Федеральное агентство по образованию

Государственное образовательное учреждение
высшего профессионального образования
«Ивановская государственная текстильная академия»
(ИГТА)

Кафедра прикладной математики и информационных технологий

Введение в C/C++

Учебное пособие
по дисциплинам
«Прикладное программирование в информационных системах», «Программная реализация средств дизайна»
для студентов дневной формы обучения
специальности 740100
«Информационные технологии в дизайне»

Иваново 2010

Учебное пособие предназначено для студентов, изучающих основы программирования в C/C++ системе. Данное учебное пособие направлено на необходимость приобретения студентами некоторых подходов к решению задач разработки программного обеспечения в C/C++ системе.

Данное учебное пособие является введением в изучение концепций программирования C/C++ системе и некоторых подходов к программной реализации средств дизайна средствами этой системы.

Составители:

Доцент кафедры ПМИТ, к.т.н., доцент Д. Д. Ветчинин

Научный редактор:

Зав. кафедрой ПМИТ д.т.н., профессор Н. А. Коробов

1. Основные элементы C/C++

```
1   /* Программа 1.1. на "классическом" C :
2     сложение двух целых чисел */
3
4   #include <stdio.h>
5
6   int main()
7   {
8           int integer1;
9           int integer2;
10          int sum;
11
12          printf( "Enter first integer\n" );
13          scanf( "%d", &integer1 );
14
15          printf( "Enter second integer\n" );
16          scanf( "%d", &integer2 );
17
18          sum = integer1 + integer2;
19
20          printf( "Sum is %d\n", sum );
21
22          return 0;
23  }
```

```
1   // Программа 1.2. на C++ - аналог программы 1.1. :
2   //   сложение двух целых чисел
3
4   #include "iostream"
5
6   int main()
7   {
8           int integer1;      //объявление переменной
9
10          std::cout << "Enter first integer\n";
11          std::cin >> integer1;
12
13          int integer2, sum; //объявление остальных переменных
14
15          std::cout << "Enter second integer\n";
16          std::cin >> integer2;
17
18          sum = integer1 + integer2;
19          std::cout << "Sum is " << sum <<std::endl;
20
21          return 0;
22  }
```

```
1   /* Программа 1.3. на "классическом С":
2      создание и использование функции, определяемой программистом */
3   #include "stdio.h"
4
5   int square( int y ); /* прототип функции */
6
7   int main()
8   {
9           int x; /* объявление переменной - счётчика */
10
11          /* 10 раз вычислить м вывести квадрат x */
12          for ( x = 1; x <= 10; x++ ) {
13                  printf( "%d  ", square( x ) );
14          } /* конец цикла */
15
16          printf( "\n" ); /* перевод строки */
17
18          return 0; /* отображает успешное завершение программы */
19  }
20
21  /* определение функции square, возвращающей квадрат её входного
22  параметра */
23  int square( int y ) /* y является копией аргумента функции */
24  {
25          return y * y; /* возвращает квадрат y как int */
26  } /* конец функции square */
```

```
1   // Программа 1.4. на C++ - аналог программы 1.3. :
2   // создание и использование функции, определяемой программистом
3
4   #include "iostream"
5
6   //  оператор using – последующее сокращение полного синтаксиса при
7   //  обращении к пространству имён
8
9           using std::cout;
10
11  // определение функции square, возвращающей квадрат её входного параметра
12  // организуется как inline – функция (встроенная функция)
13
14          inline int square( const int y ) { return y * y; }
15
16  int main()
17  {
18          for ( int x = 1; x <= 10; x++ ) {
19                  cout << " " << square( x ) << " ";
20          }
21          cout << std::endl; // перевод строки
22
23          return 0;
24  }
```

```
1   // Прогр.1.5.-Сравнение вызовов ф-ии по значению и по ссылке
2   // с параметром - сссылкой
3   #include <iostream>
4
5           using std::cout; using std::endl;
6
7           int squareByVal( int );
8           void squareByRef( int & );
9
10  void main()
11  {
12          int x = 2, y = 4;
13
14          cout << "x = " << x << " befor call squareByVal\n"
15                  << "Value returned by squareByVal: "
16                  << squareByVal( x ) << endl
17                  << "x = " << x << " after call squareByVal\n" << endl;
18
19          cout << "y = " << y << " befor call squareByRef" << endl;
20          squareByRef( y );
21          cout << "y = " << y << " after call squareByRef" << endl;
22  }
23
24  int squareByVal( int a ) { return a *= a; }
25
26  void squareByRef( int &cRef ) { cRef *= cRef; }
```

```
1   // Прогр.1.6.- Инициализация ссылок обязательна
2   // Объявление ссылки как псевдонима переменной
3   #include <iostream>
4
5           using std::cout; using std::endl;
6
7   void main()
8   {
9           int x = 3, &y = x;
10
11          cout << "x = " << x << endl << "y = " << y << endl;
12          y = 7;
13          cout << "x = " << x << endl << "y = " << y << endl;
14  }
```

```
1   // Прогр.1.7.- Использование аргументов ф-ии по умолчанию
2   // и пустых списков параметров
3   #include <iostream>
4
5           using std::cout; using std::endl;
6
7           int BoxSize(int length=1, int width=1, int height=1);
8
```

```
9   void main()
10  {
11
12          cout << "The default box volume is: " << BoxSize()
13                  << "\n\nThe volume of a box with length 11,\n"
14                  << "width 1 and height 1 is: " << BoxSize(11)
15                  << "\n\nThe volume of a box with length 11,\n"
16                  << "width 7 and height 1 is: " << BoxSize(11,7)
17                  << "\n\nThe volume of a box with length 11,\n"
18                  << "width 7 and height 2 is: " << BoxSize(11,7,2) << endl;
19  }
20
21  int BoxSize(int length, int width, int height)
22  {
23          return length * width * height;
24  }
```

```
1   // Прогр.1.8.- Пример использования унарной операции разрешения
2   // области действия к глобальной переменной
3   #include <iostream>
4           using std::cout; using std::endl; using std::ios;
5
6   #include <iomanip>
7           using std::setprecision; using std::setiosflags; using std::setw;
8
9   const double PI = 3.14159265358979;
10
11  void main()
12  {
13          const float PI = static_cast< float >( ::PI );
14
15          cout << setprecision(20)
16                  << "  Local float value of PI = " << PI
17                  << "\nGlobal double value of PI = " << ::PI << endl;
18
19          cout << setw(28) << "Local float value of PI = "
20                  << setiosflags(ios::fixed | ios::showpoint)
21                  << setprecision(10) << PI << endl;
22  }
```

```
1   // Прогр.1.9.- Перегрузка функций (различие сигнатур)
2   // Выполнение аналогичных задач перегруженными функциями
3   #include <iostream>
4           using std::cout; using std::endl;
5
6           int square( int x ) { return x * x; }
7
8           double square( double y ) { return y * y; }
9
10  void main()
```

```
{
        cout << "The square of integer 5 is " << square( 5 )
                << "\nThe square of double 5.25 is " << square( 5.25 )
                << endl;
}
```

```
// Прогр.1.10.- Учебная демонстрация использования
// шаблона функции через вызовы шаблона
#include <iostream>
using std::cout; using std::cin; using std::endl;

template < class X >
X maximum( X value1, X value2, X value3 )
{
        X max = value1;
        if ( value2 > max )
                max = value2;
        if ( value3 > max )
                max = value3;
        return max;
}

void main()
{
        int int1, int2, int3;

        cout << "\nInput three integer values: ";
        cin >> int1 >> int2 >> int3;
        cout << "The maximum integer value is: "
                << maximum( int1, int2, int3 ); // версия для целых

        double double1, double2, double3;

        cout << "\nInput three double values: ";
        cin >> double1 >> double2 >> double3;
        cout << "The maximum double value is: "
                << maximum( double1, double2, double3 ); // версия для double

        char char1, char2, char3;

        cout << "\nInput three characters: ";
        cin >> char1 >> char2 >> char3;
        cout << "The maximum characters value is: "
                << maximum( char1, char2, char3 ) // версия для литералов
                << endl;
}
```

