применяя сортировку, в один массив так, чтобы они снова оказались упорядоченными по неубыванию.

13. Создать новый массив, каждый элемент которого рассчитывается как среднее арифметическое стоящих перед ним элементов.

14. Записать в новый массив те элементы, перед которыми находятся все меньшие их значения, а после них – большие.

15. Удалить из массива элементы, образующие возрастающую подпоследовательность.

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №6

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ДВУМЕРНЫХ ДИНАМИЧЕСКИХ МАССИВОВ

Цель работы: написать программу с использованием двумерных динамических массивов.

6.1. Динамические переменные и указатели

Динамические данные безымянны, поэтому к ним обращаются не по имени, а по адресу в памяти. Адреса хранятся в переменных особого типа – **указателях**. Под каждую переменную типа указатель отводится ячейка памяти объемом **4 байта**, в которую можно поместить адрес любой переменной.

Для динамических переменных характерно следующее:

1) к ним невозможно обратиться с помощью идентификатора, т. к. они не описываются явным образом;

2) память для них выделяется в специальной heap-области (динамической памяти) только во время выполнения программы;

3) доступ к ним происходит с помощью указателей (или ссылок), которые становятся активными после определения динамического объекта.

Указатель – специальная переменная, с помощью которой осуществляется ссылка на значение какой-то другой переменной, расположенной по определенному адресу оперативной памяти, начиная с которого может размещаться значение.

Указатели могут быть:

1) **типированными** – адреса присваиваиваются только указателям заданного типа;

2) **нетипированными –** совместимы с любым типом указателя, поэтому можно присваивать адреса любого типа; описываются с типом `Pointer`.

Для объявления типированного указателя используется специальный символ **^** (каре).

Например:

```
Type
  Pb=^Byte;
  Pi=^Integer;
  Ps=^String;
Var
  p1,p2:Pointer;  // Нетипированные указатели
  i:Pi;           // Типированные указатели: i – указатель на целый тип
  k:Pb;           // Указатель на тип Byte
  S1,S2:Ps;       // Указатели на строковый тип
  P:^Extended;    // Указатель на вещественный тип
  x:Extended;     // Переменная вещественного типа
```

В общем виде объявление указателей можно записать:

Библиотека БГУИР

```
Type
   Pk=^тип;          // тип – любой базовый тип
Var
   a,b:Pk;           // Объявление указателей через имя типа
   Ptr1:^тип;        // Непосредственное объявление указателя
   Ptr:Pointer;      // Нетипированный указатель
```

6.1.1. Операции над указателями

Над указателями можно применять **операции присваивания** и **сравнения**.

Указателю можно **присвоить**:

1. Адрес другого указателя, *например* `p1:= p2;`

Используемые в операторе присваивания типированные указатели должны быть одного типа: `S1:= S2;`

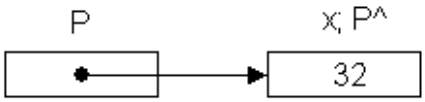
Применяя нетипированные указатели, можно передать адрес между указателями разного типа, *например:*

```
p1:=i;
k:=p1;
```

2. Константу `Nil` (пустой указатель), *например*:

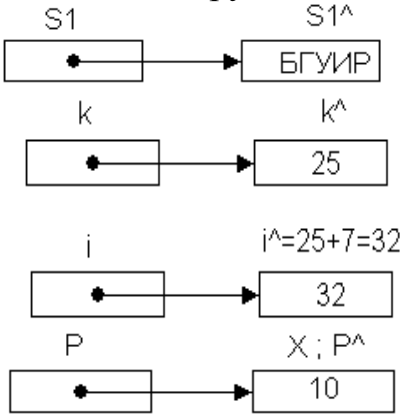
```
S1:=Nil; // Очистка адреса, т. е. указатель никуда не указывает
k:=Nil,
```

3. Адрес объекта, определенный с помощью функции `Addr()` или оператора взятия адреса @:




```
x:=32;
P:=Addr(x); // Инициализация указателя
P:=@x; // То же
```

Через указатель можно изменить значение переменной, на которую он указывает, *например*:




```
S1^:='БГУИР';
// По адресу S1 размещается строка 'БГУИР'

k^:=25;  // По адресу k размещается значение 25
i^:=k^+7; // По адресу i размещается результат k^+7

x:=10; // Переменной x присваивается значение 10
p^:=x; // По адресу p размещается значение 10
```

Символ ^, записанный после указателя, называется **оператором доступа по адресу**.

Библиотека БГУИР

Для присваивания значения (не адреса) нетипированному указателю надо его привести к соответствующему типу, *например* `Integer(p1^):=10;`

При такой записи переменная p1 по-прежнему является указателем, но теперь ей можно присваивать адреса только целых переменных.

Указатели одного типа можно **сравнивать** на равенство **=** или неравенство **<>**, *например*: `If a=b Then...// Указатели` a *и* b *ссылаются на одни и те же данные*

`If a<>b Then... // Указатели` a *и* b *ссылаются на разные данные*

Часто используется проверка условия: связан ли указатель с динамической переменной, *например* `If b<>Nil Then...`

`If p1=Nil Then New(p1); // Если указателю не выделена динамическая память,`

// то она выделяется процедурой `New();`

`If p1<>Nil Then Dispose(p1); // Если указатель указывает на какое-либо`

// значение, то происходит очистка его адреса процедурой Dispose();

Действия над указателями возможны лишь после того, как им будут присвоены конкретные значения (адреса).

6.2. Динамическое распределение памяти

Вся свободная от программ память компьютера представляет собой массив байт, называемый **кучей**. При необходимости использования программой дополнительной памяти осуществляется ее выделение одной из процедур `New();` или `GetMem();`

6.2.1. Процедура New()

Общий вид процедуры: `New` (типированный указатель);

Процедура `New();` выделяет в динамической области памяти свободный участок для размещения соответствующего указателю типа данных и присваивает этот адрес указателю.

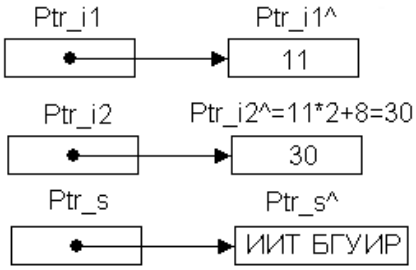
С помощью процедуры `Dispose` (типированный указатель); освобождается участок памяти, выделенный для размещения динамической переменной процедурой `New();` и значение указателя становится неопределенным.

Каждой процедуре `New();` должен соответствовать свой `Dispose();`

Например:

```
Type
  Ptr_Integer = ^Integer;
  Ptr_String = ^String;
Var
  Ptr_i1, Ptr_i2: Ptr_Integer;
  Ptr_s: Ptr_String;
Begin
```

Библиотека БГУИР




```
  New(Ptr_i1);
  New(Ptr_i2);
  New(Ptr_s);
  Ptr_i1^:=11;
  Ptr_i2^:=8;
  Ptr_s^:='БГУИР';
  If Ptr_i1^>Ptr_i2^ Then Ptr_i2^:=Ptr_i1^*2+Ptr_i2^;//Ptr_i2^ = 22 + 8 =
  =30
  Ptr_s^:='ИИТ '+Ptr_s^;          // В Ptr_s^ заносится 'ИИТ БГУИР'
  …
  Dispose(Ptr_s);        // Освобождение памяти, т. е. указатели не бу-
  дут
  Dispose(Ptr_i2);       // связаны с конкретными адресами памяти
  Dispose(Ptr_i1);
End.
```

6.2.2. Процедура GetMem()

При работе с любыми указателями можно выделить память процедурой

`GetMem` (указатель, размер выделяемой памяти);

которая выделяет блок динамической памяти указанного размера (в байтах) и присваивает адрес начала этой памяти указателю.

Освобождается выделенная память процедурой

`FreeMem` (указатель, размер освобождаемой памяти);

Размер освобождаемой памяти может определяться функцией `SizeOf (x)`, возвращающей количество байт, которое занимает переменная x.

Для работы с динамическими переменными надо:

1) создать динамическую переменную (`New();` или `GetMem();`);

2) выполнить с ней необходимые действия (присваивания или сравнения);

3) освободить выделенную для динамической переменной память в обратном порядке по сравнению с ее выделением (соответственно процедуры `Dispose();` или `FreeMem();`).

6.2.3. Массивы указателей

Элементами такого массива являются указатели на базовую структуру. Сам массив имеет фиксированный размер и занимает постоянное место в оперативной памяти. Динамические элементы базового типа создаются и размещаются в динамической памяти во время выполнения программы.

Пример организации двумерного динамического массива.

```
Type
  TMas=Array[1..1] Of тип элементов;
  PMas=^TMas;      //Указатель на массив
```

Библиотека БГУИР

```
  TMas1=Array[1..1] of PMas;
  PMas1=^TMas1;
Var
  a:PMas;
  b:PMas1;
  n,m,i,j:Integer;
Begin
  // Ввод n и m
  GetMem(a,sizeof(тип элемента)*n);
  GetMem(b,4*n);
  For i:=1 To n Do
    GetMem(b[i],m*SizeOf(Integer));
  … // Работа с массивами a[i] и b[i, j], 1 ≤ i ≤ n, 1 ≤ j ≤ m
 FreeMem(a,sizeof(тип элемента)*n);
  For i:=1 to n do
    FreeMem(b[i],m*SizeOf(тип элемента));
  FreeMem(b,4*m);
  …
End.
```

При работе с массивами указателей надо отключать проверку выхода индекса за пределы массива и следить за ним, чтобы не вышел за пределы выделенной памяти.

6.3. Примеры программ

Пример 1. Отсортировать элементы каждой строки массива A по возрастанию, применяя улучшенный метод пузырька.

```
Type
  Mas=array[1..1] of Extended;     //Тип одномерного массива
  PMas=^Mas;     //Тип указателя на одномерный массив вещественного типа
  Matr=array[1..1] of PMas;        //Тип массива указателей
  PMatr=^Matr;                     //Тип указателя на массив указателей
Var
  A:PMatr;
  i,n,j,m:Byte;
  t:Extended;    // Переменная для перестановки элементов массива
  Fl:Boolean;    // Флаг для определения: отсортирована строка или нет
Begin
   … // Ввод элементов массива
 For i:=1 To n Do
  Repeat
   Fl:=True;          // Предположение о том, что строка отсортирована
   For j:=1 To m - 1 Do    // Просмотр элементов i-й строки массива
```

Библиотека БГУИР

```
    if A^[i]^[j]>A^[i]^[j+1] Then Begin // Если элементы стоят по
            t:=A^[i]^[j];      // убыванию, то поменять их местами
            A^[i]^[j]:=A^[i]^[j+1];
            A^[i]^[j+1]:=t;
            Fl:=False;   // Строка не отсортирована
        End;
  Until Fl; // Если Fl=True, то выйти из цикла, т. к. строка отсортирована
   … // Вывод элементов массива
End;
```

Пример 2. Найти максимальный элемент, стоящий на главной диагонали, и минимальный элемент, стоящий на побочной диагонали массива A.

К указателю на элемент, стоящий на главной диагонали, обращаются как A^[i]^[i] или A^[j]^[j], т. к. индексы строки и столбца у него совпадают (A^[1]^[1], A^[2]^[2],…,A^[n]^[n]); а к элементу побочной диагонали – A^[i]^[n–i+1].

```
Max:=A^[1]^[1];      // Предположения о начальных значениях максимального
Min:=A^[1]^[n];      // и минимального элементов диагоналей
For i:=2 To n Do Begin // Поиск максимального и минимального элементов
    If A^[i]^[i]>Max Then Max:=A^[i]^[i];
    If A^[i]^[n-i+1]<Min Then Max:=A^[i]^[n-i+1];
  End;
```

Пример 3. Найти сумму нечетных элементов массива, расположенных ниже главной диагонали, и произведение четных элементов, расположенных выше ее.

```
Sum:=0;
For i:=2 To n Do    //Поиск нечетных элементов массива,
 For j:=1 To i-1 Do // расположенных ниже главной диагонали
  If Odd(A^[i]^[j]) Then Sum:=Sum+A^[i]^[j];
Pr:=1 ;
Kol:=0;
For i:=1 To n-1 Do   //Поиск четных элементов массива,
 For j:=i+1 To n Do // расположенных выше главной диагонали
  If Not Odd(A^[i]^[j]) Then Begin
    Pr:=Pr* A^[i]^[j];
    Inc(Kol); // Kol:=Kol+1;
  End;
If Kol=0 Then Pr:=0; //Если четных элементов нет, то произведение = =0.
```

6.4. Пример выполнения задания

Составить программу вычисления значения вектора $\vec{Y} = A \cdot \vec{B}$,

Библиотека БГУИР

где A – матрица размером n·m;

X, B – одномерные массивы размером n элементов.

Элементы вектора X определяются по формуле $X_i = \sum_{j=1}^{n} A_{ij} \cdot B_i$.

Значения n надо вводить в компонент TEdit, A и B – в компонент TStringGrid. После нажатия кнопки типа TButton результат вывести в компонент TStringGrid.

6.4.1. Настройка компонента TStringGrid

1.Установите на форму 3 компонента типа TStringGrid: StringGrid1, StringGrid2 и StringGrid3 (вкладка Additional – ЛКМ по пиктограмме 

.

2. Захватывая кромки компонентов, отрегулируйте их размер.

3. В инспекторе объектов установите значения свойств StringGrid1:

ColCount равным 4 (четыре столбца), RowCount равным 4 (две строки),

FixedCols – 1 (один столбец с фиксированной зоной для подписи номеров строк),

FixedRows – 1 (одна строка с фиксированной зоной для подписи индексов столбцов).

Установите значения свойств StringGrid2 и StringGrid3:

ColCount равным 1 (один столбец), RowCount равным 4 (четыре строки),

FixedCols – 0 (нет столбца с фиксированной зоной),

FixedRows – 1 (одна строка с фиксированной зоной для подписи имени массива).

Раскройте свойство Options (нажав на знак «+», стоящий слева от Options) и установите свойство goEditing в положение True для компонентов StringGrid1, StringGrid2 и StringGrid3 (по умолчанию в компонент TStringGrid запрещен ввод информации с клавиатуры).

Панель диалога приведена на рис. 6.1.

Рис. 6.1. Панель диалога формы

Библиотека БГУИР

6.4.2. Код программы

```
unit Unit1;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graph-
ics, Controls, Forms, Dialogs, Buttons, Grids, StdCtrls;
type
 TForm1 = class(TForm)
  Label1: TLabel;
  Edit1: TEdit;
  Button1: TButton;
  StringGrid1: TStringGrid;
  StringGrid2: TStringGrid;
  StringGrid3: TStringGrid;
  Label2: TLabel;
  Label3: TLabel;
  Button2: TButton;
  BitBtn1: TBitBtn;
  BitBtn2: TBitBtn;
  procedure FormCreate(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure BitBtn1Click(Sender: TObject);
private
 { Private declarations }
public
 { Public declarations }
end;
Type
 Mas=array[1..1] of Extended;   //Тип одномерного массива
 PMas=^Mas;    //Тип указателя на одномерный массив вещественного типа
 Matr=array[1..1] of PMas;      //Тип массива указателей
 PMatr=^Matr;                   //Тип указателя на массив указате-
лей
Var
 Pb,Px:PMas;                    //Указатели на массивы B и X
 Pa:PMatr;                      //Указатель на массив указателей A
 n,m,i,j:Integer;
 Form1:TForm1;
implementation
{$R *.dfm}
```

Библиотека БГУИР

```
procedure TForm1.FormCreate(Sender: TObject);
begin
 n:=3;              // Задание размера массивов: количества строк n и
 m:=3;              // количества столбцов m массива  A
 Edit1.Text:=IntToStr(n);
 Edit2.Text:=IntToStr(m);
 StringGrid1.ColCount:=m+1;     // Задание числа столбцов и
 StringGrid1.RowCount:=n+1;     // строк в таблицах, включая
 StringGrid2.RowCount:=m+1;    // фиксированную зону (+1) для подписей
 StringGrid3.RowCount:=n+1;
 StringGrid1.Cells[0,0]:=' A:';   //Ввод в левую верхнюю ячейку
 StringGrid2.Cells[0,0]:=' B:';   // таблицы названия массива
 StringGrid3.Cells[0,0]:=' X:';
 For i:=1 To n Do        // Заполнение таблицы поясняющими подписями:
   StringGrid1.Cells[0,i]:=' i='+IntToStr(i); //вывод номера строки
 For i:=1 To m Do
   StringGrid1.Cells[i,0]:=' j='+IntToStr(i); //вывод номера столбца
end;
procedure TForm1.Button1Click(Sender: TObject); // Изменить
begin                                           // размер
  n:=StrToInt(Edit1.Text);
  m:=StrToInt(Edit2.Text);
  StringGrid1.ColCount:=m+1; // Задание числа столбцов и
  StringGrid1.RowCount:=n+1; // строк в таблицах, включая
  StringGrid2.RowCount:=m+1; // фиксированную зону (+1) для подписей
  StringGrid3.RowCount:=n+1;
  For i:=1 To n Do           // Ввод поясняющих подписей:
    StringGrid1.Cells[0,i]:=' i='+IntToStr(i);   // номера строки
  For i:=1 To m Do
    StringGrid1.Cells[i,0]:=' j='+IntToStr(i); // номера столбца
end;
procedure TForm1.BitBtn1Click(Sender: TObject);   // Очистить
begin
  For i:=1 To n Do Begin
    For j:=1 To m Do Begin
     StringGrid1.Cells[j,i]:='';
     StringGrid2.Cells[0,j]:='';
    End;
    StringGrid3.Cells[0,i]:='';
```

