|  |  |
| --- | --- |
| для прик эмбл | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждениевысшего профессионального образования"Московский технологический университет"МИРЭА | |
| Факультет информационных технологий (ИТ) | |
| Кафедра практической и прикладной информатики (ППИ) | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКИМ РАБОТАМ** | |
| **по дисциплине** | |
| «Объектно-ориентированное программирование» | |
|  | |
| Выполнил студент группы ИНБО-04-18 | Тарасов Д.И. |
| Принял старший преподаватель | Хлебникова В.Л. |

|  |  |  |
| --- | --- | --- |
| Практические работы выполнены | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |

Москва 2019

Содержание

[Таблица успеваемости 3](#_Toc8073375)

[Практическая работа №1 4](#_Toc8073376)

[Практическая работа №2 6](#_Toc8073377)

[Практическая работа №3 10](#_Toc8073378)

[Практическая работа №4 20](#_Toc8073379)

[Практическая работа №5 24](#_Toc8073380)

[Практическая работа №6 33](#_Toc8073381)

[Практическая работа №7 42](#_Toc8073382)

[Практическая работа №8 51](#_Toc8073383)

# Таблица успеваемости

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №/р. | страница в отчете. | Студент выполнил:  (подпись) | Преподаватель принял:  (подпись) | Балл  (максимально возможный) | Балл (Фактический) |
| 1. |  | Тарасов Д.И. | Хлебникова В.Л. | 2 |  |
| 2. |  | Тарасов Д.И. | Хлебникова В.Л. | 2 |  |
| 3. |  | Тарасов Д.И. | Хлебникова В.Л. | 4 |  |
| 4. |  | Тарасов Д.И. | Хлебникова В.Л. | 6 |  |
| 5. |  | Тарасов Д.И. | Хлебникова В.Л. | 6 |  |
| 6. |  | Тарасов Д.И. | Хлебникова В.Л. | 8 |  |
| 7. |  | Тарасов Д.И. | Хлебникова В.Л. | 8 |  |
| 8. |  | Тарасов Д.И. | Хлебникова В.Л. | 8 |  |
|  |  |  | Итоговый балл: | 44 |  |

# Практическая работа №1

**Задание:**

1.Реализовать программу, которая считывает Ваше имя и

здоровается с Вами.

2.Реализовать программу, производящую различные операции над числами. (сложение\вычитание, целые числа\числа с плавающей запятой)

**Ход выполнения:**

*#include <iostream>*

*#include <string>*

*#include <windows.h>*

*using namespace std;*

*int main()*

*{*

*SetConsoleCP(1251);*

*SetConsoleOutputCP(1251);//корректный вывод русского языка*

*string name;*

*cin >> name;//считывание имени*

*cout<<"Привет, " << name << endl;//вывод приветствия*

*system("pause");*

*return 0;*

*}*

*#include <iostream>*

*using namespace std;*

*int main()*

*{*

*double a,b;*

*cout << "a, b" << endl;*

*cin >> a >> b;//считывание двух чисел*

*cout << "a+b" << endl ;*

*cout << a+b << endl; //сложение*

*cout << "a-b" << endl;*

*cout << a-b << endl;//вычитание*

*system("pause");*

*return 0;*

*}*

**Выводы:**

В ходе выполнения Практической работы №1 были использованы библиотеки iostream, string, windows.h. Были достигнуты результаты:

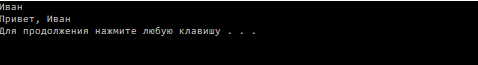


Рис. 1

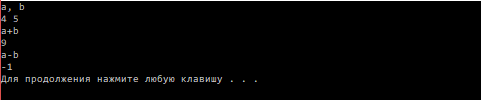


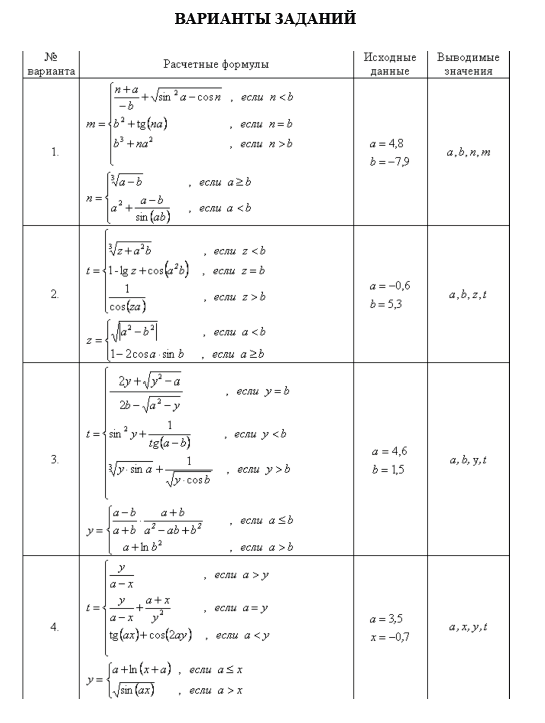
Рис. 2

<https://github.com/dneva/labs/blob/master/prac1.1.txt>

<https://github.com/dneva/labs/blob/master/prac1.2>

# Практическая работа №2

**Задание:**



**Ход выполнения:**

Вариант 1

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main()*

*{*

*double a, b, n, m;*

*cout << "a,b" << endl;*

*cin >> a >> b;*

*if (a >= b)//Ветка 1.1*

*n = pow(a - b, (1.0 / 3));*

*else if (a<b)//Ветка 1.2*

*n = a\*a + (a - b) / sin(a\*b);*

*if (n<b)//Ветка 2.1*

*m = (n + a) / -b + pow((sin(a)\*sin(a) - cos(n)), 1.0 / 2);*

*else if (n == b)//Ветка 2.2*

*m = b\*b + tan(n\*a);*

*else//Ветка 2.3*

*m = pow(b, 3) + n\*a\*a;*

*cout << "m = " << m << endl << "n = " << n << endl;*

*system("pause");*

*return 0;*

*}*

Вариант 2

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main()*

*{*

*double a, b, t, z;*

*cout << "a,b" << endl;*

*cin >> a >> b;*

*if (a >= b)//Ветка 1.1*

*z = 1 - 2 \* cos(a)\*sin(b);*

*else if (a<b)//Ветка 1.2*

*z = pow(abs(a\*a - b\*b), (1.0 / 2));*

*if (z<b)//Ветка 2.1*

*t = pow((z + a\*a\*b), 1.0 / 3);*

*else if (z == b)//Ветка 2.2*

*t = 1 - log(z) + cos(a\*a\*b);*

*else//Ветка 2.3*

*t = 1 / cos(z\*a);*

*cout << "t = " << t << endl << "z = " << z<< endl;*

*system("pause");*

*return 0;*

*}*

Вариант 3

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main()*

*{*

*double a, b, t, y;*

*cout << "a,b" << endl;*

*cin >> a >> b;*

*if (a <= b)//Ветка 1.1*

*y = (a-b)/(a+b)\*(a+b)/(a\*a - a\*b + b\*b);*

*else if (a > b)//Ветка 1.2*

*y = a + log(b\*b);*

*if (y==b)//Ветка 2.1*

*t = (2\*y+sqrt(y\*y-a)/(2\*b-sqrt(a\*a-y)));*

*else if (y < b)//Ветка 2.2*

*t = pow(sin(y),2)+1/tan(a-b);*

*else//Ветка 2.3*

*t = pow(y\*sin(a), 1.0 / 3) + 1 / sqrt(y\*cos(b));*

*cout << "t = " << t << endl << "y = " << y << endl;*

*system("pause");*

*return 0;*

*}*

Вариант 4

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main()*

*{*

*double a, x, t, y;*

*cout << "a,b" << endl;*

*cin >> a >> x;*

*if (a <= x)//Ветка 1.1*

*y = a+log(x+a);*

*else if (a > x)//Ветка 1.2*

*y = sqrt(sin(a\*x\*3.14/180));*

*if (a>y)//Ветка 2.1*

*t = y/(a-x);*

*else if (a==y)//Ветка 2.2*

*t = y/(a-x)+(a+x)/(y\*y);*

*else//Ветка 2.3*

*t = tan(a\*x\*3.14 / 180 + cos(2 \* a\*y\*3.14 / 180));*

*cout << "t = " << t << endl << "y = " << y << endl;*

*system("pause");*

*return 0;*

*}*

**Выводы:**

В ходе выполнения Практической работы №2 были использованы библиотеки iostream, cmath. Были достигнуты результаты:

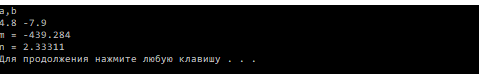


Рис. 3

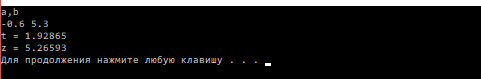


Рис. 4

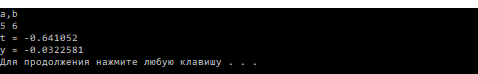


Рис. 5



Рис. 6

<https://github.com/dneva/labs/blob/master/prac2.1>

<https://github.com/dneva/labs/blob/master/prac2.2>

<https://github.com/dneva/labs/blob/master/prac2.3>

<https://github.com/dneva/labs/blob/master/prac2.4>

# Практическая работа №3

**Задание:**

1. Реализовать класс «Книга»
2. Реализовать класс «Группа студентов»
3. Реализовать класс «Автомобиль»
4. Реализовать класс «Вектор»
5. Реализовать класс «Библиотека» (повышенная сложность задания)