2. Массивы

```
1   //Прогр. 2.1
2   //Инициализация массива - 1-ый пример
3   #include <iostream>
4
5   void main()
6   {
7           int n[10];
8           int i;
9
10          //Инициализация массива n нулями
11          for(i = 0; i < 10; i++) {
12                  n[i] = 0;          }
13
14          using std::cout;
15          using std::ios;
16          using std::endl;
17
18          cout << "Element   " << "Value" << endl;
19
20          //Вывод содержимого массива
21
22          cout.setf(ios::right);
23
24          for (i = 0; i < 10; i++)  {
25                  cout.width(7);
26                  cout << i;
27                  cout.width(8);
28                  cout << n[i] << endl; }
29  }
```

```
1   // Прогр. 2.2 - Инициализация массива
2   // с помощью списка инициализации (2-ой пример)
3   #include <iostream>
4
5   void main()
6   {
7           int n[10] = {33,21,15,192,17,51,44,11,45,13};
8           int i;
9
10          using std::cout; using std::ios; using std::endl;
11
12          cout << "Element   " << "Value" << endl;
13
14          // Вывод содержимого массива
15          cout.setf(ios::right);
16
17          for (i = 0; i < 10; i++)  {
18                  cout.width(7);
19                  cout << i;
```

```
20                      cout.width(8);
21                      cout << n[i] << endl; }
22  }
```

```
1   //Прогр. 2.3
2   //Спецификация размера массива символической константой
3   //и его инициализация вычисляемыми значениями
4   #include <iostream>
5   #define SIZE 10
6
7   void main()
8   {
9           int s[SIZE];
10          int j;
11
12          for(j =0; j < SIZE; j++) {
13                  s[j] = 2 + 2 * j;   }
14
15          using std::cout; using std::ios; using std::endl;
16
17          cout << "Element   " << "Value" << endl;
18
19          cout.setf(ios::right);
20
21          for (j = 0; j < SIZE; j++)  {
22                  cout.width(7);
23                  cout << j;
24                  cout.width(8);
25                  cout << s[j] << endl; }
26  }
```

```
1   //Прогр. 2.4
2   //Вычисление суммы элементов массива
3   #include <iostream>
4   #define SIZE 12
5
6   void main()
7   {
8           int a[SIZE] = {1,3,5,9,91,7,33,51,4,18,0,21};
9           int i;
10          int sum = 0;
11
12          //суммирование содержимого массива a
13          for (i = 0; i < SIZE; i++) {
14                  sum += a[i];        }
15
16          std::cout << "Sum of array elements is " << sum << std::endl;
17  }
```

```
1   //Прогр. 2.5
2   //Опрос студентов по качеству преподавателя
3   #include <iostream>
```

```
4   #define RESP_SISE 29
5   #define FREQ_SISE 11
6
7   void main()
8   {
9           int answer; //счётчик для перебора откликов
10          int rating;//счётчик для перебора частот 1 - 10
11          int i;
12
13          //инициализировать счётчики частот нулями
14          int frecuency[FREQ_SISE] = {0};
15
16          //поместить ответы в массив откликов
17          int responses[RESP_SISE] = {3,5,10,1,4,5,3,6,2,3,4,5,7,8,9,10,2,
18          3,2,7,8,4,6,9,3,5,4,2,1};
19
20          for (answer = 0; answer < RESP_SISE; answer++) {
21                  ++frecuency[responses[answer]]; }
22
23          using std::cout;
24
25          cout << "Rating   " << "Frequency   " <<
26                  "Histogram" << std::endl;
27
28          cout.setf(std::ios::right);
29
30          for (rating = 1; rating < FREQ_SISE; rating++)  {
31                  cout.width(6);
32                  cout << rating;
33                  cout.width(12);
34                  cout << frecuency[rating];
35
36                  cout.width(12);
37
38                  for (i = 1; i <= frecuency[rating]; i++)  {
39                          cout << "*"; }     //cout.put('*');
40                  cout.put('\n');
41          }
42  }
```

```
1   //Прогр. 2.6
2   //Функции и локальные статический и автоматический массивы
3   #include <iostream>
4
5   void staticArrayInit(void);
6   void automaticArrayInit(void);
7
8   void main()
9   {
10          using std::cout; using std::endl;
11
12          cout << "First call to each function" << endl;
13          staticArrayInit();
14          automaticArrayInit();
15
16          cout.put('\n') << "Second call to each function" << endl;
17          staticArrayInit();
18          automaticArrayInit();
19  }
20
21
22  //функция, демонстрирующая локальный статический массив
23  void staticArrayInit(void)
24  {
25          //при первом вызове элементы инициализируются нулями
26          using std::cout; using std::endl;
27          static int array1[3];
28          int i;
29
30          cout.put('\n\n');  //cout << endl;
31          cout << "Values on enterning staticArrayInit" << endl;
32
33          //вывод содержимого array1
34          for (i = 0; i<= 2; i++)  {
35                  cout << "array1[" << i << "] =  " << array1[i] << "  ";  }
36
37          cout << endl;  cout << endl;
38          cout << "Values on exiting staticArrayInit" << endl;
39
40          //модифицировать и вывести содержимое array1
41          for (i = 0; i<= 2; i++)  {
42                  cout << "array1[" << i << "] =  " << (array1[i] += 5) << "  ";  }
43  }
44
45  //функция, демонстрирующая локальный автоматический массив
46  void automaticArrayInit(void)
47  {
48          //инициализация элементов при каждом вызове функции
49          using std::cout; using std::endl;
50          int array2[3] = {1,2,3};
51          int i;
```

```
52
53              cout << endl;  cout << endl;
54              cout << "Values on enterning automaticArrayInit:" << endl;
55
56              //вывод содержимого array2
57              for (i = 0; i<= 2; i++)  {
58                      cout << "array2[" << i << "] =  " << array2[i] << "  ";  }
59
60              cout << endl;  cout << endl;
61              cout << "Values on exiting automaticArrayInit:" << endl;
62
63              //модифицировать и вывести содержимое array1
64              for (i = 0; i<= 2; i++)  {
65                      cout << "array2[" << i << "] =  " << (array2[i] += 5) << "  ";  }
66      cout << endl;
67      }
```

```
1       //Прогр. 2.7
2       //Массивы символов в качестве строк
3       #include <iostream>
4
5       void main()
6       {
7               char string1[20];
8               char string2[] = "string literal";
9               int i;
10
11              using std::cout; using std::endl; using std::cin;
12
13              cout << "Enter a string \n"; //Введите фразу, напр. "Hello Dolly"
14              cin >> string1;
15
16              cout << "string1 is: " << string1 << endl;
17              cout << "string2 is: " << string2 << endl;
18              cout << "string1 with spaces between characters is: \n";
19
20              for(i = 0; string1[i] != '\0'; i++) {
21                      cout << string1[i];  }
22              cout << endl;
23
24              cout << "Output all characters string1, test 1: \n";
25              cout.write( string1, 20 );
26              cout << endl;
27
28              cout << "Output all characters string1, test 2: \n";
29              for(i = 0; i < 20; i++) {
30                      cout << string1[i];  }
31              cout << endl;
32
33              cout << "Output all characters string1, test 3: \n";
34              for(i = 0; i < 20; i++) {
```

```
35                      cout.put(string1[i]);  }
36              cout << endl;
37
38              cout << "Output all characters string1, test 4: \n";
39              cout << "Enter a sentence: \n";
40              cin.getline(string1, 20);
41              cout << string1 << endl;
42
43              cout << "Output all characters string1, test 5: \n";
44              char s;
45              cout << "Enter a sentence: \n";
46              while ( (s = cin.get() ) != EOF )
47                      cout.put(s);
48              cout << s << endl;
49
50              cout << endl;
51      }
```

```
1       /*Прогр. 2.8
2       Адресация массива (имя массива эквивалентно адресу
3       его первого элемента)*/
4       #include <stdio.h>
5
6       int main()
7       {
8               char array[5];
9
10              printf("    array = %p\n&array[0] = %p\n"
11                      "   &array = %p\n",
12                      array, &array[0], &array);
13
14      return 0;
15      }
```

```
1       //Прогр. 2.9
2       //Передача функциям массивов или их отдельных элементов
3       #include <iostream>
4       #define SIZE 5
5
6       void modifyArray(int b[], int size);
7       void modifyElement(int e);
8
9       void main()
10      {
11              int a[SIZE] = {0,1,2,3,4};
12              int i;
13
14              using std::cout; using std::endl; using std::cin;
15
16              cout << "Effects of passing entire array call by "
```

```
17                          "reference:\n";  cout << endl;
18          cout << "The values of the original array are:\n";
19
20          //Вывести исходный массив
21          for(i = 0; i < SIZE; i++) {
22                  cout.width(3); cout << a[i]; } cout << endl;
23
24          //Передать массив modifyArray по ссылке
25          modifyArray(a, SIZE);
26
27          cout << "The values of the modified array are: \n";
28
29          //Вывести модифицированный массив
30          for(i = 0; i < SIZE; i++) {
31                  cout.width(3); cout << a[i]; } cout << endl;
32
33          //Вывести значения отдельных элементов масива
34                          cout << "\n\n\nEffects of passing array element "
35                          "call by value:\n\n";
36          //Вывести значение a[3]
37                  cout << "The value of a[3] is " <<  a[3] << endl;
38
39          modifyElement(a[3]); //Передать элемент a[3] по значению
40
41          cout << "The value of a[3] is " <<  a[3] << endl;
42  }
43
44
45  //в функции modifyArray "b" указывает на
46  //исходный массив "a" в памяти
47  void modifyArray(int b[], int size)
48  {
49          int j;
50
51          //умножить каждый элемент массива на 2
52          for (j = 0; j < size; j++) {
53                  b[j] *= 2;  }
54  }
55
56
57  // в функции modifyElement "e" является локальной копией
58  //элемента массива a[3], переданного из main
59  void modifyElement(int e)
60  {
61          //умножить параметр на 2
62          std::cout << "Value in modifyElement is " << (e *= 2)
63                  << std::endl;
64  }
```