Библиотека БГУИР

```
  End;
end;
procedure TForm1.Button2Click(Sender: TObject); // Вычислить
Var
  s: Extended;
begin
 GetMem(Pb,m*SizeOf(Extended)); // Выделение памяти для массива B
 GetMem(Px,n*SizeOf(Extended)); // Выделение памяти для массива X
 GetMem(Pa,n*SizeOf(Pointer)); //Выделение памяти для указателей на строки
 For i:=1 To n Do
  GetMem(Pa^[i],m*SizeOf(Extended));      //и значения массива A
 For i:=1 To n Do       // Заполнение массива A значениями из StringGrid1
  For j:=1 To m Do
   Pa^[i]^[j]:=StrToFloat(StringGrid1.Cells[j,i]);
 For i:=1 To m Do       //Заполнение массива B значениями из StringGrid2
  Pb^[i]:=StrToFloat(StringGrid2.Cells[0,i]);
 For i:=1 To n Do                  // Умножение матрицы A на вектор B
  Begin
   Px^[i]:=0;
 For j:=1 To m Do                  // Строка матрицы A умножается
    Px^[i]:=Px^[i]+Pa^[i]^[j]*Pb^[j];       // на столбец вектора B
  End;
 For i:=1 To n Do
  Begin
   StringGrid3.Cells[0,i]:=FloatToStr(Px^[i]);//Вывод результата
   FreeMem(Pa^[i],m*SizeOf(Extended));      //Освобождение памяти
  End;                          // в обратном порядке для строк матрицы A
 FreeMem(Pa,n*SizeOf(Pointer)); //Освобождение массива указателей
 FreeMem(Px,n*SizeOf(Extended)); //Освобождение памяти для массива X
 FreeMem(Pb,m*SizeOf(Extended));//Освобождение памяти для массива B
End;
End.
```

6.5. Индивидуальные задания

Используя динамические массивы и указатели, составить программы в соответствии с индивидуальным вариантом для лабораторных работ 5 и 6. Скалярные переменные вводить с помощью компонента TEdit с соответствующим пояснением в виде компонента TLabel. Скалярный результат выводить в виде компонента TLabel. Массивы представлять на форме в виде компонентов TStringGrid, в которых 0-й столбец и 0-ю строку использовать для отображения индексов массивов. Вычисления выполнять

Библиотека БГУИР

после нажатия кнопки типа TBitBtn. Для выхода из программы использовать кнопку Close. Предусмотреть применение исключительных ситуаций.

1. Из матрицы размером n·m получить массив B, присвоив его k-му элементу значение 0, если все элементы k-го столбца матрицы нулевые, и значение 1 в противном случае.

2. Из матрицы размером n·m получить массив B, присвоив его k-му элементу значение 1, если элементы k-й строки матрицы упорядочены по убыванию, и значение 0 в противном случае.

3. В матрице размером n·m получить массив B, присвоив его k-му элементу значение 1, если k-я строка матрицы симметрична, и значение 0 в противном случае.

4. В матрице размером n·m определить k – количество «особых» элементов матрицы, считая элемент «особым», если он больше суммы остальных элементов своего столбца.

5. В матрице размером n·m определить k – количество «особых» элементов матрицы, считая элемент «особым», если в его строке слева от него находятся элементы, меньшие его, а справа – большие.

6. В символьной матрице размером n·m определить k – количество различных элементов матрицы (повторяющиеся элементы считать один раз).

7. В матрице размером n·m упорядочить строки по неубыванию их первых элементов.

8. В матрице размером n·m упорядочить столбцы по неубыванию суммы их элементов.

9. Найти в каждом столбце матрицы максимальный элемент и записать их в новый массив. Среди найденных элементов определить минимальный.

10. Определить, является ли заданная квадратная матрица n-го порядка симметричной относительно побочной диагонали.

11. Определить количество отрицательных элементов, расположенных выше главной диагонали матрицы.

12. Определить количество положительных элементов, расположенных ниже побочной диагонали матрицы.

13. В матрице n-го порядка найти максимальный среди элементов, лежащих ниже побочной диагонали, и минимальный среди элементов, лежащих выше главной диагонали.

14. В матрице размером n·m поменять местами строку, содержащую элемент с наибольшим значением, со строкой, содержащей элемент с наименьшим значением.

15. Определить сумму элементов, расположенных на главной диагонали матрицы, и произведение элементов, расположенных на побочной диагонали.

16. Для матрицы размером n·m вывести на экран все ее седловые точки. Элемент матрицы называется седловой точкой, если он является наименьшим в своей строке и одновременно наибольшим в своем столбце или наоборот.

Библиотека БГУИР

17. Из матрицы n-го порядка получить матрицу порядка n–1 путем удаления из исходной матрицы строки и столбца, на пересечении которых расположен элемент с наибольшим по модулю значением.

18. В матрице n-го порядка переставить строки так, чтобы на главной диагонали матрицы были расположены элементы, наибольшие по абсолютной величине.

19. В двумерном массиве поменять местами столбцы, содержащие соответственно максимальный и минимальный элементы массива.

20. Пребразовать двумерную квадратную матрицу в одномерный массив путем обхода всех элементов двумерной матрицы по скручивающейся против часовой стрелки спирали.

21. Определить количество столбцов матрицы, состоящих только из одних нулей.

22. Определить произведение элементов тех столбцов матрицы, последний элемент которых равен нулю.

23. Из двумерного квадратного массива построить одномерный путем обхода его элементов по скручивающейся по часовой стрелке спирали.

24. Пребразовать двумерную квадратную матрицу в одномерный массив путем обхода всех элементов матрицы по скручивающейся по часовой стрелке спирали.

25. Для каждого столбца матрицы определить количество элементов, совпадающих с первым элементом столбца.

26. В матрице размером n·m найти в каждой строке наибольший элемент и поменять его местами с элементами главной диагонали.

27. В матрице размером n·m подсчитать количество столбцов матрицы, сумма элементов которых кратна трем, но не кратна пяти.

28. В каждом столбце матрицы определить количество элементов, больших среднего арифметического значения соответствующего столбца.

29. Поменять местами элементы четных и нечетных столбцов матрицы.

30. В каждой строке матрицы подсчитать количество отрицательных элементов кратных пяти и записать найденные элементы в новый одномерный массив.

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №7

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ПОДПРОГРАММ И БИБЛИОТЕК

Цель работы: изучить возможности Delphi для написания подпрограмм и создания библиотечных модулей. Написать и отладить программу**,** использующую внешний модуль Unit с подпрограммой.

7.1. Использование подпрограмм

Подпрограмма – группа операторов, имеющая уникальное имя и оформленная определенным образом, которая может быть вызвана любое количество раз. Подпрограммы подразделяются на процедуры и функции.

Процедура имеет следующую структуру:

```
Procedure имя_процедуры (список формальных параметров);
Const   описание используемых констант;
Type    описание используемых типов;
Var     описание используемых переменных;
Begin
   Операторы;
End;
```

Вызов процедуры: имя_процедуры (список фактических параметров);

список фактических параметров – имена фактических параметров, указанные при вызове подпрограммы и перечисляемые через запятую;

список формальных параметров – имена формальных параметров с указанием их типов, перечисляемые через точку с запятой.

Список формальных и фактических параметров, а также раздел описания (`Const`,`Type`,`Var`) не обязателен и может отсутствовать.

В случае если результатом подпрограммы является значение одной переменной, то рекомендуется ее оформлять в виде **функции.** Ее структура:

```
Function имя_функции (список формальных параметров):тип результата;
Const     описание используемых констант;
Type      описание используемых типов;
Var       описание используемых переменных;
Begin
  Операторы;
  Result:= …;   // Занесение результата вычислений в Result
End;
```

В отличие от процедуры вызов функции может осуществляться разными способами, *например* переменная:=имя_функции (список фактических параметров);

Для передачи результата из функции в точку ее вызова используется имя функции или специальная переменная `Result`.

Библиотека БГУИР

В качестве формальных параметров могут быть:

1) **параметры-значения** – передаются через стек и в вызывающую программу не возвращаются; в их качестве не могут быть файловые переменные и структуры, их содержащие; перед ними не ставятся ключевые слова;

2) **параметры-переменные** – через стек в подпрограмму передается адрес фактического параметра, и поэтому изменение его в подпрограмме приводит к его изменению и для вызывающей программы. Перед ними записывается ключевое слово `Var`;

3) **параметры-константы** – похожи на внутренние константы или параметры, доступные только для чтения. Они подобны параметрам-переменным, но внутри подпрограммы им нельзя присваивать никаких значений и нельзя передавать в другие подпрограммы, как параметры-переменные. Перед такими параметрами записывается ключевое слово `Const`.

Правила при использовании подпрограмм:

– фактическими параметрами при вызове подпрограмм могут быть переменные, константы и целые выражения;

– формальными параметрами при описании подпрограмм могут быть только имена переменных;

– фактические и формальные параметры должны быть согласованы по типу, порядку следования и количеству.

Имена процедур и функций могут быть использованы в качестве формальных параметров подпрограмм. Для этого определяется следующий тип:

`Type` имя_типа=`Function`(список формальных параметров):тип результата;

или `Type` имя_типа =`Procedure`(список формальных параметров);

7.2. Использование модулей Unit

Модуль `Unit` является отдельной программной единицей, т. к. описывается в отдельном текстовом файле с расширением *.pas и транслируется отдельно. Результатом трансляции является машинный код, записываемый в файл с расширением *.dcu. Структура модуля `Unit` может иметь вид, представленный в п. 2.2.

Все объявления, сделанные в интерфейсной секции, являются **глобальными** для программ, использующих данный модуль. Модули подключаются к другим модулям в разделе `Uses` в списке подключаемых модулей.

7.3. Создание модуля

Модули могут создаваться с формой или без нее. Для их создания надо выполнить меню File – New – Unit (или Form – модуль с формой). При изменении имени модуля, предложенного средой (*например* `Unit1`), следует изменить его название на новое (*например* `unit Unit1`; заменить на `unit MathF;`) и сохранить модуль в файле с таким же именем: меню File – Save As… (**имя файла должно совпадать с именем модуля**).

Библиотека БГУИР

7.4. Подключение модуля

Для подключения библиотечного модуля к проекту следует выбрать меню Project – опцию Add to Project… – файл, содержащий модуль. После этого в раздел `Uses` проекта, к которому подключается модуль, добавить имя подключаемого модуля (*например* `MathF`). **После подключения модуля в проекте становятся доступными все конструкции, описанные в интерфейсной части модуля.**

7.5. Пример выполнения задания

Составить программу вывода на экран таблицы функции для x, изменяющегося от x_n до x_k, с заданным количеством шагов m ($h = \frac{x_k - x_n}{m}$) и числом слагаемых n, которую оформить в виде процедуры. В качестве функции использовать по выбору $y = e^{-x}$ или $s = \sum_{k=0}^{\infty} (-1)^k \frac{x^k}{k!}$.

Панель диалога приведена на рис. 7.1.

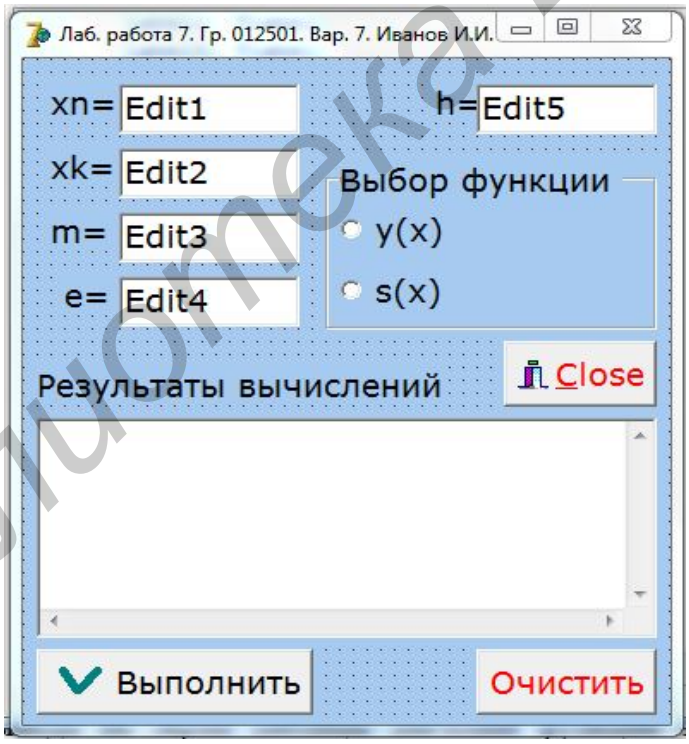


Рис. 7.1. Панель диалога

7.5.1. Порядок выполнения задания

1. Создать новый проект и сохранить его в новой папке.

Библиотека БГУИР

2. Установить на форму пять компонентов TEdit для ввода исходных данных (начала и конца отрезка [x_n, x_k], количества шагов m и слагаемых n) и вывода подсчитанного по формуле значения шага $h = \frac{x_k - x_n}{m}$, пять компонентов TLabel для подписей, TRadioGroup для выбора функции (y(x) или s(x)), TMemo для вывода результатов, кнопки TBitBtn (Выполнить, Очистить. Закрыть).

3. Создать процедуру FormCreate, в которой выполнить необходимые действия (задание начальных значений, выбор функции, очистка окна вывода).

4. Создать процедуру BitBtn1Click, в которой ввести значения исходных данных, вычислить и вывести в Edit5 шаг h, выбрать функцию для вычисления (y(x) или s(x)) и вызвать процедуру вывода результатов.

5. Создать процедуру очистки компонентов от содержащихся в них значений.

6. Перед всеми процедурами (см. пп. 3–5) написать функции вычисления значений y(x) и s(x). В функции вычисления s(x) предусмотреть обработку исключительных ситуаций.

7. Создать библиотечный модуль MathF для вывода результатов:

а) выбрать в меню File – New – Unit;

б) сохранить модуль с новым именем MathF (имя файла должно совпадать с именем модуля);

в) набрать код модуля;

г) подключить библиотечный модуль к проекту (меню Project – Add to Project… – выбрать файл, содержащий модуль – в разделе `Uses` основного модуля добавить имя подключаемого модуля). После подключения модуля в проекте становятся доступными все конструкции, описанные в интерфейсной части модуля;

д) выполнить написанную программу.