**Ход выполнения:**

Вариант 1

|  |
| --- |
| #include <iostream> |
|  | #include<string> |
|  | #include <windows.h> |
|  | using namespace std; |
|  | class Book//определение класса |
|  | { |
|  | private: |
|  | string name;//поле класса для названия |
|  | string author;//поле класса для автора |
|  | int edition;//поле класса для издания |
|  | string publisher;//поле класса для издательство |
|  | double price;//поле класса для цены |
|  | public: |
|  | void define(string n, string a, int e, string pu, double p)//метод класса, изменяющий значения полей |
|  | { |
|  | name = n; |
|  | author = a; |
|  | edition = e; |
|  | publisher = pu; |
|  | price = p; |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Название: " << name << "; Автор: " << author << "; Издание: " << edition << "; Издательсво: " << publisher << "; Цена: " << price << endl; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | setlocale(0,"rus");//вывод на русском |
|  | SetConsoleCP(1251); |
|  | SetConsoleOutputCP(1251);//ввод на русском |
|  | Book b;//определение объекта класса Book |
|  | string n, a, pu; |
|  | int e; |
|  | double p; |
|  | cout << "Введите название" << endl; |
|  | getline(cin,n); |
|  | cout << "Введите автора" << endl; |
|  | getline(cin, a); |
|  | cout << "Введите издательство" << endl; |
|  | getline(cin, pu); |
|  | cout << "Введите издание" << endl; |
|  | cin >> e; |
|  | cout << "Введите цену" << endl; |
|  | cin >> p; |
|  | b.define(n,a,e,pu,p);//вызов метода define() |
|  | b.display();//вызов метода display |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 2

|  |
| --- |
| #include <iostream> |
|  | #include<string> |
|  | //#include <windows.h> |
|  | using namespace std; |
|  | class Student |
|  | { |
|  | private: |
|  | string name;//поле класса для имени |
|  | string surname;//поле класса для фамилии |
|  | string midname;//поле класса для отчества |
|  | int mark[5];//поле класса для оценки |
|  | public: |
|  | void define(string n, string s, string m, int ma[5])//метод класса, изменяющий значения полей |
|  | { |
|  | name = n; |
|  | surname = s; |
|  | midname = m; |
|  | for (int i = 0; i < 5; i++) |
|  | mark[i] = ma[i]; |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Имя: " << name << "; Фамилия: " << surname << "; Отчество: " << midname << "; Оценки: " << endl; |
|  | for (int i = 0; i < 5; i++) cout << mark[i] << " "; |
|  | cout << endl; |
|  | } |
|  | }; |
|  | class Group\_S//определение класса |
|  | { |
|  | private: |
|  | string name;//поле для названия группы |
|  | int number;//поле для числа студентов |
|  | Student \*st;//поле для студентов |
|  | public: |
|  | void define(string n, int num) |
|  | { |
|  | name = n; |
|  | number = num; |
|  | st = new Student[num]; |
|  | string na, s, m; |
|  | int ma[5]; |
|  |  |
|  | for(int i = 0; i<num;i++) |
|  | { |
|  | cin.ignore(); |
|  | cout << "Введите имя студента" << endl; |
|  | getline(cin,na); |
|  | cout << "Введите фамилию студента" << endl; |
|  | getline(cin, s); |
|  | cout << "Введите отчество студента" << endl; |
|  | getline(cin, m); |
|  | cout << "Введите оценки" << endl; |
|  | for (int j = 0; j < 5; j++) cin >> ma[j]; |
|  | st[i].define(na,s,m,ma);//вызов метода define() |
|  | } |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Название: " << name << "; Количество студентов: " << number << endl; |
|  | for (int i = 0; i < number; i++) |
|  | { |
|  | st[i].display(); |
|  | cout << endl; |
|  | } |
|  | } |
|  | void delete\_st(void)//метод удаление массива |
|  | { |
|  | delete []st; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | setlocale(0,"rus");//вывод на русском |
|  | // SetConsoleCP(1251); |
|  | // SetConsoleOutputCP(1251);//ввод на русском |
|  | Group\_S g;//определение объекта класса Gropup\_S |
|  | string n; |
|  | int num; |
|  | cout << "Введите название группы" << endl; |
|  | getline(cin, n); |
|  | cout << "Введите количество студентов" << endl; |
|  | cin >> num; |
|  | g.define(n,num);//вызов метода define() |
|  | g.display();//вызов метода display |
|  | g.delete\_st(); |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 3

|  |
| --- |
| #include <iostream> |
|  | #include<string> |
|  | #include <windows.h> |
|  | using namespace std; |
|  | class Auto//определение класса |
|  | { |
|  | private: |
|  | string brand;//поле класса для марки |
|  | string model;//поле класса для модели |
|  | string color;//поле класса для цвета |
|  | string number;//поле класса для номера |
|  | public: |
|  | void define(string b, string m, string c, string n)//метод класса, изменяющий значения полей |
|  | { |
|  | brand = b; |
|  | model = m; |
|  | color = c; |
|  | number = n; |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Марка: " << brand << "; Модель: " << model << "; Цвет: " << color << "; Номер: " << number<< endl; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | setlocale(0,"rus");//вывод на русском |
|  | SetConsoleCP(1251); |
|  | SetConsoleOutputCP(1251);//ввод на русском |
|  | Auto a;//определение объекта класса Auto |
|  | string b,m,c,n; |
|  | cout << "Введите марку автомобиля" << endl; |
|  | getline(cin,b); |
|  | cout << "Введите модель автомобиля" << endl; |
|  | getline(cin, m); |
|  | cout << "Введите цвет автомобиля" << endl; |
|  | getline(cin, c); |
|  | cout << "Введите номер автомобиля" << endl; |
|  | getline(cin, n); |
|  | a.define(b,m,c,n);//вызов метода define() |
|  | a.display();//вызов метода display |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 4