3. Адресация

```
1   //Прогр. 3.1.
2   //Указатель - адрес, ковенная адресация (разыменование),
3   //операции над указателями
4   #include <iostream>
5
6   using std::cout; using std::endl;
7
8   void main()
9   {
10          int i; int *ptr_i;
11
12          i= 3;
13          ptr_i=&i; //pct_i установлен на адрес i
14
15          cout<<"The value of ptr_i is " << ptr_i << std::endl;
16          cout<<"The address of i is " << &i << std::endl;
17
18          cout<<"\nThe value of i is " << i << std::endl;
19          cout<<"The value of *ptr_i is " << *ptr_i << std::endl;
20
21          cout<<"\nShowing that * and & are complements of each other"
22                  <<"\n&*ptr_i = " << &*ptr_i
23                  <<"\n*&ptr_i = " << *&ptr_i << endl;
24  }
```

```
1   //Прогр. 3.2
2   //Адресация памяти
3   #include <iostream>
4
5   void main()
6   {
7           int n[10] = {33,21,15,192,17,51,44,11,45,13};
8           int i; int *pct_i;
9
10          using std::cout; using std::ios; using std::endl;
11
12          cout << "Element   " << "Value      " << "Adress  " << endl;
13
14          //Вывод содержимого массива
15          cout.setf(ios::right);
16
17          for (i = 0; i < 10; i++)  {
18
19                  pct_i=&n[i];
20
21                  cout.width(7);
22                  cout << i;
23                  cout.width(8);
24                  cout << n[i];
25                  cout.width(12);
```

```
26                      cout << pct_i << endl; }
27  }
```

```
1   //Прогр. 3.3.
2   //Возведение в квадрат переменной, передаваемой по значению
3   //Передача переменной в функцию по значению
4   #include <iostream>
5
6   int sqrByVal( int n );
7
8   using std::cout; using std::endl;
9   void main()
10  {
11          int num = 7;
12
13          cout<<"The original value of number is " << num << std::endl;
14
15          /* Передать число sqrByVal по значению */
16          num = sqrByVal( num );
17
18          cout<<"The new value of number is " << num << std::endl;
19  }
20
21  //вычислить и возвратить значение функции
22  int sqrByVal( int n )
23  {
24          return n * n; // возвратить квадрат локальной переменной n
25  }
```

```
1   /* Прогр. 3.4. -  Возведение в квадрат переменной, передаваемой
2   аргументом-указателем. Передача переменной в функцию по ссылке
3   (имитация вызова по ссылке, т.е. при передаче адреса переменной ф-ии
4   для изменения значения переменной используется операция
5   косвенной адресации (*)) */
6   #include <iostream>
7
8   void squareByRef( int *ptr_n );
9
10  using std::cout; using std::endl;
11
12  void main()
13  {
14          int num = 7;
15
16          cout<<"The original value of number is " << num << std::endl;
17
18          // Передать функции sqrByVal адрес числа - операнда
19          squareByRef( &num );
20
21          cout<<"The new value of number is " << num << std::endl;
```

```
}

//вычислить функцию, модифицировать переменную num в main
void squareByRef( int *ptr_n )
{
        *ptr_n = *ptr_n * *ptr_n; // возвести в квадрат *ptr_n
}
```

```
//Прогр. 3.5.
//Сортировка значений массива в восходящем порядке
#include <iostream>
#define SIZE 10

void main()
{
        int a[SIZE] = {1,99,5,34,67,7,3,41,37};
        int pass; //счётчик проходов
        int i; // счётчик сравнений
        int change; //ячейка для обмена элементов массива
        using std::cout; using std::ios; using std::endl;

        cout << "Data items in original order\n";
        for (i = 0; i < SIZE; i++) {
                cout.width(4);
                cout << a[i];        }

        //сортировка погружением (пузырьковая сортировка)

        for (pass = 1; pass < SIZE; pass++) { //цикл числа проходов

        //цикл контроля числа сравнений на данном проходе
                for (i = 0; i < SIZE - 1; i++) {
                        //сравнить соседние элементы и обменять их
                        //если первый элемент больше второго
                        if (a[i] > a[i + 1]) {
                                change = a[i];
                                a[i] = a[i+1];
                                a[i + 1] = change; } //конец if
                } //конец внутреннего for
        } // конец внешнего for

        cout << "\nData items in ascending order\n";

        for (i = 0; i < SIZE; i++) {
                cout.width(4);
                cout << a[i]; } cout << endl;
}
```

```
//Прогр. 3.6.
//Пример двухмерного массива
#include <iostream>
#define STUDENTS 3
```

```
#define EXAMS 4
int min(const int grades[][EXAMS], int pupils, int tests);
int max(const int grades[][EXAMS], int pupils, int tests);
double average(const int setOfGrades[], int tests);
void printArray(const int grades[][EXAMS], int pupils, int tests);
using std::cout; using std::endl; using std::ios;

void main()
{
        int student; //счётчик студентов
        //инициализировать массив оценок
        const int studentGrades[STUDENTS][EXAMS] =
        {   {71,69,85,72},
                {95,86,89,74},
                {73,91,87,82} };


        cout << "The array is:\n"; // вывести массив оценок
        printArray(studentGrades, STUDENTS, EXAMS);

        //определить - вывести наибольшую и наименьшую оценку
        cout << "\n\nLowest grade: ";
   cout << min(studentGrades, STUDENTS, EXAMS) << endl;
        cout << "Highest grade: ";
        cout << max(studentGrades, STUDENTS, EXAMS) << endl;

        //вычислить - вывести среднюю оценку для каждого студента

        for (student = 0; student < STUDENTS; student++) {
                cout << "The average grade for student ";
                cout << student << " is ";
                cout <<         average(studentGrades[student], EXAMS) << endl;
        }
}


 int min(const int grades[][EXAMS], int pupils, int tests)
 {
          int i; //счётчик студентов
          int j; //счётчик экзаменов
          int lowGrade = 100; //инициализировать максимумом

          for(i = 0; i < pupils; i++) { //цикл по строкам оценок
                  for(j = 0; j < tests; j++) { //цикл по столбцам оценок

                          if (grades[i][j] < lowGrade) {
                                  lowGrade = grades[i][j]; }
                  }
          }
 return lowGrade; }
```

```
58      int max(const int grades[][EXAMS], int pupils, int tests)
59      {
60              int i; //счётчик студентов
61              int j; //счётчик экзаменов
62              int highGrade = 0; //инициализировать минимумом
63
64              for(i = 0; i < pupils; i++) { //цикл по строкам оценок
65                      for(j = 0; j < tests; j++) { //цикл по столбцам оценок
66
67                              if (grades[i][j] > highGrade) {
68                                      highGrade = grades[i][j]; }
69                      }
70              }
71      return highGrade; }
72
73
74      //Определение средней оценки студента
75      double average(const int setOfGrades[], int tests)
76      {
77              int i; //счётчик экзаменов
78              int total = 0; //сумма оценок за экзамены
79
80              for(i = 0; i < tests; i++) { //суммирование оценок студента
81                      total += setOfGrades[i]; }
82       return (double) total/tests;
83      }
84
85
86      void printArray(const int grades[][EXAMS], int pupils, int tests)
87      {
88              int i; //счётчик студентов
89              int j; //счётчик экзаменов
90
91              cout.setf(ios::right);
92              cout.width(24); cout << "[0]";
93              cout << "    [1]" << "    [2]" << "    [3]" << endl;
94
95              //вывести оценки в формате таблицы
96
97              for(i = 0; i < pupils; i++) {
98                      cout << "\nstudentGrades" << '[' << i << ']';
99
100                     //вывести оценки одного студента
101                     for(j = 0; j < tests; j++) {
102                             cout.width(7);
103                             cout << grades[i][j];   }
104             }
105     }
```

```
1   /* Прогр. 3.7. - Преобразование букв нижнего регистра в верхний
2   с помощью не-константного указателя на не-константные данные */
3   #include <iostream>
```

```
#include <cctype> //ctype.h

void convertUp( char *ptr_string );

using std::cout; using std::endl;

void main()
{
        char string[] = "visual studio .net and $33.71"; //массив символов

        cout<<"The string befor conversion is " << string << endl;

        convertUp( string );

        cout<<"The string after conversion is " << string << endl;
}

//функция преобразования символов строки в символы верхнего регистра
void convertUp( char *ptr_string )
{
        while ( *ptr_string != '\0' ) { // текущий символ не '\0'
                //если символ в нижнем регистре, то
                  if ( islower(*ptr_string) ) { // преобразовать в верхний
                          *ptr_string = toupper(*ptr_string); }

                  ++ptr_string; } //переместить ptr_string на следующий символ
}
```

```
/* Прогр. 3.8. - Печать строки по одному символу
с помощью не-константного указателя на константные данные */
#include <iostream>

void printChar( const char *ptr_string );
void addressBy_ptr( char *ptr_string ); /* на отсутствие декларации
функции в прототипе и объявление её аргумента как const при