7.5.2. Код библиотечного модуля

```
Unit MathF;
Interface
 Uses StdCtrls, SysUtils;
 Type
  Fun=Function(x:Extended):Extended; // Объявление типа функция
      // Расчет таблицы функции f (табуляция функции)
  Procedure Tabl(f:Fun;xn,xk:Extended;m:Byte;Mem1:TMemo;Ed:TEdit);
Implementation
 Procedure Tabl; // Вывод значений выбранной функции
  Var
   x,y,h:Extended;
       i:Integer;
  Begin
   x:=xn;
```

Библиотека БГУИР

```
    h:=(xk-xn)/m;
    Ed.Text:=FloatToStrF(h,ffFixed,6,3));
    For i:=1 To m+1 Do
     Begin
      y:=f(x);
      Mem1.Lines.Add('x='+FloatToStrf(x,fffixed,8,3)+' y='+
                      FloatToStrf(y,fffixed,8,3));
      x:=x+h;
     End;
  End;
End.  // Конец модуля MathF
```

7.5.3. Код программы

```
Unit Unit1;
interface
uses
 Windows,Messages,SysUtils,Variants,Classes,Graphics,
 Controls, Forms,Dialogs,StdCtrls,Buttons,ExtCtrls,MathF;
type
 TForm1 = class(TForm)
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  Edit1: TEdit;
  Edit2: TEdit;
  Edit3: TEdit;
  Edit4: TEdit;
  RadioGroup1: TRadioGroup;
  Memo1: TMemo;
  BitBtn1: TBitBtn;
  BitBtn2: TBitBtn;
  procedure FormCreate(Sender: TObject);
  procedure BitBtn1Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;
Var
 Form1:TForm1;
 xn,xk:Extended;
```

Библиотека БГУИР

```
 m,n:Byte;
Implementation
{$R *.dfm}
Function yx(x:Extended):Extended;  // Вычисление y(x)
Begin
 Result:=exp(-x);
End;
Function Sx(x:Extended):Extended;
 Var
  a,s:Extended;      // a-рекуррентная часть суммы, s = Σ_{k=0}^{∞} (-1)^k x^k / k!
Begin
  a:=1;              // Начальные значения
  s:=1;
  Try                // Обработка исключительных ситуаций
   For i:=1 To n Do
    Begin
     a:=-a*x/n;      // Вычисление рекуррентной части
     s:=s+a;         // Вычисление суммы
    End;
   Result:=s;        // Передача результата
  Except
   On EInvalidOp do k:=MessageDlg('Неправильная операция с плавающей
   точкой. Продолжить вычисления?',mtError, [mbYes,mbNo],0);
   On EOverFlow do k:=MessageDlg('Переполнение при выполнении
   операции плавающей точкой! Продолжить вычисления?',mtError,
   [mbYes,mbNo],0);
   Else k:=MessageDlg('Возникла неизвестная исключительная ситуация!
   Продолжить вычисления?',mtError,[mbYes,mbNo], 0);
  End;
  Case k of
   mrYes: Result:=0;
   mrNo : Halt(1);
  End;
End;
Procedure TForm1.FormCreate (Sender: TObject);
Begin
 Memo1.Clear;                  // Очистка окна вывода
```

Библиотека БГУИР

```
 RadioGroup1.ItemIndex:=0;  // Выбор функции y(x)
 Edit1.Text:='0';           // Задание начальных значений
 Edit2.Text:='2';
 Edit3.Text:='10';
 Edit4.Text:='15';
 Edit5.Text:='';
End;
Procedure TForm1.BitBtn1Click(Sender: TObject);
Begin
 Memo1.Clear;
 Memo1.Lines.Add('Лаб. работа 7. Гр. 012501 Вар.7. Иванов И.И.');
 xn:=StrToFloat(Edit1.Text);          // Ввод начального значения xn
 Memo1.Lines.Add('xn='+FloatToStrF(xn,ffFixed,6,2));
 xk:=StrToFloat(Edit2.Text);          // Ввод конечного значения xk
 Memo1.Lines.Add('xk='+FloatToStrF(xk,ffFixed,6,2));
 m:=StrToInt(Edit3.Text);             // Ввод количества шагов m
 Memo1.Lines.Add('m ='+IntToStr(m));
 n:=StrToInt(Edit4.Text); // Ввод числа слагаемых n для расчета суммы
 Memo1.Lines.Add('n='+IntToStr(n));
 Case RadioGroup1.ItemIndex Of        // Выбор функции
   0: Begin
        Memo1.Lines.Add('Расчет Y(x)'); // Y(x)
        Tabl(yx,xn,xk,m,Memo1,Edit5);
      End;
   1: Begin
        Memo1.Lines.Add('Расчет S(x)'); // S(x)
        Tabl(sx,xn,xk,m,Memo1,Edit5);
      End;
 End;
End; End.
```

7.6. Индивидуальные задания

Составить программу вывода на экран таблицы значений функций Y(x) и S(x). Таблицу данных получить, изменяя параметр x от x_n до x_k с шагом h = (xk – xn) / m, где m – количество интервалов, на которые разбивается отрезок [x_n, x_k]. Ввод исходных данных организовать через компонент TEdit. Вычисление значений функций оформить в библиотечном модуле в виде функций пользователя. Самостоятельно выбрать удобные параметры настройки. В местах возможного возникновения ошибок использовать конструкции для обработки исключительных

Библиотека БГУИР

ситуаций. Выражения для функций Y(x) и S(x) следует взять из лабораторной работы №4.

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №8

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ СТРОК

Цель работы: изучить правила работы с компонентами TListBox и TComboBox. Составить и отладить программу работы со строками.

8.1. Типы данных для работы со строками

8.1.1. Короткие строки типа ShortString и String[N]

Короткие строки имеют фиксированное количество символов. Строка ShortString может содержать 255 символов. Строка String[N] может содержать N символов, но не более 255. Первый байт этих переменных содержит длину строки.

8.1.2. Длинная строка типа AnsiString

Память выделяется динамически (по мере необходимости) и может занимать до 2 Гбайт памяти. Для каждого символа отводится 1 байт. Процедуры и функции для работы с короткими и длинными строками представлены в прил. 4.

8.1.3. Широкая строка типа WideString

Аналогичен типу AnsiString, но для каждого символа выделяется по 2 байта памяти, что обеспеченивает совместимость с кодировкой Unicode.

8.1.4. Нуль-терминальная строка типа PChar

Представляет собой символы, ограниченные #0. Нуль-терминальные строки, широко использующиеся при обращениях к API-функциям Windows (API – Application Program Interface – интерфейс прикладных программ).

8.1.5. Представление строки в виде массива символов

Строка может быть описана как массив символов. Если массив имеет нулевую границу, он совместим с типом PChar.

```
Var
  MasC:Array[1..100] Of Char;
```

В отличие от нуль-терминальной строки длина имеет фиксированное значение и не может изменяться в процессе выполнения программы.

8.2. Компонент TListBox

Представляет собой список, элементы которого выбираются при помощи клавиатуры или мыши. Список элементов задается свойством Items, методы Add, Delete и Insert которого используются для добавления, удаления и вставки строк. Объект Items (TString) хранит строки, находящиеся в списке. Для определения номера выделенного элемента используется свойство ItemIndex.

Библиотека БГУИР

8.3. Компонент TComboBox

Комбинированный список TComboBox представляет собой комбинацию списка TListBox и редактора TEdit. Поэтому практически все свойства заимствованы у этих компонентов. Для работы с окном редактирования используется свойство Text, как в TEdit, а для работы со списком выбора – свойство Items, как в TListBox. Существует пять модификаций компонента, определяемых свойством Style. В модификации csSimple список всегда раскрыт, в остальных он раскрывается после нажатия кнопки справа от редактора.

8.4. Обработка событий

Обо всех происходящих в системе событиях, таких как создание формы, нажатие кнопки мыши или клавиатуры и т. д., ядро Windows информирует окна путем посылки соответствующих сообщений. Среда Delphi позволяет принимать и обрабатывать большинство таких сообщений. Каждый компонент содержит обработчики сообщений на странице Events инспектора объектов.

Для создания обработчика события надо раскрыть список компонентов в верхней части окна инспектора объектов и выбрать необходимый компонент. Затем на странице Events нажатием ЛКМ выбрать обработчик и дважды щелкнуть по его правой (пустой) части. В ответ Delphi активизирует окно редактора программного кода и покажет заготовку процедуры обработки выбранного события.

Каждый компонент имеет свой набор обработчиков событий, однако некоторые из них присущи большинству компонентов. Наиболее часто применяемые события представлены в табл. 8.1.

Таблица 8.1

Наиболее часто применяемые события

Событие	Описание события
1	2
OnActivate	Событие наступает при активации компонента
OnCreate	Возникает при создании компонента (выделения динамической памяти). В обработчике данного события следует задавать действия, которые должны происходить в момент создания компонента, например установки начальных значений
OnEnter	Компонент получает фокус, т. е. становится активным
OnExit	Компонент теряет фокус, т. е. теряет активность
OnKeyPress	Возникает при нажатии одной клавиши на клавиатуре. Параметр Key имеет тип Char и содержит ASCII – код нажатой клавиши (клавиша Enter клавиатуры имеет код #13, клавиша Esc – #27 и т. д.). Обычно это событие используется в том случае, когда необходима реакция на нажатие одной из клавиш

Библиотека БГУИР

Окончание табл. 8.1

1	2
OnKeyDown	Возникает при нажатии одной или нескольких клавиш на клавиатуре. Обработчик этого события получает информацию о нажатой клавише и состоянии клавиш Shift, Alt и Ctlr, а также о нажатой кнопке мыши. Информация о клавише передается параметром Key, который имеет тип Word
OnKeyUp	Является парным событием для OnKeyPress и возникает при отпускании ранее нажатой клавиши
OnClick	Возникает при нажатии левой клавиши мыши в области компонента
OnDblClick	Возникает при двойном нажатии левой клавиши мыши в области компонента

8.5. Пример выполнения задания

Сотавить программу подсчета числа слов в произвольной строке, слова которой разделены пробелами. Для ввода строк и работы с ними используется TComboBox. Ввод строки заканчивается нажатием Enter. Для выхода из программы используется кнопка Close.

Панель диалога приведена на рис. 8.1.

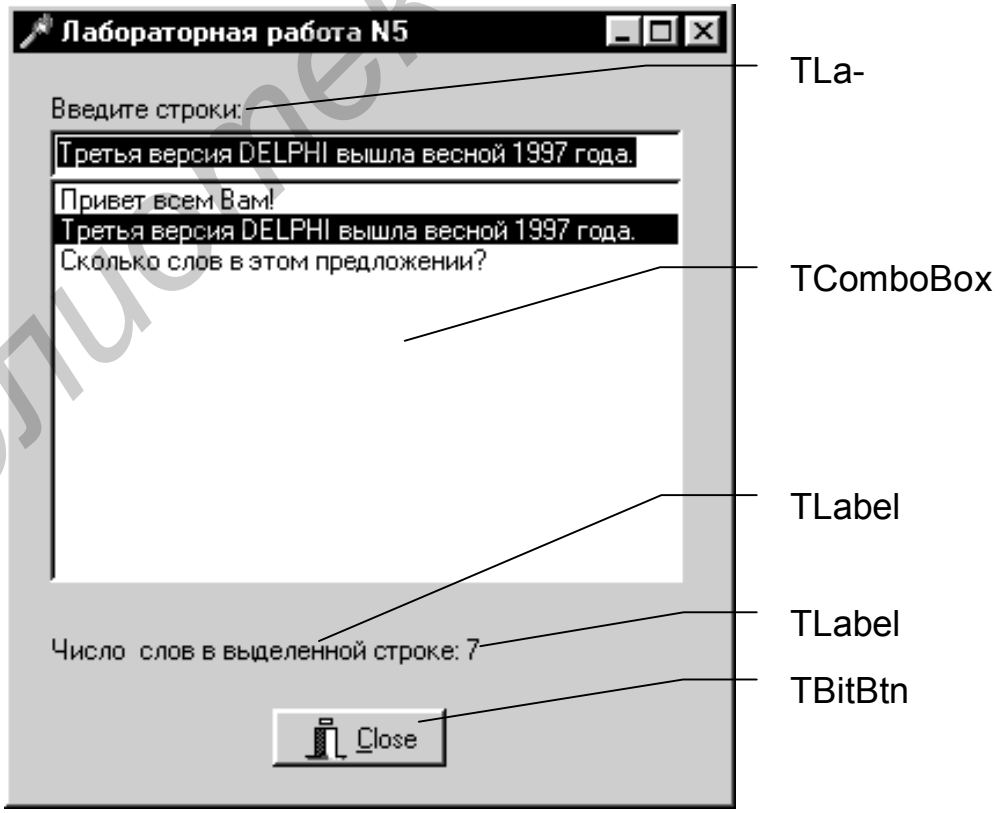


Рис. 8.1. Панель диалога формы

Библиотека БГУИР

8.5.1. Код программы

```
unit Unit1;
interface
uses
  Windows,Messages,SysUtils,Classes,Graphics,Controls,
Forms,Dialogs,StdCtrls,Buttons;
type
 TForm1 = class(TForm)
  Label2: TLabel;
  Label3: TLabel;
  BitBtn1: TBitBtn;
  ComboBox1: TComboBox;
  Label1: TLabel;
  procedure FormActivate(Sender: TObject);
  procedure ComboBox1KeyPress(Sender:TObject;var Key:Char);
  procedure ComboBox1Click(Sender: TObject);
private
 {Private declarations}
public
{Public declarations}
end;
var
 Form1:TForm1;
implementation
{$R *.DFM}
procedure TForm1.FormActivate(Sender:TObject);     // Обработка
begin                                              //  события активации  формы
  ComboBox1.SetFocus;                              // Передача фокуса ComboBox1
end;
procedure TForm1.ComboBox1KeyPress(Sender:TObject;var Key:Char);
begin                 // Обработка события нажатия левой клавиши мыши
 If Key=#13 Then Begin  // При нажатии Enter строка из окна редактирования
    ComboBox1.Items.Add(ComboBox1.Text)  // заносится в список выбора,
    ComboBox1.Text:='';                  // очищается окно редактирования
    End;
  end;
procedure TForm1.ComboBox1Click(Sender: TObject);
Var
 st : String;
 n,i,nst,ind:Integer;
Begin
  n:=0;                                  // Число слов равно 0
  ind:=0;
  nst:=ComboBox1.ItemIndex;    // Определение номера выбранной строки
  st:=ComboBox1.Items[nst]; // Занесение выбранной строки в переменную st
  For i:=1 To Length(st) Do Begin     //Просмотр символов строки st
```

Библиотека БГУИР

```
   Case Ind Of
    0: If st[i]<>' ' Then Begin   // Если после пробела
       Ind:=1;                    // встретился символ, то
       Inc(n);                    // число слов увеличивается на единицу
      End;
    1: If St[i]=' ' Then Ind:=0;//Слово закончилось, т. к. встретился пробел
   End; // Case
  End;     // For
 Label3.Caption:=IntToStr(n);       // Вывод количества слов в Label3
End;
End.
```

8.6. Индивидуальные задания

Дана строка символов, состоящая из произвольного текста, в котором слова разделены пробелами. Составить и отладить программу в соответствии с индивидуальным вариантом. Исходные данные вводить с помощью компонента TEdit в компонент TListBox или с помощью свойства Text в свойство Items компонента TComboBox. Скалярный результат выводить с помощью компонента TLabel. Ввод строки заканчивать нажатием клавиши Enter. Для выхода из программы использовать кнопку Close. Предусмотреть обработку исключительных ситуаций.

1. Найти количество слов, состоящих из пяти символов.
2. Найти самое короткое слово.
3. Заменить пробелы между словами на символ «*».
4. Определить количество символов во втором слове.
5. Вывести символы, расположенные до первого двоеточия.
6. Подсчитать количество слов, начинающихся с буквы a.
7. Найти слова, содержащие букву s.
8. Найти самое длинное слово.
9. Вывести числа строки в порядке возрастания их значений.
10. Вывести четные числа строки.
11. Вывести слова в порядке, соответствующем латинскому алфавиту.
12. Вывести порядковый номер слова, накрывающего k-ю позицию (если на k-ю позицию попадает пробел, то номер предыдущего слова).
13. Разбить исходную строку на две подстроки, причем первая длиной k символов (если на k-ю позицию попадает слово, то его следует отнести ко второй строке, дополнив первую пробелами до k позиций).
14. Вывести порядковый номер слова максимальной длины и номер позиции, с которой оно начинается.
15. Вывести порядковый номер слова минимальной длины и количество символов в нем.
16. В каждом слове заменить первую букву на прописную.
17. Удалить первые k слов, сдвинув на их место последующие слова строки.

Библиотека БГУИР

18. Поменять местами i-е и j-е слова.

19. Поменять местами первую и последнюю буквы каждого слова.

20. Заменить буквы латинского алфавита на соответствующие им буквы русского алфавита.

21. Заменить буквы русского алфавита на соответствующие им буквы латинского алфавита.

22. Определить, является ли текст палиндромом, т. е. читается ли он слева направо так же, как и справа налево (*например* ′А роза упала на лапу Азора′).

23. Вывести римскими цифрами заданное целое число от 1 до 1999.

24. Определить, является ли текст правильной записью римскими цифрами целого числа от 1 до 999. Если является, то вывести это число арабскими цифрами (в десятичной системе).

25. Вывести строчные русские буквы.

26. Вывести в алфавитном порядке различные прописные русские буквы.

27. Переставить слова в обратном порядке.

28. Расположить слова в порядке увеличения их длин.

29. В коде программы заменить Begin и End на Начало и Конец.

30. В каждом слове строки поменять порядок символов на обратный.

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №9

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ЗАПИСЕЙ И ФАЙЛОВ

Цель работы: изучить правила работы с компонентами TOpenDialog и TSaveDialog. Написать программу с использованием файлов и данных типа запись.