|  |
| --- |
| #include <iostream> |
|  | #include<string> |
|  | #include <windows.h> |
|  | using namespace std; |
|  | class Point//определение класса точки |
|  | { |
|  | private: |
|  | double x;//поле класса x |
|  | double y;//поле класса y |
|  | double z;//поле класса z |
|  | public: |
|  | void set(double n\_x, double n\_y, double n\_z)//метод класса, изменяющий значения полей |
|  | { |
|  | x = n\_x; |
|  | y = n\_y; |
|  | z = n\_z; |
|  | } |
|  | double get\_x(void)//метод класса, отображающий x |
|  | { |
|  | return x; |
|  | } |
|  | double get\_y(void)//метод класса, отображающий y |
|  | { |
|  | return y; |
|  | } |
|  | double get\_z(void)//метод класса, отображающий z |
|  | { |
|  | return z; |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout <<"x = " << x << "; y = " << y << "; z = " << z << endl; |
|  | } |
|  | }; |
|  | class Vector//определение класса |
|  | { |
|  | private: |
|  | Point begin;//поле класса для координаты начала |
|  | Point end;//поле класса для координаты конца |
|  | public: |
|  | void define(Point b, Point e)//метод класса, изменяющий значения полей |
|  | { |
|  | begin.set(b.get\_x(),b.get\_y(),b.get\_z()); |
|  | end.set(e.get\_x(), e.get\_y(), e.get\_z()); |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Координата начала: " << endl; |
|  | begin.display(); |
|  | cout << "Координата конца: " << endl; |
|  | end.display(); |
|  |  |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | setlocale(0,"rus");//вывод на русском |
|  | SetConsoleCP(1251); |
|  | SetConsoleOutputCP(1251);//ввод на русском |
|  | Vector v;//определение объекта класса Vector |
|  | int x, y, z; |
|  | Point b,e; |
|  | cout << "Введите координату начала (x,y,z)" << endl; |
|  | cin >> x >> y >> z; |
|  | b.set(x, y, z); |
|  | cout << "Введите координату конца (x,y,z)" << endl; |
|  | cin >> x >> y >> z; |
|  | e.set(x, y, z); |
|  |  |
|  | v.define(b,e);//вызов метода define() |
|  | v.display();//вызов метода display |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 5

|  |
| --- |
| #include <iostream> |
|  | #include<string> |
|  | //#include <windows.h> |
|  | using namespace std; |
|  | class Book//определение класса |
|  | { |
|  | private: |
|  | string name;//поле класса для названия |
|  | string author;//поле класса для автора |
|  | int edition;//поле класса для издания |
|  | string publisher;//поле класса для издательство |
|  | double price;//поле класса для цены |
|  | public: |
|  | void define(string n, string a, int e, string pu, double p)//метод класса, изменяющий значения полей |
|  | { |
|  | name = n; |
|  | author = a; |
|  | edition = e; |
|  | publisher = pu; |
|  | price = p; |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Название: " << name << "; Автор: " << author << "; Издание: " << edition << "; Издательсво: " << publisher << "; Цена: " << price << endl; |
|  | } |
|  | }; |
|  | class Library//определение класса |
|  | { |
|  | private: |
|  | string name;//поле класса для названия |
|  | string address;//поле класса для адреса |
|  | int number;//поле класса для колиства книг |
|  | Book \*books;//поле класса для книг |
|  | public: |
|  | void define(string n, string a, int num)//метод класса, изменяющий значения полей |
|  | { |
|  | name = n; |
|  | address = a; |
|  | number = num; |
|  | books = new Book[num]; |
|  | string na, au, pu; |
|  | int e; |
|  | double p; |
|  | for (int i = 0; i < num; i++) |
|  | { |
|  | cin.ignore(); |
|  | cout << "Введите название" << endl; |
|  | getline(cin, na); |
|  | cout << "Введите автора" << endl; |
|  | getline(cin, au); |
|  | cout << "Введите издательство" << endl; |
|  | getline(cin, pu); |
|  | cout << "Введите издание" << endl; |
|  | cin >> e; |
|  | cout << "Введите цену" << endl; |
|  | cin >> p; |
|  | books[i].define(na, au, e, pu, p); |
|  | } |
|  |  |
|  | } |
|  | void display(void)//метод класса, отображающий значения полей |
|  | { |
|  | cout << "Название: " << name << "; Адрес: " << address << "; Количество книг: " << number << endl; |
|  | for (int i = 0; i < number; i++) |
|  | { |
|  | books[i].display(); |
|  | cout << endl; |
|  | } |
|  | } |
|  | void delete\_b(void)//метод удаление массива |
|  | { |
|  | delete []books; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | setlocale(0,"rus");//вывод на русском |
|  | //SetConsoleCP(1251); |
|  | //SetConsoleOutputCP(1251);//ввод на русском |
|  | Library l;//определение объекта класса Library |
|  | string n, a; |
|  | int num; |
|  | cout << "Введите название" << endl; |
|  | getline(cin, n); |
|  | cout << "Введите адрес" << endl; |
|  | getline(cin, a); |
|  | cout << "Введите количество книг" << endl; |
|  | cin >> num; |
|  |  |
|  | l.define(n,a,num);//вызов метода define() |
|  | l.display();//вызов метода display |
|  | l.delete\_b(); |
|  | system("pause"); |
|  | return 0; |
|  | } |