попытке его изменения "номальная система" возмущена!!! */

using std::cout; using std::ios; using std::endl;

void main()
{
        char string[] = "Visual Studio 6.5"; //массив символов

        cout << "The string is " << string << endl;

        printChar( string ); cout << endl;

        addressBy_ptr( string ); cout << endl;
}

//функция вывода символов строки в поток, указатель ptr_string
//не может модифицировать символ, на который указывает
//т.е. ptr_string - указатель "только для чтения"
```

```
26
27    void printChar( const char *ptr_string )
28    {
29            for ( ; *ptr_string != '\0'; ptr_string++ ) { // текущий символ не '\0'
30
31                    cout << *ptr_string; }
32
33            cout << endl;
34    }
35
36    void addressBy_ptr( char *ptr_string )
37    {
38            int i =0;
39
40            cout << "Index" << "\tAddress" << "\t  Value" << "\t Byte_Ptr"
41                    << "  Byte_Value\n\n";
42
43                    cout.setf(ios::right);
44            for ( ; *ptr_string != '\0'; ptr_string++ ) {
45
46                    cout.width(5);
47                    cout << i;
48
49                    cout.width(10);
50                    cout << static_cast< void * >(ptr_string);
51    //указатель void * не может быть разыменован !!!?
52                    cout.width(8);
53                    cout << *ptr_string;
54                    cout.width(10);
55                    cout << sizeof ptr_string;
56
57                    cout.width(12);
58                    cout << sizeof *ptr_string;
59
60                    ++i; cout << endl;
61            }
62    }
```

```
1   //Прогр. 3.9. - Аналог прогр. 3.5
2   /* Программа сортировки значений массива в восходящем порядке
3     (пузырьковая сортировка), использующая вызов по ссылке */
4   #include <iostream>
5   #define SIZE 10
6
7   void sort( int* const array, const int size );
8
9   void main()
10  {
11          int a[SIZE] = {1,99,5,34,67,7,3,41,37,25};
12
13          int i; // счётчик
```

```
14
15          using std::cout; using std::ios; using std::endl;
16
17          cout << "Data items in original order\n";
18          // цикл по массиву a
19          for (i = 0; i < SIZE; i++) {
20                  cout.width(4);
21                  cout << a[i];       }
22
23          sort( a, SIZE); // сортировать массив
24
25          cout << "\nData items in ascending order\n";
26          // цикл по массиву a
27          for (i = 0; i < SIZE; i++) {
28                  cout.width(4);
29                  cout << a[i]; } cout << endl;
30  } // конец main
31
32  //сортировка массива целых методом погружения
33  void sort( int* const array, const int size )
34  {
35          void change( int *ptr_element1, int *ptr_element2 );
36          int pass; // счётчик числа походов
37          int j; // счётчик числа сравнений
38
39          //цикл для контроля числа проходов
40          for (pass = 0; pass < size - 1; pass++) {
41
42                  //цикл контроля сравнений на данном проходе
43                  for (j = 0; j < size - 1; j++) {
44                          //сравнить соседние элементы и обменять их
45                          //если первый элемент больше второго
46                          if ( array[ j ] > array[ j + 1 ] ) {
47                                  change( &array[ j ], &array[ j + 1 ] );
48                          } //конец if
49                  } //конец внутреннего for
50          } // конец внешнего for
51  } // конец функции sort
52  /* Определение функции change, обменивающей значения в ячейках,
53     на которые указывают указатели ptr_element1 и ptr_element2 */
54
55  void change( int *ptr_element1, int *ptr_element2 )
56  {
57          int swap = *ptr_element1;
58          *ptr_element1 = *ptr_element2;
59          *ptr_element2 = swap;
60  } // конец функции change
```

```
1   //Прогр. 3.10. Пример применения унарной операции sizeof, применяемой
2   /* для определения размера в байтах любого типа данных во время
3   компиляции программы (затраты времени выполнения отсутствуют) */
```

```
#include <iostream>

size_t Size( double *ptr );

void main()
{
        double array[ 8 ];

        using std::cout; using std::ios; using std::endl;

        cout << "The number of bytes in the array is "
                << sizeof( array ) << endl;

   cout << "The number of bytes returned by Size is "
                << Size( array ) << endl;
} // конец main


// возвратить размер ptr
size_t Size( double *ptr )
{
        return sizeof( ptr );
}
```

```
//Прогр. 3.11. - Демонстрация применения унарной операции sizeof

#include <iostream>

void main()
{
        char c;
        short s;
        int i;
        long l;
        float f;
        double d;
        long double ld;
        int array[ 20 ];
        int *ptr = array; // создать указатель на массив

        using std::cout; using std::endl;

        cout << " sizeof c = " << sizeof c << "\tsizeof(char) = " << sizeof(char)
        << "\n  sizeof s = " << sizeof s << "\tsizeof(short) = " << sizeof(short)
        << "\n  sizeof i = " << sizeof i << "\tsizeof(int) = " << sizeof(int)
        << "\n  sizeof f = " << sizeof f << "\tsizeof(float) = " << sizeof(float)
        <<"\n sizeof d = " <<sizeof d<<"\tsizeof(double) = " << sizeof(double)
        << "\n   sizeof ld = " << sizeof ld << "\tsizeof(long double) = "
        << sizeof(long double)
                << "\nsizeof array = " << sizeof array
                << "\n sizeof ptr = " << sizeof ptr << endl;
```

```
28  }


1   //Прогр. 3.12. - Демонстрация манипуляции элементами массива через
2   //              использование нотаций индексов и указателей
3   /* Используются четыре метода ссылки на элементы массива:
4     - индексация массива,
5     - указатель/смещение с именем массива в качестве указателя,
6     - индексация указателя,
7     - указатель/смещение с указателем.
8   !!! - имя массива - указатель-константа, всегда указывающий на начало
9   массива. Попытка изменения этого указателя - ошибка синтаксиса!!! */
10  #include <iostream>
11
12  void main()
13  {
14          int at[] = { 11, 22, 33, 44 };
15          int *ptr_at = at; // усстановить *ptr_at = at на массив at
16          int i; // счётчик
17          int off; // счётчик
18
19          using std::cout; using std::endl;
20
21          // вывод значений элементов массива с использованием нотации индексации
22          cout << "Array at printed with:" << "\nArray subscript notation\n";
23
24          // цикл перебора элементов массива at
25          for ( i = 0; i < 4; i++ ) {
26                  cout << "at" << '[' << i << ']' << " = " << at[ i ] << endl; }
27
28          //  вывод значений элементов массива с использованием имени и
29          //  нотации указатель/смещение
30          cout << "\nPointer/offset notation where\n"
31                  << "the pointer is the array name\n";
32
33          // цикл перебора элементов массива at
34          for ( off = 0; off < 4; off++ ) {
35                  cout << "*( at + " << off << ") = " << *( at + off )
36                      << endl; }
37
38          // вывод массива с использованием указателя ptr_at и нотации индексации
39          cout << "\nPointer subscript notation\n";
40
41          // цикл перебора элементов массива at
42          for ( i = 0; i < 4; i++ ) {
43                  cout << "ptr_at" << '[' << i << ']' << " = " << ptr_at[ i ]
44                          << endl; }
45
46  // вывод массива с использованием указателя ptr_at и нотации указатель/смещение
47          cout << "\nPointer/offset notation\n";
48
49          // цикл перебора элементов массива at
```

```
50          for ( off = 0; off < 4; off++ ) {
51                  cout << "*(ptr_ at + " << off << ") = " << *( ptr_at + off )
52                          << endl; }
53  }
```

```
1   //Прогр. 3.13. - Копирование строк с использованием
2   //арифметических операций над индексами и указателями
3   #include <iostream>
4   //undeclared identifier(error C2065) - если нет прототипа
5   void copy_1( char *ptr_1, const char *ptr_2 );
6   void copy_2( char *ptr_1, const char *ptr_2 );
7   void copy_3( char *ptr_1, const char *ptr_2 );
8
9   void main()
10  {
11          using std::cout; using std::ios; using std::endl;
12          char string_1[ 10 ];   // создать массив
13          char *string_2 = "Per Aspera";   // создать указатель на строку
14          char string_3[ 10 ];   // создать массив
15          char string_4[] = "Ad Astra";  // создать указатель на строку
16
17          copy_1( string_1, string_2 );
18          cout << string_1 << endl;
19          copy_2( string_3, string_4 );
20          cout << string_3 << endl;
21  //      copy_3( string_3, string_4 );
22  }
23
24  // копировать ptr_2 в ptr_1, используя нотацию массивов
25  void copy_1( char *ptr_1, const char *ptr_2 )
26  {
27          int i;
28          for (i = 0; ( ptr_1[ i ] = ptr_2[ i ] ) != '\0' ; i++)
29                  { /* тело цикла пустое - действий нет */ } }
30
31  // копировать ptr_2 в ptr_1, используя нотацию указателей
32  void copy_2( char *ptr_1, const char *ptr_2 )
33  {
34          for (; ( *ptr_1 = *ptr_2 ) != '\0' ; ptr_1++, ptr_2++ )
35                  { /* тело цикла пустое - ничего не делаем */ } }
36
37  //error C2296: '*=' : illegal, left operand has type 'char *'