9.1. Программирование с использованием переменных типа запись

Запись – это структура данных, объединяющая элементы одного или различных типов, называемых **полями**. Записи удобны для создания структурированных баз данных с разнотипными элементами, *например*:

```
Type
  TStudent=Record                    // Объявление типа запись
    Fio: String[20];                 // Поле ФИО
    Group: Integer;                  // Поле номера студ. группы
    Ocn: Array[1..3] Of Integer;     // Поле массива оценок
  End;
Var
  Student: TStudent;                 // Объявление переменной типа запись
```

Доступ к каждому полю осуществляется указанием имени записи и поля, разделенных точкой, *например*:

```
Student.Fio:='Иванов А.И.';      //Внесение данных в поля записи
Student.Group:=720603;
   ...
```

Доступ к полям записи можно осуществлять при помощи оператора With:

```
With Student Do Begin
    Fio:='Иванов А.И.';
    Group:=720603;
End;
```

9.2. Работа с файлами

Файл – это именованный набор данных на внешнем физическом носителе. В Delphi различают три вида файлов в зависимости от способа их организации и доступа к элементам: **текстовые**, **типизированные** и **нетипизированные**.

Текстовой файл – это файл, состоящий из строк. Каждая строка в таком файле заканчивается двумя специальными символами: #10 – конец строки и #13 – переход на следующую строку. Примером текстового файла может служить файл исходного кода программы в DELPHI (расширение *.pas). Для работы с текстовым файлом должна быть описана соответствующая файловая переменная, *например* `Var F:TextFile;`

Библиотека БГУИР

Типизированные файлы состоят из записей одинаковой длины, которые имеют строго заданную в описании `Record` структуру. Это свойство типизированных файлов позволяет получить доступ к любой записи файла по его порядковому номеру. В описании файловой переменной указывается ее тип, *например*:

```
Var F:TStudent;
```

Нетипизированные файлы похожи на типизированные, только одна запись здесь называется блоком, который воспринимается как последовательность байт. Длина блока по умолчанию – 128 байт. Файловая переменная объявляется как, *например:* `Var F:File;`

Порядок работы с файлами

1. Связывание файловой переменной, *например* F, с именем дискового файла, *например* 'Filename.txt': `AssignFile(F,'Filename.txt');`

2. Создание нового (`Rewrite(F);`) или открытие существующего файла (`Reset(F);`).

3. Чтение данных из файла (`Read(F, Stud);`) или запись в файл (`Write(F,Stud);`).

4. Закрытие файла `CloseFile(F);`

9.3. Подпрограммы работы с файлами

`AssignFile(F,FileName);` – связывание файловой переменной F и файла с именем FileName (String).

`Reset(F);` – открытие существующего файла F.

`Rewrite(F);` – создание и открытие нового файла F; если в файле есть данные, то они стираются.

`Append(F);` – открытие **текстового** файла F для дописывания текста в его конец.

`Read(F,v1[,v2,…vn]);` – чтение из файла F, начиная с текущей позиции, значений переменных `v1[,v2,…vn]` (для типизированных файлов) или строк (для текстовых).

`Write(F,v1[,v2,…vn]);` – запись в файл F, начиная с текущей позиции, значений переменных (для типизированных файлов) или строк (для текстовых).

`CloseFile(F);` – закрытие файла; все открытые файлы должны быть закрыты.

`Rename(F,NewName);` – переименование **неоткрытого** файла `F` любого типа (`NewName:String` – новое имя файла).

`Erase(F);` – удаление **неоткрытого** файла любого типа.

`Seek(F,NumRec);` – установка указателя файла на его элемент с номером NumRec:`Longint` (для **нетекстового** файла).

`SetTextBuf(F,Buf);` – устанавка для **текстового** файла нового буфера ввода – вывода объемом `Size:Word` (по умолчанию размер буфера – 128 байт).

Библиотека БГУИР

`Flush(F);` – немедленная запись в **текстовый** файл содержимого буфера ввода – вывода.

`Truncate(F);` – удаление данных из файла `F`, начиная с текущей позиции.

`LoResult` – код результата последней операции ввода–вывода.

`FilePos(F): Longint` – для нетекстовых файлов возвращает номер текущей позиции. Отсчет ведется от нуля.

`FileSize(F): Longint` – для нетекстовых файлов возвращает количество компонентов в файле.

`Eoln(F):Boolean` – при достижении конца строки в текстовом файле возвращает True.

`Eof(F): Boolean` – при достижении конца файла возвращает True.

`SeekEoln(F):Boolean` – возвращение True, если пройден последний значимый символ в строке или текстовом файле, отличный от пробела или знака табуляции.

`SeekEof(F):Boolean` – то же, что и `SeekEoln`, но для всего текстового файла.

9.4. Компоненты TOpenDialog и TSaveDialog

Компоненты TOpenDialog  и TSaveDialog  находятся на странице Dialogs (см. прил. 2). Все компоненты этой страницы вызываются методом Execute и являются невизуальными, т. е. не видны в момент работы программы. Поэтому их можно разместить в любом удобном месте формы. Оба рассматриваемых компонента имеют идентичные свойства и отличаются только внешним видом. После вызова компонента появляется диалоговое окно, с помощью которого выбирается имя программы и путь к ней. В случае успешного завершения диалога имя выбранного файла и маршрут поиска содержатся в свойстве FileName. Для фильтрации файлов, отображаемых в окне просмотра, используется свойство Filter, а для задания расширения файла, в случае если оно не задано пользователем, – свойство DefaultExt. Для изменения заголовка диалогового окна применяется свойство Title.

9.5. Пример выполнения задания

Составить программу для ввода в файл или чтения из файла ведомости абитуриентов, сдавших вступительные экзамены. Сведения об абитуриенте содержат фамилию и оценки по физике, математике и русскому языку. Вывести список абитуриентов, отсортированный в порядке уменьшения их среднего балла, и записать эту информацию в текстовой файл.

9.5.1. Настройка компонентов TOpenDialog и TSaveDialog

Для установки компонентов TOpenDialog и TSaveDialog на форму надо на странице Dialogs меню компонентов ЛКМ соответственно по пиктограммам 

Библиотека БГУИР

или  и разместить их в свободном месте формы.

Для **установки фильтра** следует:

1) выбрать соответствующий компонент (TOpenDialog или TSaveDialog) и 2ЛКМ по правой части свойства Filter инспектора объектов. Появится окно Filter Editor, в левой части которого записывается текст, характеризующий соответствующий фильтр, а в правой части – маска;

2) значение маски для OpenDialog1 устанавливается в соответствии с рис. 9.1. Формат *.dat означает, что будут видны все файлы с расширением dat, а формат *.* – будут видны все файлы (с любым именем и с любым расширением);

3) для автоматической записи файла с расширением .dat в свойство DefaultExt записывается расширение .dat.

Аналогично настраивается SaveDialog1 для текстового файла (.txt).

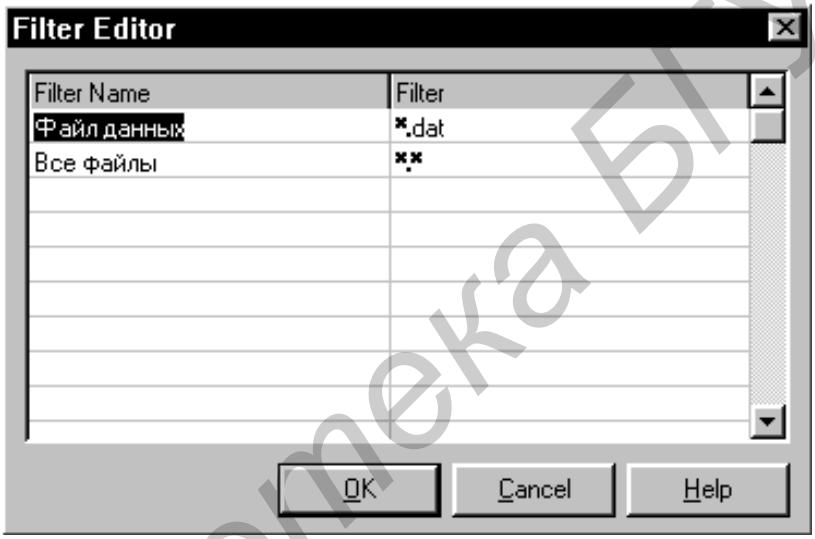


Рис. 9.1. Установка значения маски для TOpenDialog

9.5.2. Работа с программой

После запуска программы на выполнение появится диалоговое окно программы. Кнопка Ввести запись не будет видна. Надо создать новый файл записей, нажав кнопку Создать, или открыть ранее созданный, нажав кнопку Открыть. После этого станет видна кнопка Ввести запись и можно будет вводить записи. При нажатии кнопки Сортировка ведомость абитуриентов будет отсортирована по убыванию среднего балла и диалоговое окно примет вид, показанный на рис. 9.2. При нажатии кнопки Сохранить будет создан текстовый файл, содержащий отсортированную ведомость. Файл записей закрывается одновременно с программой при нажатии кнопки Close или  .

Библиотека БГУИР

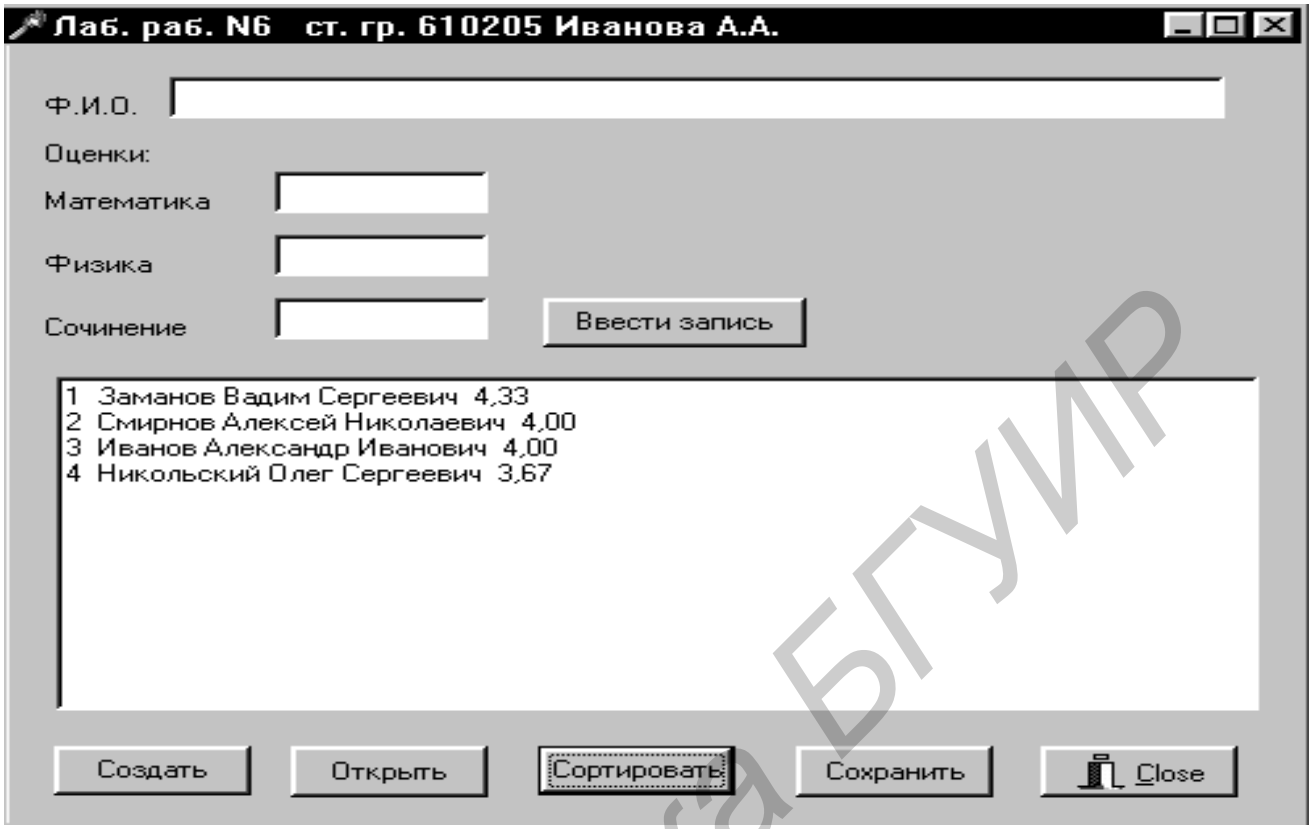


Рис. 9.2. Вид диалогового окна после нажатия кнопки Сортировка

9.5.3. Код программы

```
unit Unit1;
interface
uses
    Windows,Messages,SysUtils,Classes,Graphics,Controls,
Forms,Dialogs,StdCtrls,Buttons,ExtCtrls;
type
  TForm1=class(TForm)
   Edit1:TEdit;
   Edit2:TEdit;
   Edit3:TEdit;
   Edit4:TEdit;
   Label1:TLabel;
   Label2:TLabel;
   Label3:TLabel;
   Label4:TLabel;
   Label5:TLabel;
   Memo1:TMemo;
   Button1:TButton;
```

Библиотека БГУИР

```
   Button3:TButton;
   Splitter1:TSplitter;
   Button5:TButton;
   BitBtn1:TBitBtn;
   SaveDialog1:TSaveDialog;
   Button2:TButton;
   OpenDialog1:TOpenDialog;
   Button4:TButton;
   procedure FormCreate(Sender:TObject);
   procedure Button1Click(Sender:TObject);
   procedure Button2Click(Sender:TObject);
   procedure Button3Click(Sender:TObject);
   procedure Button4Click(Sender:TObject);
   procedure Button5Click(Sender:TObject);
   procedure BitBtn1Click(Sender:TObject);
   procedure FormClose(Sender:TObject;var Action:TCloseAction);
private
    { Private declarations }
public
    { Public declarations }
end;
Type
  Student=Record
    FIO:String[40];                    // Поле ФИО
    Otc:Array[1..3] Of 0..10;          // Поле массива оценок
    Sball:Extended;                    // Поле среднего балла
  End;
Var
  Fz:File Of Student;                  // Файл типа запись
  Ft:TextFile;                         // Текстовый файл
  Stud:Array[1..100] Of Student;       // Массив записей
  nzap:Integer;                        // Номер записи
  fz,ft:String;                        // Имена файлов
  Form1:TForm1;
implementation
{$R *.DFM}
procedure TForm1.FormCreate(Sender: TObject);
 Begin
  Edit1.Text:=' ';
  Edit2.Text:=' ';
```

Библиотека БГУИР

```
  Edit3.Text:=' ';
  Edit4.Text:=' ';
  Memo1.Clear;
  Button1.Hide;   // Сделать невидимой кнопку Ввести запись
  nzap:=0;
End;
procedure TForm1.Button1Click(Sender:TObject);   // Ввести новую
Begin                                                // запись
  nzap:=nzap+1;
  With Stud[nzap] Do Begin
    FIO:=Edit1.Text;
    Otc[1]:=StrToInt(Edit2.Text);
    Otc[2]:=StrToInt(Edit3.Text);
    Otc[3]:=StrToInt(Edit4.Text);
    Sball:=(Otc[1]+Otc[2]+Otc[3])/3;
    Memo1.Lines.Add(Fio+' '+IntToStr(Otc[1])+' '+
                    IntToStr(Otc[2])+' '+IntToStr(Otc[3]));
  End;
  Write(Fz,Stud[nzap]);          // Запись в файл
  Edit1.Text:=' ';
  Edit2.Text:=' ';
  Edit3.Text:=' ';
  Edit4.Text:=' ';
End;
procedure TForm1.Button2Click(Sender:TObject); // Создание
Begin                                              // нового файла записей
 OpenDialog1.Title:='Создать новый файл';//Изменение заголовка окна диалога
 If OpenDialog1.Execute Then // Выполнение стандартного диалога
  Begin                              // выбора имени файла
   FileNameZ:=OpenDialog1.FileName;  // Возвращение имени файла
   AssignFile(Fz,FileNameZ);// Связывание файловой переменной Fz с именем файла
   Rewrite(Fz);              // Создание нового файла
  End;
 Button1.Show;     // Сделать видимой кнопку Ввести запись
End;
procedure TForm1.Button3Click(Sender:TObject); //Открыть
Begin                                              // существующий файл
 If OpenDialog1.Execute then Begin // Выполнение стандартного
                                   // диалога выбора имени файла
  FileNameZ:=OpenDialog1.FileName;  // Запоминание имени файла
```

Библиотека БГУИР

```
  AssignFile(Fz,FileNameZ);//Связывание файловой переменной Fz  с  именем файла
  Reset(Fz);                                  // Открытие существующего файла
 End;
 While Not Eof(Fz) Do Begin
  nzap:=nzap+1;
  Read(Fz,Stud[nzap]);                  // Чтение записи из файла
  With Stud[nzap] Do
  Memo1.Lines.Add(Fio+' '+IntToStr(Otc[1])+' '+
                  IntToStr(Otc[2]) +' '+IntToStr(Otc[3]));
 End;
 Button1.Show;                          // Сделать видимой кнопку Ввести запись
End;
procedure TForm1.Button4Click(Sender:TObject); // Сортировка
Var                                                  //  записей
 i,j:Word;
 st:TStudent;
Begin
 For i:=1 To nzap-1 Do                        // Сортировка массива записей
   For j:=i+1 To nzap Do
    if Stud[i].Sball < Stud[j].Sball Then Begin
      st:=Stud[i];
      Stud[i]:=Stud[j];
      Stud[j]:=st;
    End;
 Memo1.Clear;
 For i:=1 To nzap Do // Вывод в окно Memo1  отсортированных записей
  With Stud[i] Do
   Memo1.Lines.Add(IntToStr(i)+' '+Fio+' '+
                   FloatToStrf(Sball,ffFixed,5,2));
end;
procedure TForm1.Button5Click(Sender:TObject); // Сохранение
Var                                  // результатов сортировки в текстовом файле
 I:Word;
Begin
 If SaveDialog1.Execute Then Begin // Выполнение стандартного
                                                  //  диалога выбора  имени  файла
  FileNameT:=SaveDialog1.FileName;//Возвращение имени дискового файла
  AssignFile(Ft,FileNameT);//Связывание файловой переменной Ft  с именем файла
  Rewrite(Ft);                          // Открытие нового текстового файла
 End;
```

Библиотека БГУИР

```
 For i:=1 To nzap Do
  With Stud[i] Do
    WriteLn(Ft,i:4,'.',Fio:40,Sball:6:2);  //Запись в текстовый файл
 CloseFile(Ft);                            // Закрытие текстового файла
End;
procedure TForm1.BitBtn1Click(Sender: TObject);
 Begin
 CloseFile(Fz); // Закрытие файла записей при нажатии кнопки Close
End;
procedure TForm1.FormClose(Sender:TObject;var Action:TCloseAction);
Begin
  CloseFile(Fz);     // Закрытие файла записей при нажатии кнопки
End;
End.
```

9.6. Индивидуальные задания

Составить и отладить программу в соответствии с вариантом индивидуального задания. В программе предусмотреть сохранение вводимых данных в файл и возможность чтения из ранее сохраненного файла. Результаты выводить в окно просмотра и в текстовой файл. Предусмотреть обработку исключительных ситуаций.