**Выводы:**

В ходе выполнения Практической работы №3 были использованы библиотеки iostream, string. В классах реализованы методы для присвоения значений полям и вывода этих значений. Были достигнуты результаты:

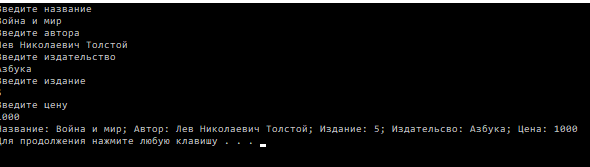


Рис. 7

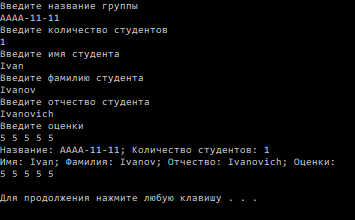


Рис. 8

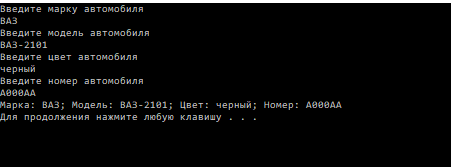


Рис. 9

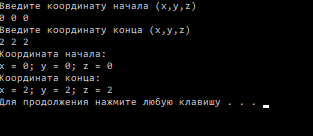


Рис. 10

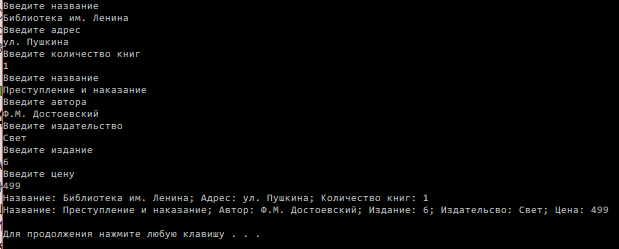


Рис. 11

<https://github.com/dneva/prac-lab/blob/master/3/prac3.1>

<https://github.com/dneva/prac-lab/blob/master/3/prac3.2>

<https://github.com/dneva/prac-lab/blob/master/3/prac3.3>

<https://github.com/dneva/prac-lab/blob/master/3/prac3.4>

<https://github.com/dneva/prac-lab/blob/master/3/prac3.5>

# Практическая работа №4

**Задание:**

1. Объявите указатель на массив типа double и предложите

пользователю выбрать его размер. Далее напишите четыре

функции: первая должна выделить память для массива, вторая —

заполнить ячейки данными, третья — показать данные на экран,

четвертая — освободить занимаемую память. Программа должна

предлагать пользователю продолжать работу (создавать новые

динамические массивы) или выйти из программы.

1. Объявите указатель на массив типа int и выделите память память

для 12-ти элементов. Необходимо написать функцию, которая

поменяет значения четных и нечетных ячеек массива.

1. Объявить и заполнить двумерный динамический массив

случайными числами от 10 до 50. Показать его на экран. Для

заполнения и показа на экран написать отдельные функции.

(подсказка: функции должны принимать три параметра — указатель на динамический массив, количество строк, количество столбцов). Количество строк и столбцов выбирает пользователь.

**Ход выполнения:**

Вариант 1

#include <iostream>

#include <ctime>

#include <cmath>

using namespace std;

int main()

{

int sw = 1;

int size;

while (sw != 0)

{

switch (sw)

{

case 1:

{

double \*a;//объявление указателя на массив

cout << "Enter the dimension of the array: ";

cin >> size;//ввод размера массива

a = new double[size];//выделение памяти под массив

srand(time(0));

for (int i = 0; i < size; i++)

{

a[i] = (double)(rand() % 1000 - 50) / 100;//заполнение массива элементами

cout << a[i] << " ";//вывод элементов массива

}

cout << endl;

delete[] a;//освобождение памяти

}

break;

case 0://выход из цикла

break;

default:

cout << "Repeat input" << endl;

break;