38  void copy_3( char *ptr_1, const char *ptr_2 )
39  {
40  //      for (; ( *ptr_1 = *ptr_2 ) != '\0' ; ptr_1 *= 2, ptr_2 *= 2 )
41  //      { /* тело цикла пустое - ничего не делаем */ }  }
```

```
1   //Прогр. 3.14.
2   /* Программа демонстрации использования массивов указателей.
3   Программа реализует алгоритм получения всеми студентами
```

```
4   всех экзаменационных вопросов в произвольном порядке.
5   Программа является крайне неоптимальной.*/
6   #include <iostream>
7   #include <cstdlib>
8   #include <ctime>
9
10  #define QUESTION 13
11  #define STUDENT 9
12
13  using std::cout; using std::endl;
14
15  // прототипы функций перемешивания и выбора
16  void to_move( int w_Table[][QUESTION] );
17  void to_choose( const int w_Table[][QUESTION], const char *w_Question[], const char
    *w_Student[]);
18
19  void main()
20  {
21          using std::cout; using std::ios; using std::endl;
22
23          const char *student[STUDENT] = {"Balandina", "Belyeva",
24                  "Vashonin", "Golovina","Gornakova","Kurepkina",
25                  "Lipanova","Pticin","Migiddorg"};
26
27          const char *question[QUESTION] = {"What is C/C++?",
28                  "Who is Dennis Ritchie?", "What is C++?",
29                  "Who is Ken Thompson?", "What is C#?",
30                  "Who is Bjarne Stroustrup?", "What is .NET?",
31                  "Who is Nicklaus Wirth?", "What is .NET Framework?",
32                  "Who is Pascal?", "What is MSIL?", "Who is Kulida?",
33                  "What is MFC or FCL?"};
34          // инициализировать массив таблицы
35          int table[STUDENT][QUESTION] = { 0 };
36
37          srand( time( 0 ) ); //запустить генератор случайных чисел
38
39          to_move( table );
40          to_choose( table, question, student );
41  }
42
43  void to_move( int w_Table[][QUESTION] ) //перемешивание
44  {
45          //номер строки, номер столбца, счётчик
46          int row, column, counter;
47
48          //для каждого сочетания из STUDENT * QUESTION случайным
49          // образом выбрать "ячейку в таблице" и назначить ей номер
50          for ( counter = 1; counter <= STUDENT * QUESTION; counter++ ) {
51
52                  //выбирать новую "ячейку", пока не найдётся свободная
53                  do {
54                          row = rand() % STUDENT; //
```

```
55                          column = rand() % QUESTION; //
56                  } while( w_Table[ row ][ column ] != 0 );
57
58                  //занести номер в нашедшуюся "ячейку" таблицы"
59                  w_Table[ row ][ column ] = counter;
60          }
61  } // конец функции to_move
62
63  void to_choose( const int w_Table[][QUESTION], const char *w_Question[],const char
    *w_Student[]) //выбор
64  {
65          //счётчики: строк, столбцов, "ячеек"
66          int row, column, counter;
67
68          //выбрать каждое из STUDENT * QUESTION сочетаний
69          for ( counter = 1; counter <= STUDENT * QUESTION; counter++ ) {
70
71                  //цикл по строкам w_Table
72                  for ( row = 0; row <= STUDENT - 1; row++ ) {
73
74                      //цикл по столбцам w_Table в текущей строке
75                      for ( column = 0; column <= QUESTION - 1; column++ ) {
76
77                      //если ячейка содержит текущее значение - вывести его
78                              if(w_Table[row][column] == counter) {
79                                cout << counter <<". "<< w_Student[row] << " - "
80                                              << w_Question[column] << endl; }
81                      }
82                  }
83          }
84  } // конц функции to_choose


1   //Прогр. 3.14m. – Модифицирован вывод 3.14
2   #include<iostream>
3   #include<cstdlib>
4   #include<ctime>
5
6   #define QUESTION 13
7   #define STUDENT 9
8
9   using std::cout; using std::endl; using std::ios;
10
11  void to_move(int w_Table[][QUESTION]);
12  void to_choose(const int w_Table[][QUESTION], const char *w_Question[], const char
    *w_Student[]);
13
14  void main()
15  {
16  const char *student[STUDENT] = {"Balandina","Belyeva","Vahonin",
17  "Golovina","Gornakova","Lipanova","Kurepkina","Pticin","Migiddorg"};
18  const char *qestion[QUESTION] = {"What is C/C++?", "Who is Dennis
```

```
Ritchie?", "What is C++?", "Who is Ken Thompson?", "What is C#?",
"Who is Bjarne Stroustrup?", "What is .NET?", "Who is Nicklaus Wirth?",
"What is .NET Framework?", "Who is Pascal?", "What is MSIL?",
"Who is Kulida?","What is MFC or FCL?"};

int table[STUDENT][QUESTION]={0};
        srand(time(0));
        to_move(table);
        to_choose(table, qestion, student);
}
void to_move(int w_Table[][QUESTION])
{
    int row,column,counter;

        for(counter=1; counter <= STUDENT*QUESTION; counter++){

do{
    row=rand()%STUDENT;
    column=rand()%QUESTION;
}while(w_Table[row][column]!=0);
    w_Table[row][column]=counter;

}}

void to_choose(const int w_Table[][QUESTION], const char *W_Question[], const char
*w_Student[])
{
int row, column, counter;
int i=1;
    for(counter=1; counter<=STUDENT; counter++) {  //счетчик
        for(row=0; row <= STUDENT-1; row++)  {
        cout << counter << "." << w_Student[row] << " -" << endl;
        for (column = 0; column <= QUESTION-1; column++)  {
            cout.setf(ios::right);
            cout.width(5);
            cout << column+1 << ". " << W_Question[column] << endl;

             }
        }
    }
}
```

```
//Прогр. 3.15 - аналог прогр. 3.5 и 3.9 по назначению.
/* Программа сортировки значений массива и в восходящем порядке,
   и в нисходящем порядке(пузырьковая сортировка/сортировка погружением).
   Программа демонстрирует использование указателей на функции. */
#include <iostream>
#define SIZE 10
using std::cout; using std::cin; using std::endl;

//Прототипы трёх функций - область действия - файл
```

```
10          //ф-ия сортировки
11  void sort(int array[], const int size, int (*ptr_fnt) (int a, int b) );
12
13  int top( int a, int b ); //ф-ия - восходящий порядок сортировки
14  int low( int a, int b ); //ф-ия - нисходящий порядок сортировки
15
16  int main()
17  {
18          int check; //выбор режима: 1 - восходящий; 2 - нисходящий
19          int i; // счётчик
20          int a[SIZE] = {1,99,5,34,67,7,3,41,37,25};
21
22          cout << "Enter 1 to sort in ascending order, 2 - descending order ";
23          cin >> check;
24
25          //вывести содержимое исходного массива
26          cout << "\nData items in original order\n";
27          for (i = 0; i < SIZE; i++) {
28                  cout.width(4);
29                  cout << a[i]; } cout << endl;
30
31  /* Вызвать ф-ию сортировки массива, передав ей в качестве аргумента
32             указатель на одну из функций.
33             Выбор ф-ии определяется выбранным порядком сортировки */
34          if ( check == 1 ) {
35                  sort( a, SIZE, top );
36                  cout << "\nData items in ascending order\n"; }
37          else if ( check == 2 ) {
38                  sort( a, SIZE, low );
39                  cout << "\nData items in descending order\n"; }
40          else return 0; //exit(EXIT_FAILURE);
41
42          //вывести содержимое отсортированного массива
43          for (i = 0; i < SIZE; i++) {
44                  cout.width(4);
45                  cout << a[i]; } cout << endl;
46          return 0;
47  } // конец main
48
49  // определение функции сортировки
50  /* реализут двунаправленную ф-ию пузырьковой сортировки,
51     *ptr_fnt - указатель на ф-ию сравнения,
52     определяющую порядок сортировки */
53  void sort( int array[],const int size,int(*ptr_fnt)(int a,int b) )
54  {
55          int pass; // счётчик числа походов
56          int count; // счётчик числа сравнений
57
58          //прототип ф-ии обмена значениями
59          void change( int *ptr_element1, int *ptr_element2 );
60
61          //цикл для задания числа проходов по массиву
```

```
62          for (pass = 1; pass < size; pass++) {
63
64                  //цикл задания числа сравнений на данном проходе
65                  for (count = 0; count < size - 1; count++) {
66
67                      /*сравнить значения соседних элементов и обменять их
68                      если они не соответствуют выбранному порядку */
69                        if ( (*ptr_fnt)(array[count], array[count + 1]) ) {
70                              change( &array[count], &array[count + 1 ] );
71                        } //конец if
72                  } //конец внутреннего for
73          } // конец внешнего for
74  } // конец функции sort
75
76  /* Определение функции change, обменивающей значения в ячейках,
77    на которые указывают указатели ptr_element1 и ptr_element2 */
78  void change( int *ptr_element1, int *ptr_element2 )
79  {
80          int swap = *ptr_element1;
81          *ptr_element1 = *ptr_element2;
82          *ptr_element2 = swap;
83  } // конец функции change
84
85  //функция сравнения для случая восходящей сортировки
86  int top( int a, int b )
87  {
88          return b < a; //отдать true, если b < a - обменять
89  }
90
91  //функция сравнения для случая нисходящей сортировки
92  int low( int a, int b )
93  {
94          return b > a; //отдать true, если b > a - обменять
95  }
```

