

Министерство образования Республики Беларусь  
Учреждение образования  
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Специальность: «Программное обеспечение информационных технологий»

**КОНТРОЛЬНАЯ РАБОТА**

По курсу: «Объектно-ориентированные технологии  
программирования и стандарты проектирования»  
Вариант 22

Студент-заочник 2 курса  
Группы № 581072  
ФИО: Богданова (Попенко)  
Кристина Евгеньевна  
Адрес: Польша, г. Реда,  
ул. Млынська 5Б-5  
Тел. +48 576 335 295

Минск, 2017

## Задание 1

Разработать класс, одной из компонент которого является символьная строка и внешнюю функцию (по отношению к классу), выполняющую удаление второго слова из символьной строки. Содержимое объекта (строку) до и после применения к нему внешней функции вывести на экран.

## Текст программы

### DeleteWord.h

```
#ifndef DELETEWORD
#define DELETEWORD

#include <iostream>
#include <string.h>
#include <Windows.h>

using namespace std;

// Класс для обмена позиции первого и последнего слова в строке.
class DeleteWord
{
public:
    // Конструкторы.
    DeleteWord();
    explicit DeleteWord(const char *);

    // Конструктор копирования.
    DeleteWord(DeleteWord &rhs);

    // Деструктор.
    ~DeleteWord();

    // Методы.
    friend void PrintSourceString(const DeleteWord & src);
    DeleteWord & SetNewString(const char *);
    char * DeleteSecondWord();

private:
    char * str;
    bool CheckString() const;
    char * TrimString() const;
};

#endif
```

### DeleteWord.cpp

```
#include "stdafx.h"
#include <stdio.h>
```

```

#include <string.h>
#include "DeleteWord.h"

// Конструктор.
DeleteWord::DeleteWord()
{
    str = new char[1];
    str[0] = '\0';
}

// char * Конструктор.
DeleteWord::DeleteWord(const char * ch)
{
    str = new char[strlen(ch) + 1];
    strcpy(str, ch);
}

// Конструктор копирования.
DeleteWord::DeleteWord>DeleteWord &rhs)
{
    str = new char[strlen(rhs.str) + 1];
    strcpy(str, rhs.str);
}

// Деструктор.
DeleteWord::~DeleteWord()
{
    delete[] str;
}

// Проверка строки.
bool DeleteWord::CheckString() const
{
    char * buffer = TrimString();
    if (strlen(buffer) == 0)
    {
        cout << "\nError #1: The string is empty!";
        return false;
    }

    if (strchr(buffer, ' ') == NULL)
    {
        cout << "\nError #2: There is only one word in string!";
        return false;
    }
    return 1;
}

// Удаление ненужных символов.
char * DeleteWord::TrimString() const
{
    char * buffer = new char[strlen(str) + 1];
    strcpy(buffer, str);

    // Удаление пробелов и табов в начала строки.
    int i = 0, j;
    while ((buffer[i] == ' ') || (buffer[i] == '\t'))
    {
        i++;
    }
    if (i>0)
    {
        for (j = 0; j< int(strlen(buffer)); j++)
        {

```

```

        buffer[j] = buffer[j + i];
    }
    buffer[j] = '\0';
}

// Удаление пробелов и табов в конце строки.
i = strlen(buffer) - 1;
while ((buffer[i] == ' ') || (buffer[i] == '\t'))
{
    i--;
}
if (i < int(strlen(buffer) - 1))
{
    buffer[i + 1] = '\0';
}

return buffer;
}

// Вывод строки на экран. (friend - функция)
void PrintSourceString(const DeleteWord & src)
{
    cout << src.str;
}

// Ввод строки.
DeleteWord & DeleteWord::SetNewString(const char * newstr)
{
    delete[] str;
    str = new char[strlen(newstr) + 1];
    strcpy(str, newstr);
    return *this;
}

// Удаление второго слова.
char * DeleteWord::DeleteSecondWord()
{
    if (!CheckString())
    {
        return NULL;
    }
    else
    {
        char * buffer = TrimString();
        int bufferlen = strlen(buffer);

        char * ptFirst = strchr(buffer, ' ');
        int firstStrLen = ptFirst - buffer;

        int otherBufStrLen = bufferlen - firstStrLen;
        char * otherBufString = new char[otherBufStrLen + 1];
        strcpy(otherBufString, ptFirst);

        char * buf = new char[otherBufStrLen + 1];
        strcpy(buf, otherBufString);
        int i = 0, j;
        while ((buf[i] == ' ') || (buf[i] == '\t'))
        {
            i++;
        }
        if (i > 0)
        {
            for (j = 0; j < int(strlen(buf)); j++)