1. В магазине формируется список лиц, записавшихся на покупку товара повышенного спроса. Каждая запись этого списка содержит порядковый номер, ФИО, домашний адрес покупателя и дату постановки на учет. Удалить из списка все повторные записи, проверяя ФИО и домашний адрес.

2. Список товаров, имеющихся на складе, включает наименование товара, количество единиц товара, цену единицы и дату поступления товара на склад. Вывести в алфавитном порядке товары, хранящиеся больше месяца, стоимость которых превышает **1 000 000** рублей.

3. Для получения места в общежитии формируется список студентов, включающий ФИО студента, группу, средний балл, доход на члена семьи. Общежитие, в первую очередь, предоставляется тем, у кого доход на члена семьи меньше двух минимальных зарплат, затем остальным в порядке уменьшения среднего балла. Вывести список очередности предоставления мест в общежитии.

4. В справочной автовокзала хранится расписание движения автобусов. Для каждого рейса указаны его номер, тип автобуса, пункт назначения, время отправления и прибытия. Вывести информацию о рейсах, которыми можно воспользоваться для прибытия в пункт назначения раньше заданного времени.

5. На междугородной АТС информация о разговорах содержит дату разговора, код и название города, время разговора, тариф, номер телефона в этом городе и номер телефона абонента. Вывести по каждому городу общее время разговоров с ним и сумму.

Библиотека БГУИР

6. Информация о сотрудниках фирмы включает ФИО, табельный номер, количество отработанных часов за месяц, почасовой тариф. Рабочее время свыше **144** часов считается сверхурочным и оплачивается в двойном размере. Вывести размер заработной платы каждого сотрудника фирмы за вычетом подоходного налога, составляющего **12 %** от суммы заработка.

7. Информация об участниках спортивных соревнований содержит наименование страны, название команды, ФИО игрока, его игровой номер, возраст, рост и вес. Вывести информацию о самой молодой команде.

8. Для книг библиотеки указывают регистрационный номер, автора, название, год издания, издательство, количество страниц. Вывести список книг с фамилиями авторов в алфавитном порядке, изданных после указанного года.

9. Различные цеха завода выпускают продукцию нескольких наименований. Сведения о выпущенной продукции включают ее наименование и количество, номер цеха. Для заданного цеха вывести записи в порядке убывания количества выпущенных изделий.

10. Информация о сотрудниках предприятия содержит ФИО, номер отдела, должность, дату начала работы. Вывести данные о сотрудниках в порядке убывания стажа.

11. Ведомость абитуриентов, прошедших централизованное тестирование и поступающих в университет, содержит ФИО, адрес, суммарный балл. Определить количество абитуриентов, проживающих в Минске и набравших количество баллов не ниже **280**, и вывести их фамилии в алфавитном порядке.

12. В справочной аэропорта хранится расписание вылета самолетов на следующие сутки. Для каждого рейса указаны номер рейса, тип самолета, пункт назначения, время вылета. Вывести для заданного пункта назначения в порядке возрастания времени вылета номера рейсов, типы самолетов и их время вылета.

13. У администратора железнодорожных касс хранится информация о свободных местах на поезда, включающая дату и время отправления, пункт назначения, число свободных мест. Вывести время отправления, если можно зарезервировать m мест до указанного города со временем отправления поезда не позднее t часов или сообщение о невозможности выполнить заказ.

14. Ведомость студентов, сдавших сессию, содержит ФИО студента и его оценки по трем предметам. Определить средний балл по университету и вывести список студентов, средний балл которых выше среднего балла по университету. Первыми в списке должны идти студенты, сдавшие экзамены на **9** и **10**.

15. В радиоателье хранятся квитанции о сданной в ремонт радиоаппаратуре. Каждая квитанция содержит информацию о наименовании группы изделий (телевизор, радиоприемник и т. п.), марке изделия, дате приемки в ремонт, состоянии готовности заказа (выполнен, не выполнен). Вывести информацию о состоянии заказов на текущие сутки по группам изделий.

Библиотека БГУИР

16. Ведомость об успеваемости студентов содержит номер группы, ФИО студента, оценки за последнюю сессию. Вывести списки студентов по группам.

17. В исполкоме формируется список учета нуждающихся в улучшении жилищных условий. Каждая запись содержит порядковый номер, ФИО, величину жилплощади на одного члена семьи и дату постановки на учет. По заданному количеству квартир, выделяемых в течение года, вывести список очередников с указанием ожидаемого года получения квартиры.

18. Имеются списки женихов и невест, каждая запись которых содержит пол, имя, возраст, рост, вес, а также требования к партнеру: наименьший и наибольший возраст, вес, рост. Объединить данные в пары с учетом требований к партнерам без повторений женихов и невест.

19. В библиотеке имеется список книг, каждая запись которого содержит фамилию автора, название книги, год издания. Вывести информацию о книгах, в названии которых встречается заданное ключевое слово (ввести с клавиатуры).

20. В магазине имеется список поступивших в продажу автомобилей, каждая запись которого содержит марку автомобиля, стоимость, расход топлива на 100 км, надежность (число лет безотказной работы), комфортность (отличная, хорошая, удовлетворительная). Вывести перечень автомобилей, удовлетворяющих требованиям покупателя, задаваемым с клавиатуры в виде интервала допустимых значений.

21. В список вакантных рабочих мест входит наименование организации, должность, квалификация (разряд или образование), стаж работы по специальности, заработная плата, наличие социального страхования (да/нет), продолжительность ежегодного оплачиваемого отпуска. Вывести список рабочих мест в соответствии с задаваемыми требованиями.

22. В технической службе аэропорта имеется справочник, содержащий тип самолета, год выпуска, расход горючего на **1000** км. Для определения потребности в горючем техническая служба запрашивает расписание полетов. Каждая запись расписания содержит номер рейса, пункт назначения, дальность полета. Вывести суммарное количество горючего, необходимое для обеспечения полетов на следующие сутки.

23. Создать каталог книг библиотеки, содержащий сведения об авторе, названии книги, издательстве, годе издания, а также количестве запросов на данную книгу. Определить три наиболее «читаемые» книги и вывести сведения о них.

24. Для участия в конкурсе на замещение вакантной должности желающие подают информацию, включающую ФИО, год рождения, образование (среднее, специальное, высшее), знание иностранных языков (английский, немецкий, французский, владею свободно, читаю и перевожу со словарем), владение компьютером (MS DOS,Windows и т. д.), стаж работы, наличие рекомендаций. Вывести список претендентов в соответствии с требованиями руководства.

25. При постановке на учет в ГАИ автолюбители указывают марку автомобиля, год выпуска, номер двигателя, номер кузова, цвет, номерной знак, ФИО и

Библиотека БГУИР

адрес владельца. Вывести список автомобилей, проходящих техосмотр в текущем году, сгруппированных по маркам автомобилей. Учесть, что в четный год проходят техосмотр автомобили с четными номерами двигателей, иначе – с нечетными.

26. Список студентов группы содержит ФИО, рост и вес. Вывести ФИО студентов, рост и вес которых чаще всего встречается в списке.

27. В магазине имеется перечень ноутбуков с указанием марки, цены, типа процессора, объема оперативной памяти, объема дисковой памяти и их количества. Вывести полную стоимость всех компьютеров.

28. В роддоме ведется учет родившихся детей. В списке указывается ФИО матери ребенка, дата и время его рождения, пол ребенка, его вес и окружность головы, дата выписки из роддома. Определить число детей, родившихся с заданным весом, и вывести информацию о них.

29. В метеоцентре ведется наблюдение за погодой и каждый день делается запись, включающая дату, атмосферное давление, направление и силу ветра, облачность, осадки, влажность. Определить дни с минимальным давлением.

30. В домоуправлении имеются сведения о жильцах района, которые включают сведения об улице, номере дома и квартире, метраже, числе комнат. Для каждого проживающего указывается ФИО, дата рождения, дата прописки и выписки, отношение к владельцу квартиры. Определить число жильцов, проживающих в однокомнатных квартирах и вывести сведения о них.

Библиотека БГУИР

ЛАБОРАТОРНАЯ РАБОТА №10

ПОСТРОЕНИЕ ГРАФИКОВ ФУНКЦИЙ

Цель работы: изучить возможности Delphi для построения графиков функций с помощью компонента отображения графической информации TChart. Написать и отладить программу построения на экране графика заданной функции, использующую внешний модуль `Unit` с подпрограммой.

10.1. Построение графика функции с помощью компонента TChart

Обычно результаты расчетов представляются в виде графиков и диаграмм. Среда Delphi имеет мощный пакет стандартных программ вывода на экран и редактирования графической информации, который реализуется с помощью визуально отображаемого на форме компонента TChart (рис. 10.1).

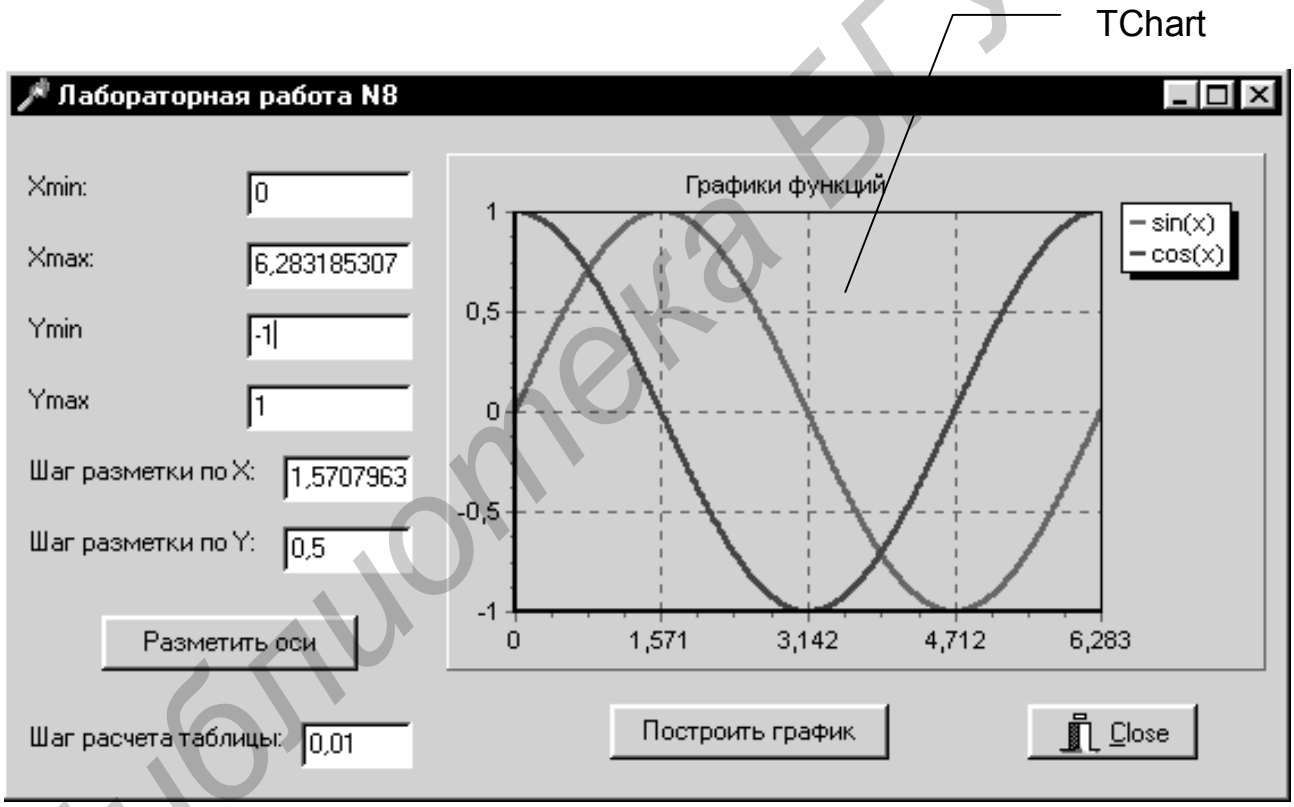


Рис. 10.1. Вывод графиков функций

Построение графика (диаграммы) производится после вычисления таблицы значений функции y=f(x) на интервале [Xmin, Xmax] с заданным шагом. Полученная таблица передается в специальный двумерный массив Seriesk (k – номер графика) компонента TChart с помощью метода Add. Компонент TChart осуществляет всю работу по отображению графиков, переданных в объект Seriesk: строит и размечает оси, рисует координатную сетку, подписывает название осей и самого

Библиотека БГУИР

графика, отображает переданную таблицу в виде всевозможных графиков или диаграмм. При необходимости с помощью встроенного редактора EditingChart компоненту TChart передаются данные о толщине, стиле и цвете линий, параметрах шрифта подписей, шагах разметки координатной сетки и другие настройки. В процессе работы программы изменение параметров возможно через обращение к соответствующим свойствам компонента TChart. *Например,* свойство Chart1.BottomAxis содержит значение максимального предела нижней оси графика, и при его изменении во время работы программы автоматически изменяется изображение графика.

10.2. Пример выполнения задания

Составить программу, отображающую графики функций Sin(x) и Cos(x) на интервале [Xmin, Xmax]. Предусмотреть возможность изменения разметки координатных осей, а также шага построения таблицы. Значения функций Sin(x) и Cos(x) вычислить в библиотечном модуле.

10.2.1. Настройка формы

Окно формы приведено на рис. 10.1.

Для ввода исходных данных используются строки ввода TEdit. Компонент TChart находится в меню компонентов Standard и обозначается пиктограммой  .

10.2.2. Работа с компонентом TChart

Для изменения параметров компонента TChart надо дважды щелкнуть по нему мышью в окне формы. Появится окно редактирования EditingChat1 (рис. 10.2). Для создания нового объекта Series1 следует щелкнуть по кнопке Add на странице Series. В появившемся диалоговом окне TeeChart Gallery выбрать пиктограмму с надписью Line (график выводится в виде линий). Если нет необходимости представления графика в трехмерном виде, надо отключить независимый переключатель 3D. После нажатия на кнопку OK появится новая серия с название Series1. Для изменения названия необходимо нажать кнопку Title… В появившемся однострочном редакторе следует набрать имя отображаемой функции: Sin(x).

Аналогичным образом создается объект Series2 для функции Cos(x).

Для изменения надписи над графиком на странице Titles в многострочном редакторе надо ввести Графики функций.

Для разметки осей выбрать страницу Axis и установить параметры настройки осей.

Нажатие различных кнопок меню способствует ознакомлению с другими возможностями **EditingChat**.