4. Структуры

```
1   //Прогр. 4.1. - Структуры
2   /* Отладочный вариант 1: 1 - БД не нормализована (нет реляции);
3     2 - соответственно, отсутствуют другие структуры и указатели
4     на них (их элементы); 3 – кириллица; 4 - файлы и т.д */
5   #include <iostream>
6   #include <cctype>
7   #include <cstring>
8   #include <cstdlib>
9   #define SIZE 100
10  using namespace std;
11
12  struct tbl1_type {
13          char name_1[20]; // wchar_t
```

```
14              int index;
15              short unsigned course;
16              char faculty[5];
17              char specialization[50];
18              char date[10];
19              bool budget;
20      } list_student[SIZE];
21
22      void enter(), init_list(), display();
23      void update(), input(int i);
24      int menu();
25
26      int main()
27      {
28              char check;
29
30              init_list();
31
32              for(;;) {
33                      check = menu();
34                      switch(check) {
35                              case 'e': enter(); break;
36                              case 'd': display(); break;
37                              case 'u': update(); break;
38                              case 'q': return 0; }
39              }
40      }
41
42      // Инициализация массива структур
43      void init_list()
44      {
45              int k;
46              // Имя нулевой длины - пустое имя
47              for(k=0; k<SIZE; k++) *list_student[k].name_1 = '\0';
48      }
49
50      // Получение выбранной команды меню
51      int menu()
52      {
53              char chk;
54
55      //      cout << 'n';
56              do {
57                      cout << " (E)nter\n";    // Ввести новый пункт
58                      cout << " (D)isplay\n";  // Отобразить список
59                      cout << " (U)pdate\n";   // Изменить элемент
60                      cout << " (Q)uit\n\n";   // Выход из программы
61                      cout << "Choose a command: "; // Введите команду
62                      cin >> chk;
63              } while(!strchr("eduq", tolower(chk)));
64              return tolower(chk);
65      }
```

```
66
67  // Ввод пунктов в список
68  void enter()
69  {
70          int i;
71
72          // Поиск первой свободной структуры
73          for(i = 0; i<SIZE; i++)
74                  if(!*list_student[i].name_1) break;
75
76          // Если массив заполнен, значение i будет равно SIZE
77                  if(i==SIZE) {
78                          cout << "List is complete\n"; // Список заполнен
79                          return;  }
80
81                  input(i);
82  }
83
84  // Ввод данных
85  void input(int i)
86  {
87          cout << "Name: ";
88          cin >> list_student[i].name_1;
89
90          cout << "Номер СБ: ";
91          cin >> list_student[i].index;
92
93          cout << "Course: ";
94          cin >> list_student[i].course;
95
96          cout << "Faculty: ";
97          cin >> list_student[i].faculty;
98
99          cout << "Specialization: ";
100         cin >> list_student[i].specialization;
101
102         cout << "Birth Date: ";
103         cin >> list_student[i].date;
104
105         cout << "Budget?: ";
106         cin >> list_student[i].budget;
107 }
108
109 // Изменение существующего пункта списка
110 void update()
111 {
112         int i;
113         char name[30];
114
115         cout << "Enter a Name: ";
116         cin >> name;
117
```

```
118             for(i = 0; i < SIZE; i++)
119                     if(!strcmp(name, list_student[i].name_1)) break;
120
121                     if(i==SIZE) {
122                             cout << "This name don't exist\n";
123                             return;  }
124
125                     cout << "Enter new information\n";
126                     input(i); // Введите новую информацию
127     }
128
129     // Вывод на экран списка
130     void display()
131     {
132             int k;
133
134             for( k = 0; k < SIZE; k++) {
135                     if(*list_student[k].name_1) {
136                             cout << list_student[k].name_1 << '\n';
137                             cout << list_student[k].index << '\n';
138                             cout << list_student[k].faculty << '\n';
139                             cout << list_student[k].specialization << '\n';
140                             cout << list_student[k].date << '\n';
141                             cout << list_student[k].budget << '\n'; }
142             }
143     }
```

```
1       // Прогр. 4.2. - Структуры, операции: элемента структуры (.) и указателя
2       // структуры (->).
3       /* Учебный пример 2: 1 - имеет место реляция только 2х структур,
4       реализованная по методике POP; 2 - программа не оптимизирована */
5       #include <iostream>
6       #include <cctype>
7       #include <cstring>
8       #include <cstdlib>
9       #define SIZE_TBL_1 100
10      #define SIZE_TBL_2 10
11      using namespace std;
12
13      struct tbl_1_student {
14              char name_1[20];
15              int index;
16              char course;
17              struct tbl_2_faculty *ptr_faculty;
18              char specialization[50];
19              char date[10];
20              bool budget;
21      } list_student[SIZE_TBL_1];
22
23      struct tbl_2_faculty {
24              short unsigned key_faculty; //short unsigned int 0-65535(16bit)
```

```
25          char faculty_short[5];
26          char faculty_full[50];
27  } list_faculty[SIZE_TBL_2];
28
29  void faculty_enter(), student_enter(), init_list(), display(); int menu();
30  void input_faculty(int i), input_student(int i), choice_faculti(int i);
31
32  int main()
33  {
34          char check;
35          init_list();
36          for(;;) {
37                  check = menu();
38                  switch(check) {
39                          case 'f': faculty_enter(); break;
40                          case 's': student_enter(); break;
41                          case 'd': display(); break;
42                          case 'q': return 0; }
43          }
44  }
45
46  // Инициализация массивов структур
47  void init_list()
48  {
49          int k;
50          // Имя нулевой длины - пустое имя - инициализация "пустым"
51          for(k=0; k<SIZE_TBL_1; k++) *list_student[k].name_1 = '\0';
52          for(k=0; k<SIZE_TBL_2; k++) *list_faculty[k].faculty_short = '\0';
53  }
54
55  // Получение выбранной команды меню
56  int menu()
57  {
58          char chk;
59          do {
60                  cout << " (F)aculty enter\n"; // Ввести факультет
61                  cout << " (S)tudent enter\n"; // Ввести студента
62                  cout << " (D)isplay\n";  // Отобразить список
63                  cout << " (Q)uit\n\n";   // Выход из программы
64                  cout << "Choose a command: "; // Введите команду
65                  cin >> chk;
66          } while(!strchr("fsdq", tolower(chk)));
67          return tolower(chk);
68  }
69
70  // Ввод новых пунктов в список факультетов
71  void faculty_enter()
72  {                                   //можно было бы использовать шаблон ф-ии?
73          int i;
74          // Поиск первой свободной записи
75          for(i = 0; i<SIZE_TBL_2; i++)
76                  if(!*list_faculty[i].faculty_short) break;
```

```
77              // Если массив заполнен, значение i будет равно SIZE_TBL_2
78                      if(i==SIZE_TBL_2) {
79                              cout << "List of faculty is complete\n";
80                              return;  }
81                      input_faculty(i);
82      }
83
84      // Ввод новых пунктов в список студентов
85      void student_enter()
86      {                                       //можно было бы использовать шаблон ф-ии?
87              int i;
88              // Поиск первой свободной записи
89              for(i = 0; i<SIZE_TBL_1; i++)
90                      if(!*list_student[i].name_1) break;
91              // Если массив заполнен, значение i будет равно SIZE_TBL_1
92                      if(i==SIZE_TBL_1) {
93                              cout << "List of student is complete\n";
94                              return;  }
95                      input_student(i);
96      }
97
98      // Ввод данных по факультету
99      void input_faculty(int i)
100     {
101             cout << "Name faculty short: ";
102             cin >> list_faculty[i].faculty_short;
103             cout << "Name faculty_full: ";
104             cin >> list_faculty[i].faculty_full;
105             list_faculty[i].key_faculty = i; // индекс - ключевое поле
106             /* в данной программе не используется, связывать поля
107             можно было бы и с его использованием */
108     }
109
110     // Ввод данных по студенту
111     void input_student(int i)
112     {
113             cout << "Enter name student: ";
114             cin >> list_student[i].name_1;
115             cout << "Enter number book to reckon: ";
116             cin >> list_student[i].index;
117             cout << "Course: ";
118             cin >> list_student[i].course;
119
120             cout << "Faculty: ";
121             choice_faculti(i);
122
123             cout << "Specialization: ";
124             cin >> list_student[i].specialization;
125             cout << "Birth Date: ";
126             cin >> list_student[i].date;
127             cout << "Budget?: ";
128             cin >> list_student[i].budget;
```

```
129 }
130 // Формирование меню выбора факультета - выбор факультета
131 void choice_faculti(int i)
132 {
133         int k = 0;
134         while(*list_faculty[k].faculty_short) {
135                 cout << k + 1 << " - " << list_faculty[k].faculty_short
136                         << "; " << '\n';
137                 ++k;
138         }
139         cout << "Enter a command: "; // Выбор факультета (1, ...)
140         int chk;
141                 cin >> chk;
142                 list_student[i].ptr_faculty = &list_faculty[chk-1];
143 }
144
145 // Вывод на экран списка студентов
146 void display()
147 {
148         int k;
149
150         for( k = 0; k < SIZE_TBL_1; k++) {
151                 if(*list_student[k].name_1) {
152                         cout << list_student[k].name_1 << '\n';
153                         cout << list_student[k].index << '\n';
154
155                         cout << list_student[k].ptr_faculty->faculty_short << '\n';
156                         cout << list_student[k].ptr_faculty->faculty_full << '\n';
157
158                         cout << list_student[k].specialization << '\n';
159                         cout << list_student[k].date << '\n';
160                         cout << list_student[k].budget << '\n'; }
161         }
162 }
```

```
1   //Прогр. 4.3. - "Блочная" запись/считывание из файла
2   /* Модификация программ 4.1 и 4.2 */
3   #include <iostream>
4   #include <fstream> // ***
5   #include <cctype>
6   #include <cstring>
7   #include <cstdlib>
8   #define SIZE_TBL_1 100
9   #define SIZE_TBL_2 10
10  using namespace std;
11
12  struct tbl_1_student {
13          char name_1[20];
14          int index;
15          char course;
16          struct tbl_2_faculty *ptr_faculty;
```

```
17              char specialization[50];
18              char date[10];
19              bool budget;
20      } list_student[SIZE_TBL_1];
21
22      struct tbl_2_faculty {
23              short unsigned key_faculty; //short unsigned int 0-65535(16bit)
24              char faculty_short[5];
25              char faculty_full[50];
26      } list_faculty[SIZE_TBL_2];
27
28      void faculty_enter(), student_enter(), init_list(), display();
29      void input_faculty(int i), input_student(int i), choice_faculti(int i);
30      int save_to_file(), extract_from_file(), int menu();
31
32      int main()
33      {
34              char check;  init_list();
35              for(;;) {
36                      check = menu();
37                      switch(check) {
38                              case 'f': faculty_enter(); break;
39                              case 's': student_enter(); break;
40                              case 'd': display(); break;
41                              case 'w': save_to_file(); break; //*****
42                              case 'r': extract_from_file(); break; //*****
43                              case 'q': return 0; }
44              }
45      }
46
47      // Инициализация массивов структур (очистка массивов)
48      void init_list()
49      {
50              int k;
51              // Имя нулевой длины - пустое имя - инициализация "пустым"
52              for(k=0; k<SIZE_TBL_1; k++) *list_student[k].name_1 = '\0';
53              for(k=0; k<SIZE_TBL_2; k++) *list_faculty[k].faculty_short = '\0';
54      }
55
56      // Получение выбранной команды меню
57      int menu()
58      {
59              char chk;
60              do {
61                      cout << " (F)aculty enter\n"; // Ввести факультет
62                      cout << " (S)tudent enter\n"; // Ввести студента
63                      cout << " (D)isplay\n";  // Отобразить список
64                      cout << " (W)rite\n";  // Записать в файл //*****
65                      cout << " (R)ead\n";  // Считать из файла //*****
66                      cout << " (Q)uit\n\n";   // Выход из программы
67                      cout << "Choose a command: "; // Введите команду
68                      cin >> chk;
```

```
69          } while(!strchr("fswrdq", tolower(chk))); //*****
70          return tolower(chk);
71  }
72
73  // Ввод новых пунктов в список факультетов
74  void faculty_enter()
75  {                                   //можно было бы использовать шаблон ф-ии
76          int i;
77          // Поиск первой свободной записи
78          for(i = 0; i<SIZE_TBL_2; i++)
79                  if(!*list_faculty[i].faculty_short) break;
80          // Если массив заполнен, значение i будет равно SIZE_TBL_2
81                  if(i==SIZE_TBL_2) {
82                       cout << "List of faculty is complete\n"; // Список заполнен
83                          return;  }
84                  input_faculty(i);
85  }
86  // Ввод новых пунктов в список студентов
87  void student_enter()
88  {
89          int i;
90          // Поиск первой свободной записи
91          for(i = 0; i<SIZE_TBL_1; i++)
92                  if(!*list_student[i].name_1) break;
93          // Если массив заполнен, значение i будет равно SIZE_TBL_1
94                  if(i==SIZE_TBL_1) {
95                          cout << "List of student is complete\n";
96                          return;  }
97                  input_student(i);
98  }
99
100 // Ввод данных по факультету
101 void input_faculty(int i)
102 {
103         cout << "Name faculty short: ";
104         cin >> list_faculty[i].faculty_short;
105         cout << "Name faculty_full: ";
106         cin >> list_faculty[i].faculty_full;
107         list_faculty[i].key_faculty = i; // индекс - ключевое поле
108         /* в данной программе не используется, связывать поля
109         можно было бы и с его использованием */
110 }
111
112 // Ввод данных по студенту
113 void input_student(int i)
114 {
115         cout << "Enter name student: ";
116         cin >> list_student[i].name_1;
117         cout << "Enter number book to reckon: ";
118         cin >> list_student[i].index;
119         cout << "Course: ";
120         cin >> list_student[i].course;
```

```
121
122             cout << "Faculty: ";
123             choice_faculti(i);
124             cout << "Specialization: ";
125             cin >> list_student[i].specialization;
126             cout << "Birth Date: ";
127             cin >> list_student[i].date;
128             cout << "Budget?: ";
129             cin >> list_student[i].budget;
130     }
131     // Формирование меню выбора факультета - выбор факультета
132     void choice_faculti(int i)
133     {
134             int k = 0;
135             while(*list_faculty[k].faculty_short) {
136                     cout << k + 1 << " - " << list_faculty[k].faculty_short
137                             << "; " << '\n';
138                     ++k;
139             }
140             cout << "Enter a command: "; // Выбор факультета (1, ...)
141             int chk;
142                     cin >> chk;
143                     list_student[i].ptr_faculty = &list_faculty[chk-1];
144     }
145
146     // Вывод на экран списка студентов
147     void display()
148     {
149             int k;
150
151             for( k = 0; k < SIZE_TBL_1; k++) {
152                     if(*list_student[k].name_1) {
153                             cout << list_student[k].name_1 << '\n';
154                             cout << list_student[k].index << '\n';
155
156                             cout << list_student[k].ptr_faculty->faculty_short << '\n';
157                             cout << list_student[k].ptr_faculty->faculty_full << '\n';
158
159                             cout << list_student[k].specialization << '\n';
160                             cout << list_student[k].date << '\n';
161                             cout << list_student[k].budget << '\n'; }
162             }
163     }
164
165     // Запись таблиц в файлы (имя safe_record - что то служебное?)
166     int save_to_file()
167     {
168             ofstream out_1("tbl_1", ios::out | ios::binary);
169             if(!out_1) {
170                     cout << "Fail is not open.\n";
171                     return 1;
172             }
```

```
173             out_1.write((char *) &list_student, sizeof list_student);
174             out_1.close();
175
176             ofstream out_2("tbl_2", ios::out | ios::binary);
177             if(!out_2) {
178                     cout << "Fail is not open.\n";
179                     return 1;
180             }
181             out_2.write((char *) &list_faculty, sizeof list_faculty);
182             out_2.close();
183             return 0;
184     }
185
186     // Считывание таблиц из файлов - вывод списка
187     int extract_from_file()
188     {
189             init_list(); // очистка массивов
190
191             ifstream in_1("tbl_1", ios::in | ios::binary);
192             if(!in_1) {
193                     cout << "Fail is not open.\n";
194                     return 1;
195             }
196             in_1.read((char *) &list_student, sizeof list_student);
197             in_1.close();
198
199             ifstream in_2("tbl_2", ios::in | ios::binary);
200             if(!in_2) {
201                     cout << "Fail is not open.\n";
202                     return 1;
203             }
204             in_2.read((char *) &list_faculty, sizeof list_faculty);
205             in_2.close();
206
207             display();
208
209             return 0;
210     }
```