```

```

        {
            buf[j] = buf[j + i];
        }
        buf[j] = '\0';
    }

    char * strBuf = new char[strlen(buf) + 1];
    strcpy(strBuf, buf);

    char * ptSecond = strchr(strBuf, ' ');
    int secondStrLen = ptSecond - strBuf;
    int otherStrLen = strlen(strBuf) - secondStrLen;

    char * res = new char[bufferlen + 1];
    char * firstWord = new char[firstStrLen + 1];
    char * secondWord = new char[secondStrLen + 1];
    char * otherString = new char[otherStrLen + 1];

    res[0] = '\0';
    firstWord[firstStrLen] = '\0';
    secondWord[secondStrLen] = '\0';
    otherString[otherStrLen] = '\0';

    strncpy(firstWord, buffer, firstStrLen);
    strncpy(secondWord, buffer, secondStrLen);
    strncpy(otherString, ptSecond, otherStrLen);

    strcat(res, firstWord);
    strcat(res, otherString);

    delete[] firstWord;
    delete[] secondWord;
    delete[] otherString;
    delete[] buffer;

    cout << res << endl;
    return res;
}
}

```

## Lab1.cpp

```

#include "stdafx.h"
#include <stdio.h>
#include <string.h>
#include "DeleteWord.h"

int main()
{
    DeleteWord * obj1;
    DeleteWord * obj2;

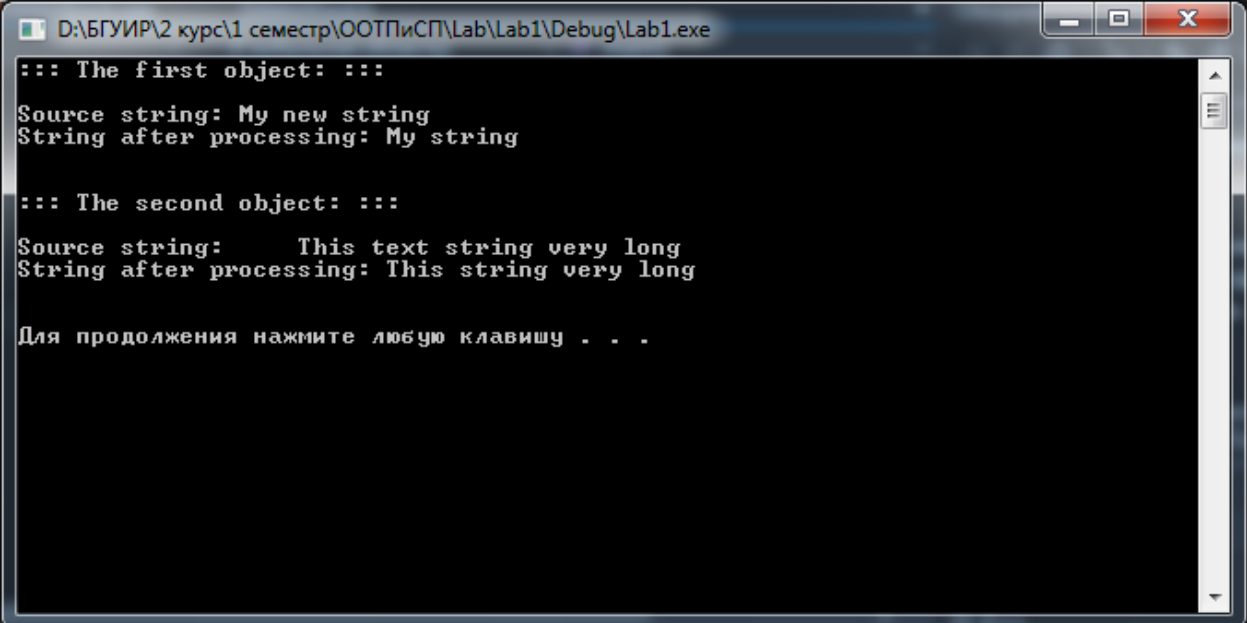
    cout << "::: The first object: :::\n\n";
    obj1 = new DeleteWord();
    obj1->SetNewString("My new string");
    cout << "Source string: ";
    PrintSourceString(*obj1);
    cout << "\nString after processing: ";
    obj1->DeleteSecondWord();

    cout << "\n\n::: The second object: :::\n\n";
    obj2 = new DeleteWord();
    obj2->SetNewString("    This text string very long    ");
}

```

```
    cout << "Source string: ";  
    PrintSourceString(*obj2);  
    cout << "\nString after processing: ";  
    obj2->DeleteSecondWord();  
    cout << "\n\n";  
  
    delete obj1;  
    delete obj2;  
  
    system("pause");  
  
    return 0;  
}
```

## Результат выполнения программы



```
D:\БГУИР\2 курс\1 семестр\ООТПиСП\Lab\Lab1\Debug\Lab1.exe  
::: The first object: :::  
Source string: My new string  
String after processing: My string  
  
::: The second object: :::  
Source string:      This text string very long  
String after processing: This string very long  
  
Для продолжения нажмите любую клавишу . . .
```

## Задание 2

Создать несколько объектов (например, а и b) разработанного класса. Класс – вектор (одномерный массив). Реализовать для объектов данного класса перегрузку операции ++ (b=++a). Содержимое объектов (а, b, их векторов), до и после выполнения операции, вывести на экран.