}

cout << "1 - continue, 0 - exit" << endl;//предложение продолжить работу

cin >> sw;

}

system("pause");

return 0;

}

Вариант 2

|  |
| --- |
| #include <iostream> |
|  | using namespace std; |
|  | void swap(int &a, int &b)//функция для перестановки местами значений переменных |
|  | { |
|  | int buff; |
|  | buff = a; |
|  | a = b; |
|  | b = buff; |
|  | } |
|  | int main() |
|  | { |
|  | double \*a = new double[12];//объявление указателя на массив и выделение памяти под массив |
|  | cout << "Input 12 items" << endl; |
|  | for (int i = 0; i < 12; i++) cin >> a[i]; |
|  | for (int i = 0; i < 12; i+=2) |
|  | { |
|  | swap(a[i], a[i+1]);//перестановка элементов |
|  | } |
|  | cout << "New array: " << endl; |
|  | for (int i = 0; i < 12; i++) cout << a[i] << " ";//вывод измененного массива |
|  | cout << endl; |
|  | delete[] a;//освобождение памяти |
|  |  |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 3

|  |
| --- |
| #include <iostream> |
|  | #include <ctime> |
|  | using namespace std; |
|  | void fill(int \*\*a, int i, int j)//функция заполнения массива |
|  | { |
|  | srand(time(0)); |
|  | for(int l=0;l<i;l++) |
|  | for (int c = 0; c < j; c++) |
|  | { |
|  | a[l][c] = 10+rand()%40; |
|  | } |
|  | } |
|  | void show(int \*\*a, int i, int j)//функция вывода массива |
|  | { |
|  | for (int l = 0; l < i; l++) |
|  | { |
|  | for (int c = 0; c < j; c++) |
|  | { |
|  | cout << a[l][c] << " "; |
|  | } |
|  | cout << endl; |
|  | } |
|  |  |
|  | } |
|  | int main() |
|  | { |
|  | int i, j; |
|  | cout << "Input the number of lines" << endl;//ввод строк |
|  | cin >> i; |
|  | cout << "Input the number of columns" << endl;//ввод столбцов |
|  | cin >> j; |
|  | int \*\*a = new int\*[i];//создание динамического двумерного массива |
|  | for (int c = 0; c < i; c++) |
|  | { |
|  | a[c] = new int[j]; |
|  | } |
|  | fill(a,i,j); |
|  | show(a, i, j); |
|  | system("pause"); |
|  | return 0; |
|  | } |

**Выводы:**

В ходе выполнения Практической работы №4 были использованы библиотеки iostream, cmath, ctime. Были реализованы функции перестановки, заполнения, вывода массива. Были достигнуты результаты:

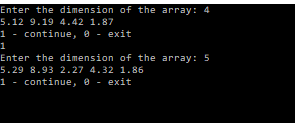


Рис. 12

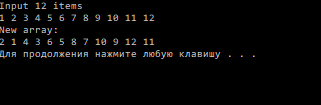


Рис. 13

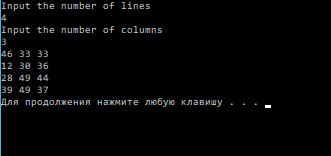


Рис. 14

<https://github.com/dneva/prac-lab/blob/master/4/prac4.1>

<https://github.com/dneva/prac-lab/blob/master/4/prac4.2>

<https://github.com/dneva/prac-lab/blob/master/4/prac4.3>

# Практическая работа №5

**Задание:**

1. Реализовать сортировку пузырьком для целых чисел, а затем

перегрузить её для дробных.

1. Реализовать сортировку выбором для целых чисел, а затем

перегрузить её для дробных.

1. Реализовать сортировку вставками для целых чисел, а затем

перегрузить её для дробных.