5. Классы

```
1     / Прогр. 5.1.
2     // Класс Time
3
4     #include <iostream>
5     using std::cout; using std::endl; using std::cin;
6
7     // Определение класса Time (абстрактного типа данных - ADT)
8     class Time {
9     public:
```

```
10          Time();                                                 // конструктор
11          void setTime(int, int, int); // установка часов, минут, секунд
12          void printMilitary();           // вывести в полгном формате
13          void printStandard();           // вывести в формате «12»
14  private:
15          int hour;   // диапазон 0 - 23
16          int minute; // диапазон 0 - 59
17          int second; // диапазон 0 - 59
18  }; // конец тела класса Time
19
20  /* Конструктор класса Time инициализирует каждый элемент данных
21   нулём и гарантирует, что объект класса Time создаётся согласованно */
22
23  Time::Time() {hour = minute = second = 0;}
24
25  /* Функция установки нового значения Time (в полном формате).
26  Проверка допустимости значений. Недействительные значения заменить
27  нулём. */
28  void Time::setTime(int h, int m, int s)
29  {
30          hour = ( h >= 0 && h < 24 ) ? h : 0;
31          minute = ( m >= 0 && m < 60 ) ? m : 0;
32          second = ( s >= 0 && s < 60 ) ? s : 0;
33  } // конец тела функции setTime
34
35  // Функция вывода времени в полном формате.
36  void Time::printMilitary()
37  {
38          cout << ( hour <10 ? "0" : "" ) << hour << ":"
39            << ( minute < 10 ? "0" : "" ) << minute;
40  } // конец тела функции pritMilitary
41
42  // Функция вывода времени в формате «12»
43  void Time::printStandard()
44  {
45          cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 )
46                  << ":" << (minute < 10 ? "0" : "" ) << minute
47                  << ":" << (second < 10 ? "0" : "" ) << second
48                  << ( hour < 12 ? "AM" : "PM" );
49  } // конец тела функции printStandard
50
51  // Программа - клиент класса Time
52  void main()
53  {
54          Time t; // создать объект (представитель) t класса Time
55
56          cout << "The inicial military time is ";
57          t.printMilitary();
58          cout << "\nThe initial standard timi is ";
59          t.printStandard();
60
61          t.setTime(13,27,6);
```

```
62          cout << "\nMilitary time after setTime is ";
63          t.printMilitary();
64          cout << "\nStandard time after setTime is ";
65          t.printStandard();
66
67          t.setTime(99,99,99);
68          cout << "\n\nAfter attempting invalid settings: "
69                  << "\nMilitary time: ";
70          t.printMilitary();
71          cout << "\nStandard time: ";
72          t.printStandard();
73          cout << endl;
74
75          cout << "\n\nEnter time (hour : minute : second): ";
76          int a,b,c;
77          cin >> a >> b >> c;
78          t.setTime(a,b,c);
79          cout << "\nMilitary time after enterTime is ";
80          t.printMilitary();
81          cout << "\nStandard time after enterTime is ";
82          t.printStandard();
83          cout << endl;
84
85  } // конец тела main
```

```
1   // Прогр. 5.2.
2   // Демонстрационная программа операций доступа к элементам класса,
3   // операция выбора элемента ->, применяемая в сочетании с указателем на
4   // объект. ОТКРЫТЫЕ ДАННЫЕ КЛАССА - НЕКОРРЕКТНЫЙ СТИЛЬ
5   //    ПРОГРАММИРОВАНИЯ
6   #include <iostream>
7   using std::cout; using std::endl;
8
9   // Простейший класс Count
10  class Count {
11  public:
12          int x;
13          void print() {cout << x << endl;}
14  }; // конец скелета класса Count
15
16  void main()
17  {
18          Count counter, // использование класса в качестве типа в
19                                  // объявлении объектов: создать объект counter,
20          *counterPtr = &counter, // указатель на counter,
21          &counterRef = counter; // ссылку на counter
22
23          cout << "Assign 7 to x and print using the object's name: ";
24          counter.x = 7;  // присвоить 7 элементу данных x
25          counter.print(); // вызвать элемент-функцию print
26
```

```
27          cout << "Assign 8 to x and print using reference: ";
28          counterRef.x = 8;  // присвоить 8 элементу данных x
29          counterRef.print(); // вызвать элемент-функцию print
30
31          cout << "Assign 10 to x and print using a pointer: ";
32          counterPtr->x = 10;  // присвоить 10 элементу данных x
33          counterPtr->print(); // вызвать элемент-функцию print
34  } // конец тела функции main
```

```
1   // Прогр. 5.3. - L_5_3.h
2   // Объявление класса Time в заголовочном файле L_5_3.h
3   // Элементы-функции класса определяются в файле L_5_3.cpp
4   // Отделение интерфейса класса Time от реализации
5   // Директивы препроцессора, исключающие возможность
6   // многократного (повторного - по ошибке) включения
7   // в проект заголовочного файла
8   #ifndef TIME1_H
9   #define TIME1_H
10
11  // Определение ADT - Time
12  class Time {
13  public:
14          Time();                                                 // конструктор
15          void setTime(int, int, int); // установка часов, минут, секунд
16          void printMilitary();           // вывести в  формате «24»
17          void printStandard();           // вывести в формате «12»
18  private:
19          int hour;   // диапазон 0 - 23
20          int minute; // диапазон 0 - 59
21          int second; // диапазон 0 - 59
22  }; // конец тела класса Time
23  #endif
24
25  // Прогр. 5.3. - L_5_3.cpp
26  // Определение элементов-функций класса Time - файл L_5_3.cpp
27  // Отделение интерфейса класса Time от реализации - файл L_5_3.cpp
28
29  #include <iostream>
30  using std::cout;
31  #include "L_5_3.h"
32
33  // Конструктор класса Time инициализирует каждый элемент данных
34  // нулём и гарантирует, что объект класса Time создаётся согласованно.

35  Time::Time() {hour = minute = second = 0;}
36
37  // Функция установки нового значения Time (в формате «24»). Проверка
38  // допустимости значений. Недействительные значения заменить нулём.
39  void Time::setTime(int h, int m, int s)
40  {
41          hour = ( h >= 0 && h < 24 ) ? h : 0;
```

```
42          minute = ( m >= 0 && m < 60 ) ? m : 0;
43          second = ( s >= 0 && s < 60 ) ? s : 0;
44  } // конец тела функции setTime
45
46  // Функция вывода времени в формате «24».
47  void Time::printMilitary()
48  {
49          cout << ( hour <10 ? "0" : "" ) << hour << ":"
50            << ( minute < 10 ? "0" : "" ) << minute;
51  } // конец тела функции pritMilitary
52
53  // Функция вывода времени в формате «12»
54  void Time::printStandard()
55  {
56          cout << ( ( hour = = 0 || hour = = 12 ) ? 12 : hour % 12 )
57                  << ":" << (minute < 10 ? "0" : "" ) << minute
58                  << ":" << (second < 10 ? "0" : "" ) << second
59                  << ( hour < 12 ? "AM" : "PM" );
60  } // конец тела функции printStandard
61
62  // Прогр. 5.3. - Test_L_5_3.cpp
63  // Программа-клиент тестер класса Time
64  // ЗАМЕЧАНИЕ: компилируется совместно с L_5_3.cpp
65  #include <iostream>
66  #include "L_5_3.h"
67  using std::cout; using std::endl; using std::cin;
68
69  void main()
70  {
71          Time t; // создать объект (представитель) t класса Time
72
73          cout << "The inicial military time is ";
74          t.printMilitary();
75          cout << "\nThe initial standard timi is ";
76          t.printStandard();
77
78          t.setTime(13,27,6);
79          cout << "\nMilitary time after setTime is ";
80          t.printMilitary();
81          cout << "\nStandard time after setTime is ";
82          t.printStandard();
83
84          t.setTime(99,99,99);
85          cout << "\n\nAfter attempting invalid settings: "
86                  << "\nMilitary time: ";
87          t.printMilitary();
88          cout << "\nStandard time: ";
89          t.printStandard();
90          cout << endl;
91
92          cout << "\n\nEnter time (hour : minute : second): ";
93          int a,b,c;
```

```
94              cin >> a >> b >> c;
95              t.setTime(a,b,c);
96              cout << "\nMilitary time after enterTime is ";
97              t.printMilitary();
98              cout << "\nStandard time after enterTime is ";
99              t.printStandard();
100             cout << endl;
101
102     } // конец тела main
```

```
1       // Прогр. 5.4.
2       // Объявление класса Time в заголовочном файле L_5_4.h
3       // Элементы-функции класса определяются в файле L_5_4.cpp
4
5       // Директивы препроцессора, исключающие многократное
6       // включение в проект заголовочного файла
7       #ifndef TIME1_H
8       #define TIME1_H
9
10      // Определение ADT - Time
11      class Time {
12      public:
13              Time(int = 0, int = 0, int = 0); // конструктор по умолчанию
14              void setTime(int, int, int); // установка часов, минут, секунд
15              void printMilitary();           // вывести в "полгом" формате
16              void printStandard();           // вывести в "сокращённом" формате
17      private:
18              int hour;   // диапазон 0 - 23
19              int minute; // диапазон 0 - 59
20              int second; // диапазон 0 - 59
21      }; // конец тела класса Time
22      #endif
23
24      // Прогр. 5.4.
25      // Определение функций-элементов класса Time - файл L_5_4.cpp
26      #include <iostream>
27      using std::cout;
28      #include "L_5_4.h"
29
30      // Конструктор класса Time инициализирует элементы данных нулями.
31      // и гарантирует, что все объекты класса Time создаются в корректном
32      // состоянии.
33      Time::Time(int hr, int min, int sec)
34              {setTime(hr, min, sec);}
35
36      // Установить новые значения Time (в "полном" формате). Проверить
37      // допустимости значений. Недействительные значения заменить нулём.
38      void Time::setTime(int h, int m, int s)
39      {
40              hour   = ( h >= 0 && h < 24 ) ? h : 0;
41              minute = ( m >= 0 && m < 60 ) ? m : 0;
```

```
42          second = ( s >= 0 && s < 60 ) ? s : 0;
43  } // конец тела функции setTime
44
45  // Вывести время в "полном" формате.
46  void Time::printMilitary()
47  {
48          cout << ( hour <10 ? "0" : "" ) << hour << ":"
49            << ( minute < 10 ? "0" : "" ) << minute;
50  } // конец тела функции pritMilitary
51
52  // Напечатать время в "сокращённом" формате
53  void Time::printStandard()
54  {
55          cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 )
56                  << ":" << (minute < 10 ? "0" : "" ) << minute
57                  << ":" << (second < 10 ? "0" : "" ) << second
58                  << ( hour < 12 ? " AM" : " PM" );
59  } // конец тела функции printStanrd
60
61  // Прогр. 5.4. - Test_L_5_4.cpp
62  // Демонстрация конструктора по умолчанию для класса Time
63  #include <iostream>
64  #include "L_5_4.h"
65  using std::cout; using std::endl; using std::cin;
66
67  void main()
68  {
69          // создать объекты класса Time
70          Time t1,          // все аргументы по умолчанию
71                  t2(2),       // минуты и секунды по умолчанию
72                  t3(21,34),   // секунды по умолчанию
73                  t4(12,25,42), // указаны все значения
74                  t5(27,74,99); // все значения недопустимы
75
76          cout << "Constructed with:\n"
77                  << "all arguments defaulted:\n   ";
78          t1.printMilitary();
79          cout << "\n   ";
80          t1.printStandard();
81
82          cout << "\nHour specified; minute and second defaulted:"
83                  << "\n   ";
84          t2.printMilitary();
85          cout << "\n   ";
86          t2.printStandard();
87
88          cout << "\nHour and minute specified; second defaulted:"
89                  << "\n   ";
90          t3.printMilitary();
91          cout << "\n   ";
92          t3.printStandard();
93
```

```
94              cout << "\nHour, minute and second specified:"
95                      << "\n   ";
96              t4.printMilitary();
97              cout << "\n   ";
98              t4.printStandard();
99
100             cout << "\nAll invalid values specified:"
101                     << "\n   ";
102             t5.printMilitary();
103             cout << "\n   ";
104             t5.printStandard();
105     } // конец тела main
```