Библиотека БГУИР

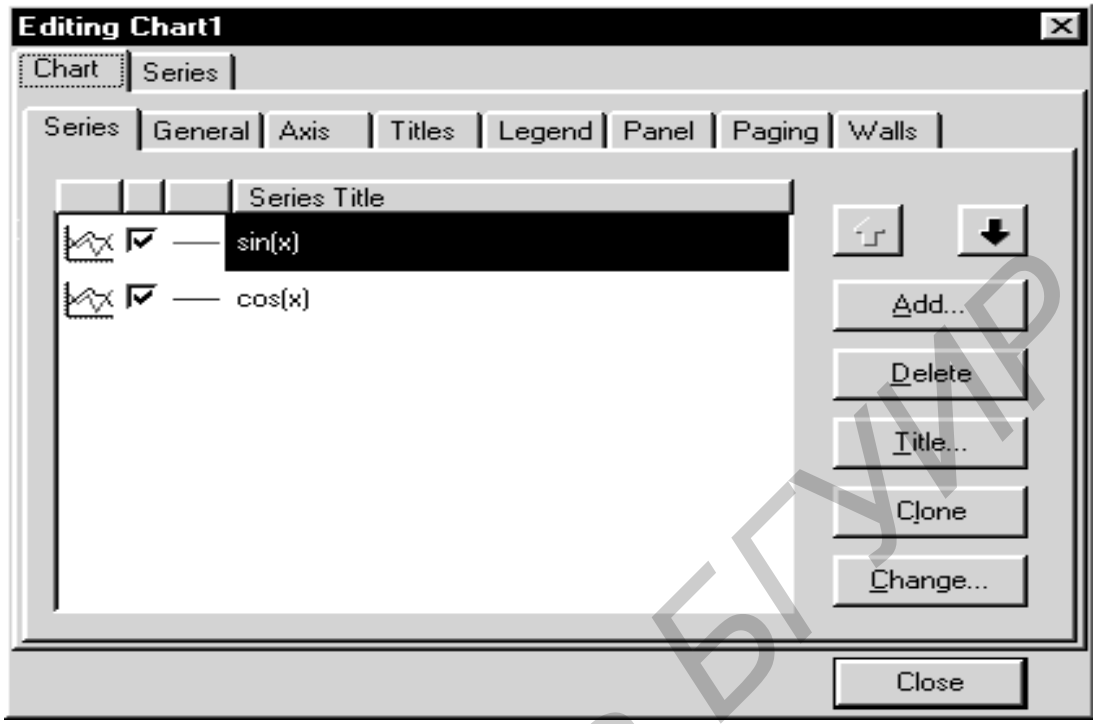


Рис. 10.2. Окно редактирования EditingChat1

10.2.3. Написание программы обработки события создания формы

В данном месте программы устанавливаются начальные пределы и шаг разметки координатных осей. Когда свойство Chart1.BottomAxis.Automatic имеет значения False, автоматическая установка параметров осей не работает.

10.2.4. Написание программ обработки событий нажатия кнопок

Процедура TForm1.Button1Click обрабатывает нажатие кнопки Установить оси. Процедура TForm1.Button2Click обрабатывает нажатие кнопки Построить график. Для добавления координат точек (X,Y) из таблицы значений в двумерный массив объекта **Seriesk** используется процедура Series1.AddXY(Const AXValue, AYValue: Double; Const AXLabel: String; AColor: TColor) : Longint;
где AXValue, AYValue – координаты точки по осям X и Y;

AXLabel может принимать значение `''`;

AColor задает цвет линий (если равен clTeeColor, то принимается цвет, определенный при проектировании формы).

10.2.5. Код библиотечного модуля

```
Unit Matfu;
Interface
Function Sinx(x:Extended):Extended;      // Функция вычисления синуса
Function Cosx(x:Extended):Extended;      // Функция вычисления  косинуса
```

Библиотека БГУИР

```
Implementation
Function Sinx;
Begin
  Result:=Sin(x);
End;
Function Cosx;
Begin
  Result:=Cos(x);
End;
End.
```

10.2.6. Код основного модуля

```
Unit Unit1;
Interface
Uses
 Windows,Messages,SysUtils,Classes,Graphics,Controls,Forms,Dialogs,
ExtCtrls,TeeProcs,TeEngine,Chart,Buttons,StdCtrls,Series, MatF;
Type
 TForm1 = class(TForm)
  Edit1: TEdit;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  Label5: TLabel;
  Edit2: TEdit;
  Edit3: TEdit;
  Edit4: TEdit;
  Edit5: TEdit;
  Button1: TButton;
  Button2: TButton;
  BitBtn1: TBitBtn;
  Chart1: TChart;
  Series2: TLineSeries;
  Label6: TLabel;
  Edit6: TEdit;
  Label7: TLabel;
  Edit7: TEdit;
  Series1: TLineSeries;
  procedure FormCreate(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure Button2Click(Sender: TObject);
```

Библиотека БГУИР

```
private
{Private declarations}
public
{Public declarations}
end;
Var
 Form1:TForm1;
 Xmin,Xmax,Ymin,Ymax,Hx,Hy,h:Extended;
Implementation
{$R *.DFM}
Procedure TForm1.FormCreate(Sender:TObject);
Begin
 Xmin:=0;            //Установка начальных параметров координатных осей
 Xmax:=2*Pi;
 Ymin:=-1;
 Ymax:=1;
 Hx:=Pi/2;
 Hy:=0.5;
 h:=0.01;            //  Установка шага расчета таблицы
 Edit1.Text:=FloatToStr(Xmin);       // Вывод данных
 Edit2.Text:=FloatToStr(Xmax);
 Edit3.Text:=FloatToStr(Ymin);
 Edit4.Text:=FloatToStr(Ymax);
 Edit5.Text:=FloatToStr(Hx);
 Edit6.Text:=FloatToStr(Hy);
 Edit7.Text:=FloatToStr(h);
 Chart1.BottomAxis.Automatic:=False;//Отключение автоматического
                                    // определения параметров нижней оси
 Chart1.BottomAxis.Minimum:=Xmin;//Установка левой границы нижней оси
 Chart1.BottomAxis.Maximum:=Xmax;//Установка правой границы нижней оси
 Chart1.LeftAxis.Automatic:=False;   // Отключение автоматического
                                     // определения параметров левой оси
 Chart1.LeftAxis.Minimum:=Ymin;//Установка нижней границы левой оси
 Chart1.LeftAxis.Maximum:=Ymax;//Установка верхней границы левой оси
 Chart1.BottomAxis.Increment:=Hx;//Установка шага разметки по нижней оси
 Chart1.LeftAxis.Increment:=Hy;//Установка шага разметки по левой оси
End;
Procedure TForm1.Button1Click(Sender: TObject);
Begin
 Xmin:=StrToFloat(Edit1.Text);       //Ввод даннных
```

Библиотека БГУИР

```
 Xmax:=StrToFloat(Edit2.Text);
 Ymin:=StrToFloat(Edit3.Text);
 Ymax:=StrToFloat(Edit4.Text);
 Hx:=StrToFloat(Edit5.Text);
 Hy:=StrToFloat(Edit6.Text);
 Chart1.BottomAxis.Minimum:=Xmin;//Установка левой границы нижней оси
 Chart1.BottomAxis.Maximum:=Xmax;//Установка правой границы нижней оси
 Chart1.LeftAxis.Minimum:=Ymin;  //Установка нижней границы левой оси
 Chart1.LeftAxis.Maximum:=Ymax;  //Установка верхней границы левой оси
 Chart1.BottomAxis.Increment:=Hx;//Установка шага разметки по нижней оси
 Chart1.LeftAxis.Increment:=Hy;//Установка шага разметки по левой оси
End;
Procedure TForm1.Button2Click(Sender:TObject);
Var
 x,y1,y2:Extended;
Begin
  Series1.Clear;                    //Очистка графиков
  Series2.Clear;
  Xmin:=StrToFloat(Edit1.Text);
  Xmax:=StrToFloat(Edit2.Text);
  h:=StrToFloat(Edit7.Text);    // Шаг расчета таблицы для графика
  x:=Xmin;                      // Начальное значение по оси  X
  Repeat
    y1:=Sinx(x);                    // Расчет функции
    Series1.AddXY(x,y1,' ',clTeeColor); // Вывод точки на график
    y2:=Cosx(x);                                // Расчет функции
    Series2.AddXY(x,y2,' ',clTeeColor); // Вывод точки на график
    x:=x+h;                  // Увеличение значения  x  на величину шага
  Until x>Xmax;
End;
End.
```

10.3. Индивидуальные задания

В соответствии с вариантом индивидуального задания составить программу построения графиков функций Y(x) и S(x). Таблицу данных получить, изменяя параметр x от x_n до x_k с шагом $h = (x_k - x_n) / m$, где m – количество интервалов, на которые разбивается отрезок $[x_n , x_k]$. Ввод исходных данных организовать через компоненты TEdit. Вычисление значений функций оформить в отдельном модуле Unit.

Выражения для функций Y(x) и S(x) следует взять из лабораторной работы №4.

Библиотека БГУИР

ПРИЛОЖЕНИЕ 1

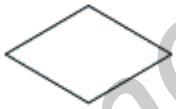
БЛОК-СХЕМА АЛГОРИТМА

Благодаря своей наглядности данный способ записи алгоритмов получил наибольшее распространение. При построении блок-схемы алгоритм изображается геометрическими фигурами (блоками), связанными линиями (направление потока информации) со стрелками. Внутри блоков записывается последовательность действий.

Ниже приведены виды и назначение основных блоков, применяемых при составлении блок-схемы алгоритма.

Таблица П. 1.1

Виды и назначение основных блоков

Наименование блока	0бозначение блока	Функции блока
Процесс		Выполнение действий, изменяющих значение, форму представления или расположение данных
Ввод – вывод		Ввод или вывод данных
Условие		Выбор направления выполнения алгоритма в зависимости от заданного в блоке условия
Предопределенный процесс		Вызов подпрограммы
Пуск – остановка		Начало или конец описания алгоритма
Цикл с параметром		Цикл с параметром; внутри блока указывается изменение параметра (переменной цикла), *например* i = 1,10
Внутристраничный соединитель		Переход между блоками в пределах данной страницы
Межстраничный соединитель		Переход между блоками, расположенными на разных листах
Комментарий		Пояснение действий, указанных в блоке

Библиотека БГУИР

Правила оформления блок-схем:

– в пределах одной схемы блоки изображают одинаковых размеров;

– блоки нумеруются в верхнем левом углу блока с разрывом линии;

– линии, соединяющие блоки и указывающие последовательность связей между ними, проводятся параллельно линиям рамки;

– стрелка в конце линии не ставится, если линия направлена слева направо или сверху вниз;

– из блока «условие» могут выходить две линии, из других блоков – только одна линия;

– если схема занимает более одного листа, то в случае разрыва линии используется межстраничный соединитель;

– внутри каждого соединителя указывается номер блока (откуда или куда направлена соединительная линия). Внутри межстраничного соединителя в первой строке указывается номер листа, во второй – номер блока, куда или откуда передается управление.

В зависимости от поставленной задачи выделяют три основных вида алгоритмов:

– линейный;

– разветвляющийся;

– циклический.

Линейным называется алгоритм, в котором все действия, указанные в блоках, выполняются по порядку их следования.

Пример блок-схемы алгоритма вычисления площадей прямоугольника и квадрата (рис. П. 1.1).

Библиотека БГУИР

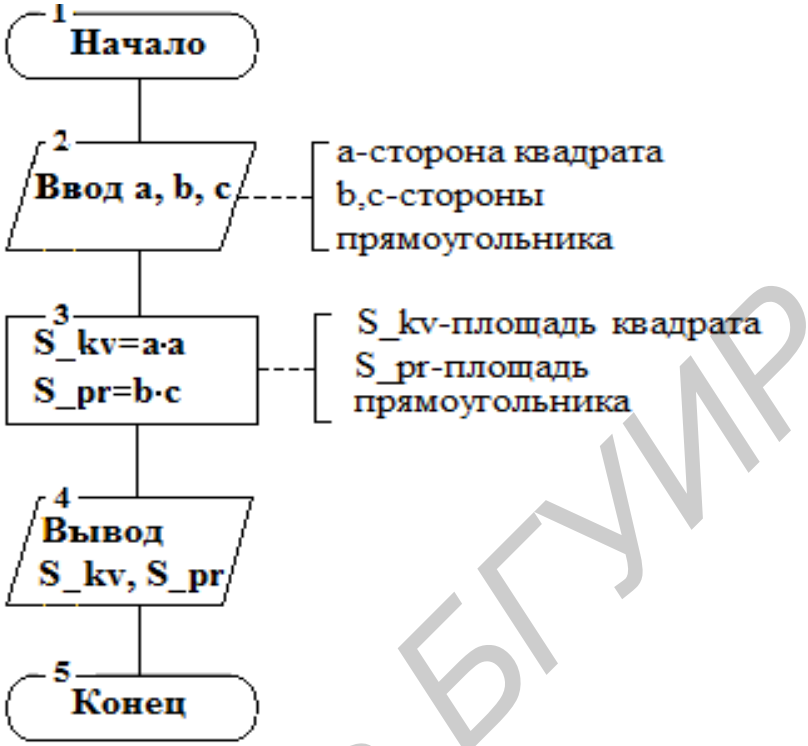


Рис. П. 1.1. Блок-схема линейного алгоритма

Разветвляющимся называют алгоритм, в котором в зависимости от значения условия (выполняется или не выполняется) изменяется последовательность выполнения действий алгоритма.

Пример блок-схемы алгоритма определения принадлежности точки с координатами (x, y) номеру сектора (рис. П. 1.2).

Библиотека БГУИР

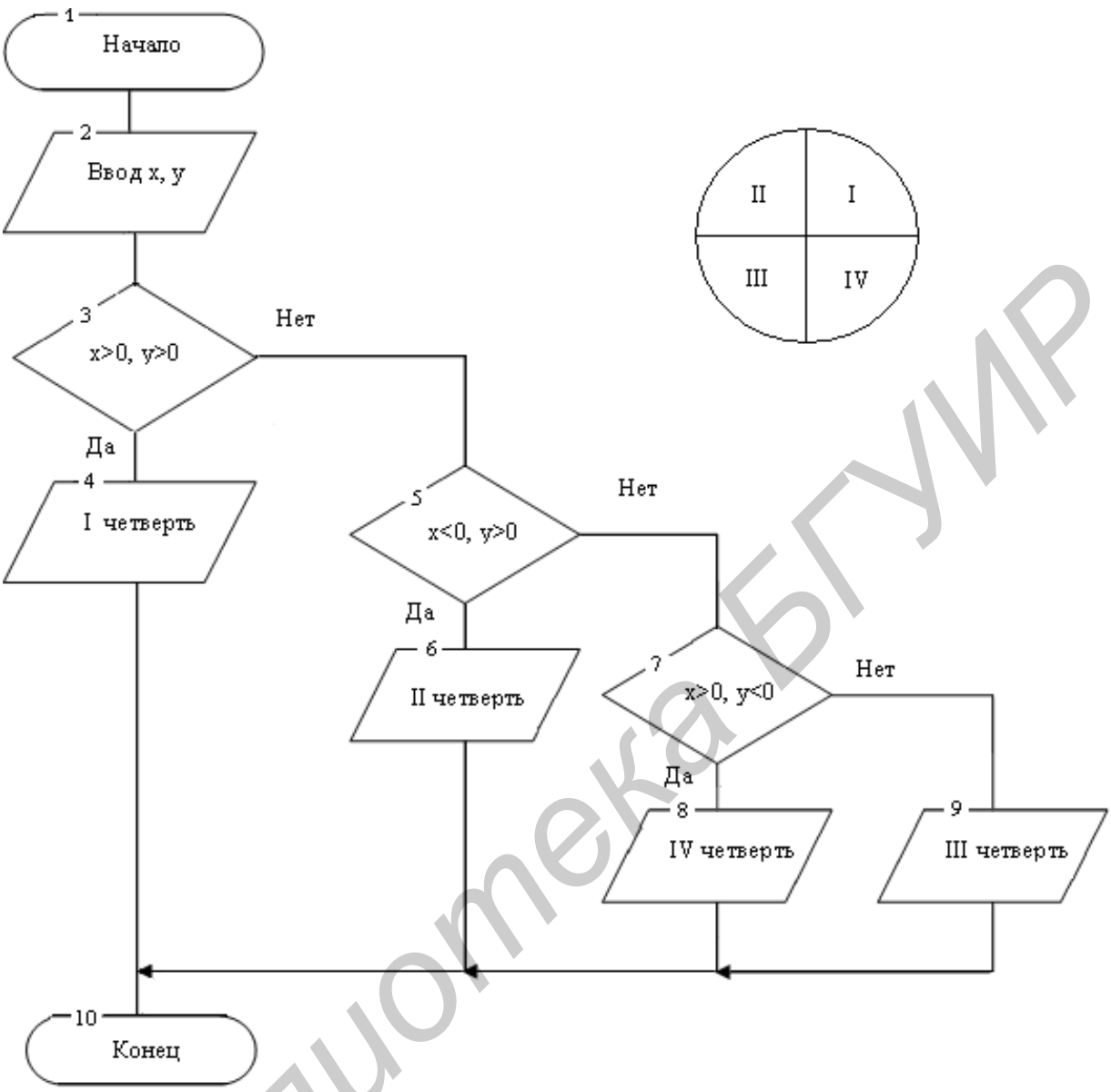


Рис. П. 1.2. Блок-схема разветвляющегося алгоритма

Циклическим называется алгоритм, в котором некоторая последовательность действий повторяется определенное количество раз.