1. Реализовать программу-калькулятор, работающий с различными

типами данных

**Ход выполнения:**

Вариант 1

|  |
| --- |
| include <iostream> |
|  | #include <ctime> |
|  | #include <cmath> |
|  | using namespace std; |
|  | void bubleSort(int \*a, int size)//сортировка пузырьком для целых чисел |
|  | { |
|  | int buff; |
|  | for (int i = 0; i < size; i++) { |
|  | for (int j = 0; j < size - 1; j++) { |
|  | if (a[j] > a[j + 1]) { |
|  | buff = a[j]; // |
|  | a[j] = a[j + 1]; // меняем местами |
|  | a[j + 1] = buff; // значения элементов |
|  | } |
|  | } |
|  | } |
|  | } |
|  | void bubleSort(double \*a, int size)//сортировка пузырьком для дробных чисел |
|  | { |
|  | double buff; |
|  | for (int i = 0; i < size; i++) { |
|  | for (int j = 0; j < size - 1; j++) { |
|  | if (a[j] > a[j + 1]) { |
|  | buff = a[j]; |
|  | a[j] = a[j + 1]; // меняем местами |
|  | a[j + 1] = buff; // значения элементов |
|  | } |
|  | } |
|  | } |
|  | } |
|  | void show(int \*a, int size)//функция вывода массива для целых чисел |
|  | { |
|  | cout << "Int array:" << endl; |
|  | for (int i = 0; i < size; i++) cout << a[i] << " "; |
|  | cout << endl; |
|  | } |
|  | void show(double \*a, int size)//функция вывода массива для дробных чисел |
|  | { |
|  | cout << "Double array:" << endl; |
|  | for (int i = 0; i < size; i++) cout << a[i] << " "; |
|  | cout << endl; |
|  | } |
|  | int main() |
|  | { |
|  | int a[10]; |
|  | double b[10]; |
|  | srand(time(0)); |
|  | for (int i = 0; i < 10; i++)//заполнение массивов |
|  | { |
|  | a[i]= 10 + rand() % 40; |
|  | b[i]= (double)(rand() % 1000 - 50) / 100; |
|  | } |
|  | show(a, 10); |
|  | show(b, 10); |
|  | bubleSort(a,10); |
|  | bubleSort(b, 10); |
|  | show(a, 10); |
|  | show(b, 10); |
|  | system("pause"); |
|  | return 0; |
|  |  |
|  | } |

Вариант 2

|  |
| --- |
| #include <iostream> |
|  | #include <ctime> |
|  | #include <cmath> |
|  | using namespace std; |
|  | void choicesSort(int \*a, int size)//сортировка выбором для целых чисел |
|  | { |
|  | int buff; |
|  | for (int i = 0; i < size; i++) |
|  | { |
|  | buff = a[0]; |
|  | for (int j = i + 1; j <size; j++) |
|  | { |
|  | if (a[i] > a[j]) |
|  | { |
|  | buff = a[i]; |
|  | a[i] = a[j]; |
|  | a[j] = buff; |
|  | } |
|  | } |
|  | } |
|  | } |
|  | void choicesSort(double \*a, int size)//сортировка выбором для дробных чисел |
|  | { |
|  | double buff; |
|  | for (int i = 0; i < size; i++) |
|  | { |
|  | buff = a[0]; |
|  | for (int j = i + 1; j <size; j++) |
|  | { |
|  | if (a[i] > a[j]) |
|  | { |
|  | buff = a[i]; |
|  | a[i] = a[j]; |
|  | a[j] = buff; |
|  | } |
|  | } |
|  | } |
|  | } |
|  | void show(int \*a, int size)//функция вывода массива для целых чисел |
|  | { |
|  | cout << "Int array:" << endl; |
|  | for (int i = 0; i < size; i++) cout << a[i] << " "; |
|  | cout << endl; |
|  | } |
|  | void show(double \*a, int size)//функция вывода массива для дробных чисел |
|  | { |
|  | cout << "Double array:" << endl; |
|  | for (int i = 0; i < size; i++) cout << a[i] << " "; |
|  | cout << endl; |
|  | } |
|  | int main() |
|  | { |
|  | int a[10]; |
|  | double b[10]; |
|  | srand(time(0)); |
|  | for (int i = 0; i < 10; i++)//заполнение массивов |
|  | { |
|  | a[i]= 10 + rand() % 40; |
|  | b[i]= (double)(rand() % 1000 - 50) / 100; |
|  | } |
|  | show(a, 10); |
|  | show(b, 10); |
|  | choicesSort(a,10); |
|  | choicesSort(b, 10); |
|  | show(a, 10); |
|  | show(b, 10); |
|  | system("pause"); |
|  | return 0; |
|  |  |
|  | } |

Вариант 3

|  |
| --- |
| #include <iostream> |
|  | #include <ctime> |
|  | #include <cmath> |
|  | using namespace std; |
|  | void insertSort(int \*a, int size)//сортировка вставками для целых чисел |
|  | { |
|  | int key, buff; |
|  | for (int i = 0; i < size - 1; i++) |
|  | { |
|  | key = i + 1; |
|  | buff = a[key]; |
|  | for (int j = i + 1; j > 0; j--) |
|  | { |
|  | if (buff < a[j - 1]) |
|  | { |
|  | a[j] = a[j - 1]; |
|  | key = j - 1; |
|  | } |
|  | } |
|  | a[key] = buff; |
|  | } |
|  | } |
|  | void insertSort(double \*a, int size)//сортировка вставками для дробных чисел |
|  | { |
|  | int key; |
|  | double buff; |
|  | for (int i = 0; i < size - 1; i++) |
|  | { |
|  | key = i + 1; |
|  | buff = a[key]; |
|  | for (int j = i + 1; j > 0; j--) |
|  | { |
|  | if (buff < a[j - 1]) |
|  | { |
|  | a[j] = a[j - 1]; |
|  | key = j - 1; |
|  | } |
|  | } |
|  | a[key] = buff; |
|  | } |
|  | } |
|  | void show(int \*a, int size)//функция вывода массива для целых чисел |
|  | { |
|  | cout << "Int array:" << endl; |
|  | for (int i = 0; i < size; i++) cout << a[i] << " "; |
|  | cout << endl; |
|  | } |
|  | void show(double \*a, int size)//функция вывода массива для дробных чисел |
|  | { |
|  | cout << "Double array:" << endl; |
|  | for (int i = 0; i < size; i++) cout << a[i] << " "; |
|  | cout << endl; |
|  | } |
|  | int main() |
|  | { |
|  | int a[10]; |
|  | double b[10]; |
|  | srand(time(0)); |
|  | for (int i = 0; i < 10; i++)//заполнение массивов |
|  | { |
|  | a[i]= 10 + rand() % 40; |
|  | b[i]= (double)(rand() % 1000 - 50) / 100; |
|  | } |
|  | show(a, 10); |
|  | show(b, 10); |
|  | insertSort(a,10); |
|  | insertSort(b, 10); |
|  | show(a, 10); |
|  | show(b, 10); |
|  | system("pause"); |
|  | return 0; |
|  |  |
|  | } |