Приложения

1. Арифметические операции

Операция	Обозначение операции в арифметике	Алгебраическое выражение	Команда в синтаксисе C/C++
Сложение	+	a + b	a + b
Вычитание	–	a – b	a – b
Умножение	·	a · b или ab	a * b
Деление	/	a/b или a : b	a / b
Взятие по модулю		a mod b	a % b

2. Операции определения условия

Операции отношения – обозначения в арифметике	Обозначение операции в C/C++	Пример условия в синтаксисе C/C++	Расшифровка условия
=	==	a == b	a равно b
≠	!=	a != b	a не равно b
>	>	a > b	a больше b
<	<	a < b	a меньше b
≥	>=	a >= b	a больше или равно b
≤	<=	a <= b	a меньше или равно b

3. Операции присвоения значений

Обозначение операций присвоения значения в C/C++	Примеры соответствующих операторов C/C++	Примеры соответствующих операторов в синтаксисе VB, Delphi (в C/C++ можно и аналогично)	Результаты выполнения команд примеров
/* Пусть */ int b = 5, c = 4, d = 7, e = 6, f = 12;			
=	a = 1;	a = 1	a присвоено значение 1
+ =	b + = 3;	b = b + 3	b присвоено значение 8
– =	c – = 1;	c = c – 1	c присвоено значение 3
* =	d *= 2;	d = d * 2	d присвоено значение 14
/ =	e /= 3;	e = e / 3	e присвоено значение 2
% =	f %= 9;	f = f % 9; f = f mod 9;	f присвоено значение 3

4. Операции задания шага

Операция в C/C++	Примеры операторов C/C++	Примеры соответствующих операторов в синтаксисе VB, Delphi (C/C++)	Результаты выполнения команд
Преинкремент	+ +a;	a = a + 1 (a += 1)	Изменение a на 1 до использования
Предекремент	– –a;	a = a – 1 (a += 1)	
Постинкремент	a+ +;	a = a + 1 (a += 1)	Изменение a на 1 после использования
Постдекремент	a– –;	a = a – 1 (a – = 1)	

Литература

1. Х. М. Дейтел, П. Дж. Дейтел. Как программировать на С.: Четвёртое издание. Пер. с англ. – М.:ООО «Бином-Пресс», 2006 г. – 912 с.: ил.

2. Г. Шилдт. Базовый курс С++.: . Пер. с англ. – М.: Издательский дом «Вильямс», 2009. – 752 с.: ил.

Оглавление

Учебное пособие по курсам «Прикладное программирование в информационных системах», «Программная реализация средств дизайна»
Введение в С/С++
Составители: Дмитрий Давидович Ветчинин
Научный редактор: Николай Анатольевич Коробов
Печатается в авторской редакции

УП № 020305 от 28.11.99. Подписано в печать 11.01.2010.
Формат 1/16 60*84. Бумага писчая. Плоская печать.
Усл. печ. л. 3,41 Уч.-изд. 3,2. Тираж 20 экз. Заказ № 23

РИО ИГТА
ЦОТ кафедры ПМИТ ИГТА
152000, г. Иваново пр. Ф. Энгельса 21