Тело цикла – действия, выполняемые в цикле.

Циклические алгоритмы могут быть:

– **с предусловием** – условие выполнения тела цикла задается в начале цикла (рис. П. 1.3);

– **с постусловием** – условие выполнения тела цикла задается в конце цикла (рис. П. 1.4);

– **с параметром** – в начале цикла задается правило изменения его параметра (рис. П. 1.5).

Библиотека БГУИР

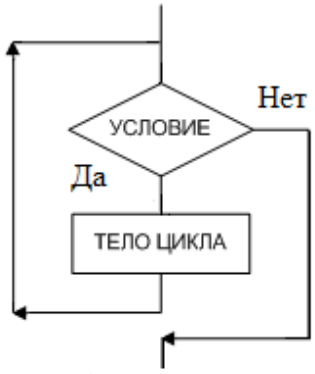


Рис. П. 1.3. Цикл с предусловием

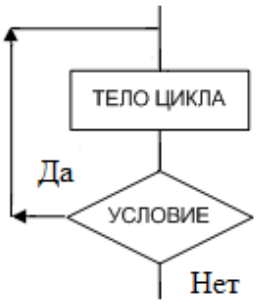


Рис. П. 1.4. Цикл с постусловием

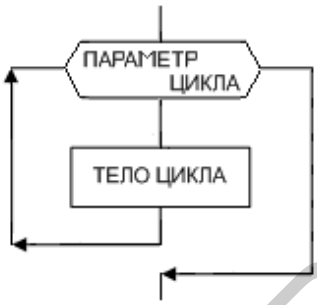


Рис. П. 1.5. Цикл с параметром

Приведем алгоритм нахождения НОД (a, b) (наибольшего общего делителя).

1. Определим наибольшее из значений a и b → max (a, b).
2. Из большего значения вычтем меньшее.
3. Пп. 1 и 2 будем повторять, пока значения a и b не станут равными.

Полученное значение будет НОД (a, b).

Например, если a = 20 и b = 15, то

max(a, b) = max(20, 15) =20 → a = 20 – 15 = 5 b = 15

max(a, b) = max(5, 15) =15 → a = 5 b = 15 – 5 = 10

max(a, b) = max(5, 10) =10 → a = 5 b = 10 – 5 = 5,

т. к. a = b, то НОД(a, b) = 5

Блок-схема рассмотренного алгоритма нахождения НОД (a, b) приведена на рис. П. 1.6.

Библиотека БГУИР

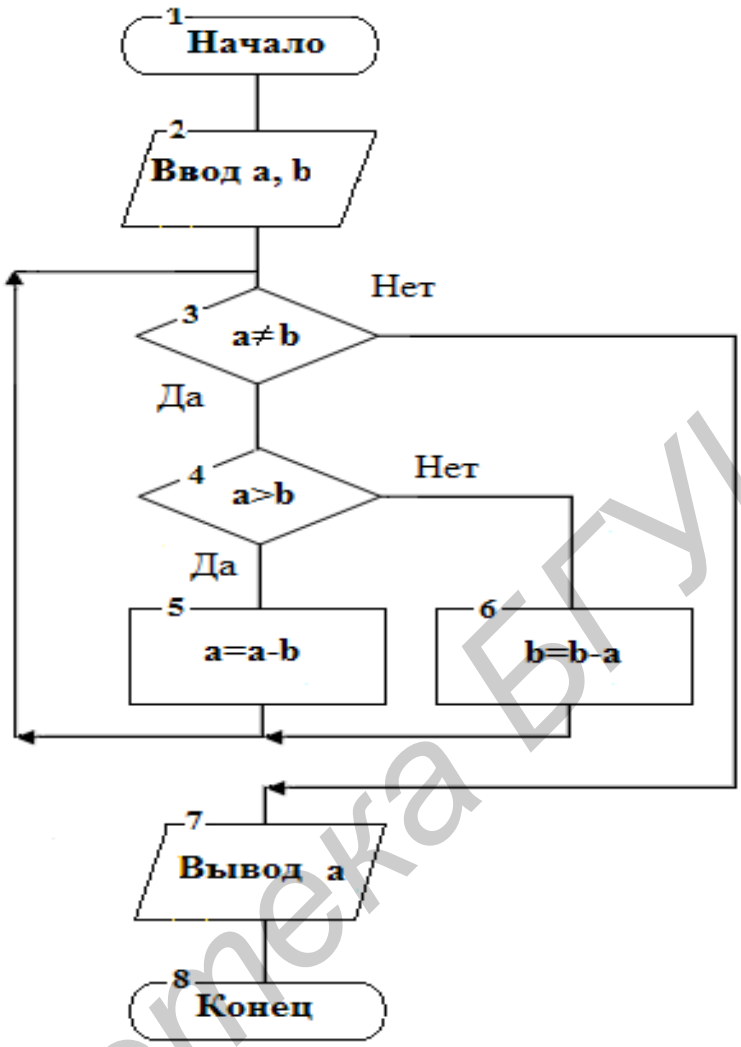


Рис. П. 1.6. Блок-схема алгоритма нахождения НОД (a, b)

Библиотека БГУИР

ПРИЛОЖЕНИЕ 2

МАТЕМАТИЧЕСКИЕ ФОРМУЛЫ

Наиболее часто встречающиеся стандартные математические функции приведены в таблице П. 2.1.

Таблица П. 2.1

Стандартные математические функции

Функция	Назначение	Тип аргумента	Тип функции
`Abs(x)`	Вычисление абсолютного значения (модуля) x, |x|	Вещественный Целый	Такой же, как и тип аргумента
`Sqr(x)`	Вычисление квадрата x, x^2	Вещественный Целый	Такой же, как и тип аргумента
`Sqrt(x)`	Вычисление квадратного корня из x, $\sqrt{x}$, x>0	Вещественный Целый	Вещественный
`Sin(x)`	Вычисление синуса x, sin x	Вещественный Целый	Вещественный
`Cos(x)`	Вычисление косинуса x, cos x	Вещественный Целый	Вещественный
`Arctan(x)`	Вычисление арктангенса x, arctg x	Вещественный Целый	Вещественный
`Exp(x)`	Вычисление экспоненты (e ≈ 2,71828) e^x, e –> exp(1)	Вещественный Целый	Вещественный
`Ln(x)`	Вычисление натурального логарифма x, $\ln x$, x > 0	Вещественный Целый	Вещественный
`Log(x)`	Вычисление десятичного логарифма, log x	Вещественный Целый	Вещественный
`Pi`	Число π ≈ 3,1415…	Нет	Вещественный
`Odd(x)`	Проверка числа на нечетность (True), `Not Odd(x)` – на четность	Целый	Логический
`Inc(x)`	Увеличение x на 1	Целый	Целый
`Inc(x,n)`	Увеличение x на n	Целый	Целый
`Dec(x)`	Уменьшение x на 1	Целый	Целый
`Dec(x,n)`	Уменьшение x на n	Целый	Целый

Библиотека БГУИР

Функции модуля Math

Наиболее часто используемые функции расширенного списка основных математических функций указаны в таблице П. 2.2.

Для использования в программе расширенного списка основных математических функций надо в разделе `Uses` дописать в конец списка математический модуль `Math`: `Uses .....,Math;`

Таблица П. 2.2

Наиболее часто используемые функции модуля `Math`

Функция	Описание	Тип результата и аргументов
`Power(a,x)`	a^x – возведение a в степень x	`Extended`
`ArcCos(X)`	ArcCos x	`Extended`
`ArcSin(X)`	ArcSin x	`Extended`
`Tan(X)`	Tg x, $x \neq 0$	`Extended`
`ArcTan2(Y,X)`	Результат в диапазоне $-\pi..+\pi$	`Extended`
`Cotan(X)`	$\frac{1}{\mathrm{tg}x}$	`Extended`
`Secant(X)`	$\frac{1}{\cos x}$	`Extended`
`Cosecant(X)`	$\frac{1}{\sin x}$	`Extended`
`Hypot(X,Y)`	$\sqrt{x^2+y^2}$	`Extended`
`Cosh(X)`	$\frac{e^x+e^{-x}}{2}$	`Extended`
`Sinh(X)`	$\frac{e^x-e^{-x}}{2}$	`Extended`
`LogN(N,X)`	$\mathrm{Log}_n(x)$	`Extended`
`Sign(x)`	1, если x > 0; 0, если x = 0; –1, если x < 0	`Extended`

Библиотека БГУИР

В таблице П. 2.3 перечислены целочисленные операции вычисления кратного и остатка от деления.

Таблица П. 2.3

Целочисленные операции

Операция	Назначение	Пример записи	Тип операндов	Тип
`Div`	Вычисление частного при делении a на b	`c:=a Div b;`	Целый	Целый
`Mod`	Вычисление остатка от деления a на b	`d:=a Mod b;`	Целый	Целый
`Shr`	Побитовый сдвиг вправо	`k:=I shr 2;`	Целый	Целый
`Shl`	Побитовый сдвиг влево	`k:=I shl 3;`	Целый	Целый

Примеры:
```
k:=18 div 7;   // результат k = 2
k:=18 mod 7;   // результат k = 4
k:=1 shl 3;    // результат k = 8
k:=8 shr 2;    // резальтат k = 2
```

В таблице П. 2.4 рассмотрены функции преобразования.

Таблица П. 2.4

Функции преобразования

Функция	Назначение	Тип аргумента	Тип функции
`Trunc(x)`	Нахождение целой части x (дробная часть числа отбрасывается)	Вещественный целый	LongInt
`Round(x)`	Округление x в сторону ближайшего целого по математическим правилам	Вещественный целый	LongInt
`Int(x)`	Вычисление целой части x	Вещественный	Вещественный
`Frac(x)`	Вычисление дробной части числа x	Вещественный	Вещественный

Примеры:
```
y:=Trunc(13.999);  // y = 13
y:=Trunc(13.111);  // y = 13
y:=Round(3.145);   // y = 3
y:=Round(23.5);    // y = 24
y:=Round(-12.5);   // y = –13
y:=Int(2.7);       // y = 2
y:=Int(-32.3);     // y = –32
y:=Frac(-32.3);    // y = –0.3
```

Библиотека БГУИР

Функции получения случайных чисел и примеры их записи приведены в таблице П. 2.5.

Таблица П. 2.5

Получение случайных чисел

Функция	Назначение	Пример записи	Тип аргумента и результата
`Random`	Получение (**генерация**) случайного числа в диапазоне 0 ≤ … < 1	`y:=Random;`	Вещественный
`Random(x)`	Получение случайного числа в диапазоне 0 ≤ … < x	`y:=Random(39);`	Целый

В таблице П. 2.6 указаны функции, используемые для работы с порядковыми переменными.

Таблица П. 2.6

Функции, используемые для работы с порядковыми переменными

Функция	Назначение	Тип аргумента	Тип функции
`Pred(x)`	Определение предшественника взятого символа x	Порядковый	Порядковый
`Succ(x)`	Определение последующего символа за взятым символом x	Порядковый	Порядковый
`Ord(x)`	Определение кода символа, *например* `Ord('A')` → 65	Порядковый	Целый
`Chr(x)`	Определение символа по коду, *например* `Chr(65)` → 'A'	Целый	Char

В таблице П. 2.7 рассмотрены процедуры и функции работы со строками.

Таблица П. 2.7

Процедуры и функции работы со строками

Функция	Описание
1	2
`Concat` (S1, S2, ..., SN)	Возвращает строку, представляющую собой сцепление строк – параметров S1, S2, …, SN
`Copy` (St,Index,Count)	Копирует из строки St Count:Integer символов, начиная с символа с номером Index:Integer
`Delete` (St,Index,Count);	Удаляет Count:Integer символов из строки St, начиная с символа с номером Index (тип Integer)
`Insert` (SubSt, St,Index);	Вставляет подстроку SubSt в строку St, начиная с символа с номером Index:Integer
`Length` (St)	Возвращает текущую длину (тип Integer) строки St

Библиотека БГУИР

Продолжение табл. П. 2.7

1	2
Pos (SubSt, St)	Определение номера позиции (тип Integer) первого вхождения подстроки SubSt в строке St. Если подстрока не найдена, то возвращается **нуль**
SetLength (St,NewLength);	Устанавливает новую (меньшую) длину NewLength: Integer строки St, если NewLength больше длины строки, то обращение к SetLength игнорируется
Преобразование строк в другие типы	
StrToCurr (St)	Преобразует строку St в целое число типа Currency. Строка не должна содержать ведущих или ведомых пробелов
StrToDate (St)	Преобразует строку St в дату типа TDateTime. Строка должна содержать 2 или 3 числа, разделенных правильным для Windows разделителем даты (в русифицированной версии таким разделителем является «.»). Первое число – день, второе – месяц, при задании третьего числа задается год
StrToFloat (St)	Преобразует символы строки St в вещественное число. Строка не должна содержать ведущих или ведомых пробелов
StrToInt (St)	Преобразует строку St в целое число. Строка не должна содержать ведущих или ведомых пробелов
StrToIntDef (St,Default)	Преобразует строку St в целое число. Если строка не содержит правильного представления целого числа, то возвращается значение Default: Integer
StrToIntRange (St, Min, Max)	Преобразует строку St в целое число (тип Longint) и возбуждает исключение ERangeError, если число выходит из диапазона Min,Max:Longint
StrToTime (St)	Преобразует значение строки St: String во время
Val (St, X, Code);	Преобразует строку St во внутреннее представление целой или вещественной переменной X, определяемое ее типом. Параметр Code: Integer = **0**, если преобразование прошло успешно и в X помещается результат преобразования; иначе Code содержит номер позиции ошибочного символа в строке St и тогда содержимое X не меняется. В строке St могут быть ведущие и (или) ведомые пробелы
Подпрограммы обратного преобразования	
DateToStr (Value)	Преобразует дату из параметра Value: TDateTime в строку символов
DateTimeToStr (Value)	Преобразует дату и время из параметра Value: TDateTime в строку символов

Библиотека БГУИР

Окончание табл. П. 2.7

1	2
DateTimeToString (St,Format,Value) ;	Преобразует дату и время из параметра Value: TDateTime формата Format:String в строку St
FormatDateTime (Format,Value)	Преобразует дату и время из параметра Value: TDateTime формата Format:String в строку
FloatToStr (Value)	Преобразует вещественное значение Value: Extended в строку символов
FloatToStrF (Value, Format, Precision, Digits)	Преобразует вещественное значение Value: Extended формата Format:String в строку символов с учетом параметров Precision и Digits типа Integer (см. пояснения ниже)
FormatFloat (Format, Value)	Преобразует вещественное значение Value: Extended формата Format: String в строку
TimeToStr (Value)	Преобразует время Value:TDateTime в строку
Str (X:Width:Decimals,St);	Преобразует X вещественного или целого типа в строку St. Параметры Width и Decimals (могут отсутствовать) задают формат преобразования: Width – общую ширину поля, выделенного под символьное представление числа X, Decimals – количество символов в дробной части (когда X: Extended)

Параметры функции FloatToStrF перечислены в таблице П. 2.8.

Таблица П. 2.8

Правила использования параметров функции FloatToStrF

Значение Format	Описание
1	2
ffExponent	Научная форма представления с множителем eXX («умножить на 10 в степени XX»). Precision задает общее количество десятичных цифр мантиссы. Digits – количество цифр в десятичном порядке XX. Число округляется с учетом первой отбрасываемой цифры, *например* 3.1416E+00
ffFixed	Формат с фиксированным положением разделителя целой и дробной частей. Precision задает общее количество десятичных цифр в представлении числа. Digits – количество цифр в дробной части. Число округляется с учетом первой отбрасываемой цифры. *Например,* при значении Digits, равным 2,будет выведено 3,14

Библиотека БГУИР

Окончание табл. П. 2.8

1	2
ffGeneral	Универсальный формат, использующий наиболее удобную для чтения форму представления вещественного числа. Соответствует формату ffFixed, если количество цифр в целой части меньше или равно Precision, а само число – больше или равно 0,00001; иначе соответствует формату ffExponent, например 3,1416
ffNumber	Отличается от ffFixed использованием символа-разделителя тысяч при выводе больших чисел (для русифицированной версии Windows таким разделителем является **пробел**). Для Value = π* 1000 получим 3 141,60
ffCurrency	Денежный формат. Соответствует ffNumber, но в конце строки ставится символ денежной единицы (для русифицированной версии Windows – символы «р.»). Для Value = π · 1000 получим 3 141,60 р.