## Текст программы

### MyClass.h

```
#ifndef MYCLASS
#define MYCLASS

#include <iostream>
#include <string.h>

using namespace std;

class MyClass
{
public:
    MyClass();
    MyClass(const int * src);
    MyClass(const MyClass & src);
    ~MyClass();

    MyClass & Setarr(const int *);
    void Printarr() const;
    MyClass & operator ++ (const MyClass &);
private:
    int * arr;
};

#endif
```

### MyClass.cpp

```
#include "MyClass.h"

MyClass::MyClass()
{
    arr = new int[1];
    arr[0]='\0';
}

MyClass::MyClass(const int * src)
{
    arr = new int[arrlen(src)+1];
    arrcpy(arr, src);
}
```

```

MyClass::MyClass(const MyClass & src)
{
    arr = new int[arrlen(src.arr)+1];
    arrcpy(arr, src.arr);
}

MyClass::~MyClass()
{
    delete [] arr;
}

MyClass & MyClass::Setarr(const int * src)
{
    delete [] arr;
    arr = new int[arrlen(src)+1];
    arrcpy(arr, src);
    return *this;
}

void MyClass::Printarr() const
{
    cout << arr << endl;
}

MyClass & MyClass::operator -= (const MyClass & src)
{
    int * temp = new int [arrlen(arr)+1];
    arrcpy(temp, arr);

    int * rightarr = 0;
    int * leftarr = 0;
    int * pt = 0;

    while (0 != (pt=arrarr(temp, src.arr)))
    {
        leftarr = new int [pt - temp + 1];
        leftarr[pt - temp] = '\0';
        arrncpy(leftarr, temp, pt - temp);

        rightarr = new int [arrlen(pt) - arrlen(src.arr) + 1];
        rightarr[arrlen(pt) - arrlen(src.arr)] = '\0';
        arrncpy(rightarr, pt + arrlen(src.arr), arrlen(pt) -
arrlen(src.arr));

        delete [] temp;
        temp = new int [arrlen(leftarr) + arrlen(rightarr) + 1];
        temp[arrlen(leftarr) + arrlen(rightarr)] = '\0';
        arrcpy(temp, leftarr);
        arrcat(temp, rightarr);

        delete [] leftarr;
        delete [] rightarr;
    }
    this->Setarr(temp);
    delete [] temp;
    return *this;
}

```

## Lab2.cpp

```

#include "MyClass.h"

int main()

```



```
{  
    MyClass a(1, 2, 3, 4, 5);  
    MyClass b(2, 4, 6, 8, 10);  
  
    a.Printarr();  
    b=++a;  
    b.Printarr();  
  
    return 0;  
}
```

## Результат выполнения программы



```
D:\БГУИР\2 курс\1 семестр\ООП\ПисП\Lab\Lab2\Debug\Lab2.exe  
a:  
1, 2, 3, 4, 5  
b=++a:  
2, 3, 4, 5, 6  
Для продолжения нажмите любую клавишу . . . _
```

### Задание 3

Создать иерархию классов, представляющих простое наследование. Базовый класс – строка символов (char \*). Производный класс – методы, работающие с данными базового класса. Реализовать в производном классе метод – поиска и замены местами слов с максимальной и минимальной длиной в строке базового класса.

### Текст программы

#### CharBase.h

```
#ifndef CharBASE
#define CharBASE

#include <iostream>
#include <stdio.h>
#include <windows.h>
#include <time.h>

using namespace std;

class CharBase
{
public:
    CharBase():colCount(0), (0), a(0){}
    CharBase(int, int, bool=true);
    CharBase(const CharBase &);
    CharBase & RandomFilling(int=1, int=100);
    ~CharBase();
    void PrintChar() const;

    int ** Get() const;
    void Set(const int**);

protected:
    int colCount, ;
    int **a;
    int Random(int=1, int=100) const;
};

#endif
```