Библиотека БГУИР

ПРИЛОЖЕНИЕ 3
НАСТРОЙКА ПАРАМЕТРОВ СРЕДЫ DELPHI

Для упрощения работы с любой программой в среде Delphi следует задать новую переменную среды окружения, *например* Lab1. Для этого надо запустить Delphi и установить настройки: Tools – Environmet Options – Environment Variables – в окне User Overrides задать новый параметр с именем (Variable Name), *например* Lab1, и значением (Variable Value), описывающем путь к основному каталогу программы, *например* d:\Work\Mylab\Lab1.

Обычно при профессоинальном программировании в этом каталоге создают подкаталоги для хранения:

– исходного кода программы (*например* Sourse), где будут храниться файлы с расширениями *.dpr, *.pas, *.dfm, *.res, *.cfg, *.dof;

– результатов трансляции модулей Unit (*например* Lib) – файлы с расширением *.dcu;

– готовой к выполнению программы и динамических библиотек (*например* Bin) – файлы с расширениями *.exe и *.dll.

В этом каталоге можно создать подкаталог для описания программы и правил работы с ней и подкаталог исходных данных для различных вариантов расчетов в программе.

Затем сохранить файл проекта (File – Save Project As…) и определить для него директорию, *например* d:\ Work \ Mylab \ Lab1 \ Source \ Project.dpr.

В основном меню выбрать Project – Options – Directories / Conditionals и в окне Directores задать выходную директорию (**Output** Directories), *например* $(Lab1) \ BIN. В этом окне надо определить директорию для результатов трансляции модулей (Unit Output Directory), *например* $(Lab1) \ LIB.

Запустить трансляцию и выполнение текущей программы (меню Run).

При сохранении проекта File – Save сохранятся текущие настройки проекта.

Следующий запуск программы не потребует повторения указанных настроек.

Для режима отладки программы надо отключить режим оптимизации при работе трансляции, т. к. он иногда не позволяет проводить отладку программы. Для этого следует задать путь: Project – Options – Compiler. Вид формы должен соответствовать рис. П. 3.1.

Библиотека БГУИР

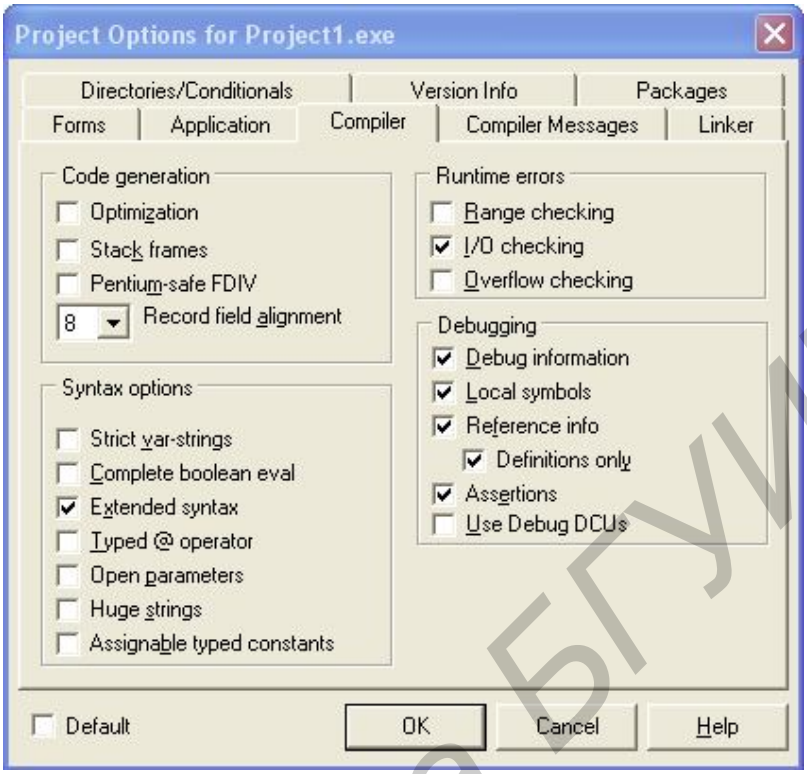


Рис. П. 3.1. Вид формы окна вкладки Compiler

Библиотека БГУИР

ПРИЛОЖЕНИЕ 4

СВОЙСТВА КОМПОНЕНТОВ

Свойства стандартных визуальных компонентов рассмотрены в табл. П. 4.1 – П. 4.8.

Таблица П. 4.1

Базовые свойства VCL-компонентов

Свойство	Назначение
1	2
Alignment	Определяет горизонтальное выравнивание текста относительно границ компонента: taCenter (по центру), taLeftJustify (по левому краю), taRightJustify(по правому краю)
Caption или Text (TEdit)	Заголовок компонента
Color	Задает цвет фона компонента, который выбирается из стандартных, перечисленных в списке или вводимых с клавиатуры. *Например,* `Color:=$00FF0000;` определяет ярко голубой цвет. Младший байт задает уровень красного цвета, второй байт – уровень зеленого, третий байт – уровень синего (RGB)
Ctl3D	Задает вид компонента. Если значение свойства False – компонент имеет двумерный вид, True – трехмерный (значение по умолчанию)
Cursor	Определяет вид курсора мыши в активной области компонента
DragCursor	Определяет вид курсора мыши при перемещении другого компонента в данный
DragMode	Определяет режим поддержки протокола drag–and–drop. Возможные значения: DmAutomatic – компонент можно перемещать мышью, dmManual – компонент не может быть перемещен без вызова метода BeginDrag
Enabled	True – компонент реагирует на сообщения от мыши, клавиатуры и таймера; иначе (False) – эти сообщения игнорируются.
Font	Определяет шрифт текстовых элементов компонента. При создании компонента устанавливаются параметры: Name=System, Size=10, Color=clWindowText, Pitch= FpDefault. При нажатии ⊞ раскрываются свойства шрифта, на ··· – окно установки свойств шрифта.
HelpContext	Задает номер контекста справочной системы (должен быть уникальным для каждого компонента). Если компонент активен (находится в фокусе), то нажатие F1 выводит окно справочной системы, если оно существует для данного компонента

Библиотека БГУИР

Продолжение табл. П. 4.1

1	2
Height	Задает вертикальный размер компонента (в пикселах), вместе со свойствами Width, Left и Top задает его размер и положение
Hint	Задает текст, который будет отображаться при обработке события OnHint, если курсор находится в области компонента
Left	Задает горизонтальную координату левого угла компонента относительно формы в пикселах. Для формы это значение указывается относительно экрана
Name	Указывает внутреннее имя компонента, используемое в программном коде для обращения к объекту. Является идентификатором
ParentColor	Определяет цвет компонента: если True (по умолчанию), используется цвет родительского компонента, иначе (False) компонент использует значение собственного свойства Color. При смене свойства Color значение ParentColor автоматически меняется на False
ParentCtl3D	Указывает способ определения вида компонента (трехмерный или нет): значение свойства True – вид компонента задается значением свойства Ctl3D его владельца, значение свойства False – значением его собственного свойства Ctl3D
ParentFont	Аналогично свойствам ParentColor и ParentCtl3D, но для шрифта: True – используется шрифт, заданный у владельца компонента, False – шрифт задается значением собственного свойства Font
PopupMenu	Задает название локального меню, отображаемое при нажатии правой кнопки мыши. Локальное меню отображается, когда свойство AutoPopup=True или при вызове метода Popup
ReadOnly	Запрещает редактирование текста, отображаемого в TEdit (значение True)
TabOrder	Задает очередность получения компонентами фокуса при нажатии клавиши Tab. По умолчанию определяется порядком размещения компонентов на форме: у первого компонента TabOrder=0, у второго – 1 и т. д. Компонент с TabOrder=0 получает фокус при выводе формы.
TabStop	Указывает возможность получения фокуса для компонента. Компонент получает фокус, если TabStop равно True
Tag	«Привязывает» к любому компоненту значение типа LongInt
Top	Задает вертикальную координату левого верхнего угла интерфейсного элемента относительно формы в пикселах. Для формы это значение указывается относительно экрана

Библиотека БГУИР

Окончание табл. П.4.1

1	2
Visible	Определяет видимость компонента на экране. Значением этого свойства управляют методы Show и Hide
Width	Задает горизонтальный размер интерфейсного элемента или формы в пикселах

Таблица П. 4.2

Выравнивание компонента внутри родителя (свойство Align)

Значение	Расположение компонента
alNone	Выравнивание не используется; располагается там, куда был помещен во время создания проекта. Принимается по умолчанию
alTop	Перемещается в верхнюю часть родительского окна, а его ширина становится равной ширине родительского окна. Высота не изменяется
alBottom	Перемещается в нижнюю часть родительского окна, а его ширина становится равной ширине родительского окна. Высота не изменяется
alLeft	Перемещается в левую часть родительского окна, а его высота становится равной высоте родительского окна. Ширина не изменяется
alRight	Перемещается в правую часть родительского окна, а его высота становится равной высоте родительского окна. Ширина не изменяется
alClient	Занимает всю рабочую область родительского окна

Таблица П. 4.3

Задание цвета фона компонента (свойство Color

Значение	Цвет
clBlack	Черный (Black)
clMaroon	Темно-красный (Maroon)
clGreen	Зеленый (Green)
clOlive	Оливковый (Olive green)
clNavy	Темно-синий (Navy blue)
clPurple	Фиолетовый (Purple)
clTeal	Сине-зеленый (Teal)
clGray	Серый (Gray)

Значение	Цвет
clSilver	Серебряный (Silver)
clRed	Красный (Red)
clLime	Ярко-зеленый (Lime green)
clBlue	Голубой (Blue)
clFuchsia	Сиреневый (Fuchsia)
clAqua	Ярко-голубой (Aqua)
dWhite	Белый (White)

Библиотека БГУИР

Таблица П. 4.4

Системные цвета Windows, определяемые цветовой схемой

Значение	Цвет для элемента
clBackground	Фон окна
clActiveCaption	Заголовок активного окна
clInactiveCaption	Заголовок неактивного окна
clMenu	Фона меню
clWindow	Фон Windows
clWindowFrame	Рамка окна
clMenuText	Текст элемента меню
clWindowText	Текст внутри окна
clCaptionText	Заголовок активного окна
clActiveBorder	Рамка активного окна
clInactiveBorder	Рамка неактивного окна
clAppWorkSpace	Рабочая область окна
clHighlight	Фон выделенного текста
clHightlightText	Выделенный текст
clBtnFace	Кнопка
clBtnShadow	Фон кнопки
clGrayText	Недоступный элемент меню
clBtnText	Текст кнопки

Таблица П. 4.5

Свойства компонента TCheckBox
(позволяет выбрать или отменить определенную функцию)

Свойство	Назначение
Alignment	Определяет положение размещения надписи текста кнопки: taLeftJustify (с левой стороны компонента), taRightJustify (с правой)
AllowGrayed	Значение False (по умолчанию) –2 состояния флажка: выделен (не выделен), значение True – 3 состояния флажка: выделен, не выделен, промежуточное (серое окно индикатора и серая галочка –)
Caption	Надпись возле компонента TCheckBox
Checked	Содержит выбор пользователя типа Да/Нет. При значении True компонент выделен (установлена черная галочка –), False – не выделен (пустое окно индикатора –)
State	Устанавливает значение кнопки, которая может находиться во включенном, выключенном и неактивном состоянии: cbChecked (выделен), cbUnchecked (не выделен) и **cbGrayed** (промежуточное значение) при значении True у свойства AllowGrayed

Библиотека БГУИР

Таблица П. 4.6

Свойства компонента TListBox
(содержит список элементов, выбираемых мышью или клавиатурой)

Свойство	Назначение
Columns	Определяет количество столбцов в списке
MultiSelect	Разрешает (True) или отменяет (False) выбор нескольких строк из списка
SelCount	Определяет количество выделенных строк в списке. Свойство доступно только для чтения, не может быть изменено в проекте, не отражается в инспекторе объектов
Selected[N]	Содержит признак выбора для элемента с номером n (нумерация начинается с нуля): True – элемент выделен в списке. Не отражается в инспекторе объектов
ItemIndex	Содержит индекс выбранного элемента в списке; по умолчанию = –1
Items	Содержит набор строк, отображаемых в компоненте, определяет количество элементов списка и их содержимое
Sorted	Разрешает (True) или отменяет (False) сортировку строк списка в алфавитном порядке

Таблица П. 4.7

Компоненты для реализации стандартных диалогов
(страница Dialogs, компоненты невидимы во время выполнения,
поэтому место их размещения на форме не имеет значения)

Пиктограмма	Компонент	Назначение: создание окна диалога
	OpenDialog	Открыть файл
	SaveDialog	Сохранить файл
	OpenPictureDialog	Открыть рисунок
	SavePictureDialog	Сохранить рисунок
	FontDialog	Шрифт – выбор атрибутов шрифта
	ColorDialog	Цвет – выбор цвета
	PrintDialog	Печать
	PrintSetupDialog	Установка принтера
	FindDialog	Найти – контекстный поиск в тексте
	ReplaseDialog	Заменить – контекстная замена фрагментов текста
	PageSetupDialog	Параметры страницы

Библиотека БГУИР

Таблица П. 4.8

Свойства компонентов OpenDialog и SaveDialog

Свойства	Тип	Описание
DefaultExt	String	Задает расширение, автоматически подставляемое к имени файла, если не указано расширение
FileName	String	Указывает имя и полный путь файла, выбранного в диалоге
Filter	String	Задает маску имени файлов
FilterIndex	Integer	Указывает маску фильтра, отображаемую в списке
InitialDir	Integer	Определяет каталог, содержимое которого будет отображаться при вызове окна
Options	TOpenOptions	Применяется для настройки параметров, управляющих внешним видом и функциональными возможностями диалога
ofOverwritePrompt		Предупреждает, что файл уже существует и требует подтверждения
ofNoChangeDir		Вызывает текущий каталог при открытии
ofAllowMultiSelect		Разрешается одновременный выбор из списка нескольких файлов
ofPathMustExist		Задаются файлы только из существующих каталогов
ofFileMustExist		Задаются только существующие файлы
ofCreatePrompt		Формируется запрос на создание файла при вводе несуществующего имени файла
Title	String	Задает заголовок окна

Библиотека БГУИР

ЛИТЕРАТУРА

1. Колосов, С. В. Программирование в среде Delphi : учеб. пособие по курсу «Программирование» / С. В. Колосов. – Минск : БГУИР, 2005.
2. Фаронов, В. В. Delphi. Программирование на языке высокого уровня / В. В. Фаронов. – СПб. : Питер, 2009.
3. Архангельский, А. Я. DELPHI 2006 / А. Я. Архангельский. – М. : Бином – Пресс, 2010.
4. Культин, Н. Б. Delphi в задачах и примерах / Н. Б. Культин. – 2-е изд. – СПб. : BHV – СПб., 2008.
5. Окулов, С. М. Программирование в алгоритмах / С. М. Окулов. – М. : Бином. Лаборатория знаний, 2006.
6. Бакнелл, Дж. Фундаментальные алгоритмы и структуры данных в Delphi. Библиотека программиста / Дж. Бакнелл. – М. : ООО «ДиаСофтЮт»; СПб. : Питер, 2006.
7. Долинский, М. С. Решение сложных и олимпиадных задач по программированию. / М. С. Долинский. – СПб. : Питер, 2006.
8. Окулов, С. М. Основы программирования / С. М. Окулов. – Бином. Лаборатория знаний, 2008.
9. Семакин, И. Г. Основы алгоритмизации и программирования / И. Г. Семакин, А. П. Шестаков. – М. : Академия, 2008.
10. Чеснокова, О. В. Delphi 2007. Алгоритмы и программы / О. В. Чеснокова. – М. : НТ Пресс, 2008.
11. Потапов, В. И. Основы компьютерной арифметики и логики: учеб. пособие / В. И. Потапов, О. П. Шафеева, И. В. Червенчук. – Омск : изд-во ОмГТУ, 2004.

Библиотека БГУИР

Св. план 2012, поз. 25

Учебное издание

Коренская Ирина Николаевна
Лущицкая Ирина Владимировна

ОСНОВЫ ПРОГРАММИРОВАНИЕ В СРЕДЕ DELPHI. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Редактор *И. В. Ничипор*
Корректор *Е. Н. Батурчик*
Компьютерная правка, оригинал-макет *А. А. Лысеня*

Подписано в печать 18.04.2013. Формат 68x84 1/16. Бумага офсетная. Гарнитура «Таймс».
Отпечатано на ризографе. Усл. печ. л. Уч.-изд. л. 8,0. Тираж 100 экз. Заказ 212.

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0494371 от 16.03.2009. ЛП № 02330/0494175 от 03.04.2009.
220013, Минск, П. Бровки, 6

Библиотека БГУИР