#### CharBase.cpp

```
#include "CharBase.h"

CharBase::CharBase( int col, bool fillRand)
{
    colCount = cols;
    a = new char * [ ];
    for(int i=0; i<colCount; a[i++] = new char [colCount]);
    if(fillRand) RandomFilling();
}
```

```

CharBase::CharBase(const CharBase & src)
{
    colCount = src.colCount;
    a = new int * [ ];
    for(int i=0; i<colCount; a[i++] = new char [colCount]);
        for(int j=0; j<colCount; ++j) a[i][j] = src.a[i][j];
}
CharBase::~CharBase()
{
    for(int i=0; i< ; delete [] a[i++]);
    delete [] a;
}
int CharBase::Random(int min, int max) const
{
    return rand()%(max - min) + min;
}
CharBase & CharBase::RandomFilling(int min, int max)
{
    srand (time(NULL));
    for(int i=0; i< ; ++i)
        for(int j=0; j<colCount; a[i][j++] = Random(min, max));
    return *this;
}
void CharBase::PrintChar() const
{
    for(int i=0; i< ; ++i, cout << endl)
        for(int j=0; j<colCount; cout << a[i][j++] << "\t");
    cout << endl;
}
int ** CharBase::Get() const
{
    return a;
}
void CharBase::Set(const int ** arr, int NewcolCount, int New )
{
    for(int i=0; i<this-> ; delete [] a[i++]);
    delete [] a;
    colCount = NewcolCount;
    a = new int * [ ];
    for(int i=0; i<colCount; a[i++] = new int [colCount]);
    for(int i=0; i< ; ++i)
        for(int j=0; j<colCount; ++j) a[i][j] = arr[i][j];
}

```

## CharExt.h

```

#include "CharBase.h"

#ifndef CharEXT
#define CharEXT

class CharExt: public CharBase
{
public:
    CharExt():CharBase(){}
    CharExt(int rows, int cols, bool fillRand=true):CharBase(rows, cols,
fillRand){}
    CharExt(const CharBase &src):CharBase(src){}

    int MaxElementsCol() const;
    CharExt & SpawCols(int, int);
};

#endif

```

## CharExt.cpp

```
#include "CharExt.h"

int CharExt::MaxElementsCol() const
{
    if(colCount<1)
    {
        cout << "Error! Columns count is 0!" << endl;
        return -1;
    }
    int ColElementsSumm = 0;
    int MaxSumm = 0;
    int ColMaxSumm = 0;
    for(int j=0; j<colCount; ++j)
    {
        ColElementsSumm = 0;
        for(int i=0; i< ; ++i)
        {
            ColElementsSumm += a[i][j];
        }
        if(j==0) MaxSumm = ColElementsSumm;
        else
            if(ColElementsSumm>=MaxSumm)
            {
                MaxSumm = ColElementsSumm;
                ColMaxSumm = j;
            }
    }
    cout << "Max summ = " << MaxSumm << " in column #" << ColMaxSumm + 1 <<
endl;
    return ColMaxSumm;
}

int CharExt::MinElementsCol() const
{
    if(colCount<1)
    {
        cout << "Error! Columns count is 0!" << endl;
        return -1;
    }

    int ColElementsSumm = 0;
    int MinSumm = 0;
    int ColMinSumm = 0;
    for(int j=0; j<colCount; ++j)
    {
        ColElementsSumm = 0;
        for(int i=0; i< ; ++i)
        {
            ColElementsSumm += a[i][j];
        }
        if(j==0) MinSumm = ColElementsSumm;
        else
            if(ColElementsSumm>=MinSumm)
            {
                MinSumm = ColElementsSumm;
                ColMinSumm = j;
            }
    }
    cout << "Min summ = " << MinSumm << " in column #" << ColMinSumm + 1 <<
endl;
    return ColMaxSumm;
}

CharExt & CharExt::SpawCols(int col1, int col2)
```

```

{
    if(col1>colCount || col1 < 0 || col2>colCount || col2 < 0)
    {
        cout << "Error! Column number out of range!" << endl;
        return *this;
    }

    for(int i=0; i< ; ++i)
    {
        int t = a[i][col1];
        a[i][col1] = a[i][col2];
        a[i][col2] = t;
    }
    return *this;
}

```

### Lab3.cpp

```

#include "CharExt.h"

int main()
{
    CharExt a('my new string');
    a.PrintChar();
    a.SpawCols(a.MaxElementsCol(), 0).PrintChar();
    a.SpawCols(a.MinElementsCol(), 0).PrintChar();
    a.PrintChar();

    CharExt b('min and max');
    b.PrintChar();
    b.SpawCols(b.MaxElementsCol(), 0).PrintChar();
    b.SpawCols(b.MinElementsCol(), 0).PrintChar();
    b.PrintChar();

    return 0;
}

```

### Результат выполнения программы

```

D:\БГУИР\2 курс\1 семестр\ООТПИС\Lab\Lab3\Debug\Lab3.exe
my new string
string new my
min and max
min and max
Для продолжения нажмите любую клавишу . . . _

```