Вариант 4

|  |
| --- |
| #include <iostream> |
|  | #include <ctime> |
|  | #include <cmath> |
|  | using namespace std; |
|  | int sum(int a,int b)//сложение целых чисел |
|  | { |
|  | return a + b; |
|  | } |
|  | double sum(double a, double b)//сложение дробных чисел |
|  | { |
|  | return a + b; |
|  | } |
|  | int min(int a, int b)//вычитание целых чисел |
|  | { |
|  | return (a - b); |
|  | } |
|  | double min(double a, double b)//вычитание дробных чисел |
|  | { |
|  | return a - b; |
|  | } |
|  | int multiply(int a, int b)//умножение целых чисел |
|  | { |
|  | return a \* b; |
|  | } |
|  | double multiply(double a, double b)//умножение дробных чисел |
|  | { |
|  | return a \* b; |
|  | } |
|  | int divide(int a, int b)//деление целых чисел |
|  | { |
|  | if (b != 0) |
|  | return a / b; |
|  | else |
|  | { |
|  | cout << "Division by zero" << endl; |
|  | return -1; |
|  | } |
|  |  |
|  | } |
|  | double divide(double a, double b)//деление дробных чисел |
|  | { |
|  | if (b != 0) |
|  | return a / b; |
|  | else |
|  | { |
|  | cout << "Division by zero" << endl; |
|  | return -1; |
|  | } |
|  | } |
|  | int main() |
|  | { |
|  | int type; |
|  | double a, b; |
|  | int sw = 1; |
|  | cout << "1 - sum, 2 - minus, 3 - multiply, 4 - divide, 0 - exit" << endl;//выбор |
|  | cin >> sw; |
|  | while (sw != 0) |
|  | { |
|  | switch (sw) |
|  | { |
|  | case 1://сложение |
|  | { |
|  | cout << "Input a,b" << endl; |
|  | cin >> a >> b; |
|  | cout << "INT: " << sum((int)a, (int)b) << endl; |
|  | cout << "DOUBLE: " << sum(a, b) << endl; |
|  | } |
|  | break; |
|  | case 2://вычитание |
|  | { |
|  | cout << "Input a,b" << endl; |
|  | cin >> a >> b; |
|  | cout << "INT: " << min((int)a, (int)b) << endl; |
|  | cout << "DOUBLE: " << min(a, b) << endl; |
|  | } |
|  | break; |
|  | case 3://умножение |
|  | { |
|  | cout << "Input a,b" << endl; |
|  | cin >> a >> b; |
|  | cout << "INT: " << multiply((int)a, (int)b) << endl; |
|  | cout << "DOUBLE: " << multiply(a, b) << endl; |
|  | } |
|  | break; |
|  | case 4://деление |
|  | { |
|  | cout << "Input a,b" << endl; |
|  | cin >> a >> b; |
|  | cout << "INT: " << divide((int)a, (int)b) << endl; |
|  | cout << "DOUBLE: " << divide(a, b) << endl; |
|  | } |
|  | break; |
|  | case 0://выход из цикла |
|  | break; |
|  | default: |
|  | cout << "Repeat input" << endl; |
|  | break; |
|  | } |
|  | cout << "1 - sum, 2 - minus, 3 - multiply, 4 - divide, 0 - exit" << endl; |
|  | cin >> sw; |
|  | } |
|  | system("pause"); |
|  | return 0; |
|  |  |
|  | } |

**Выводы:**

В ходе выполнения Практической работы №5 были использованы библиотеки iostream, cmath, ctime. Были реализованы функции сортировок массива и функции работы с целыми и дробными числами. Были достигнуты результаты:

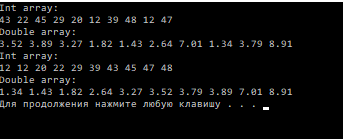


Рис. 15

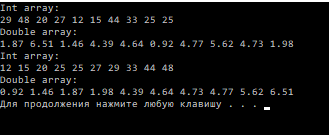


Рис. 16

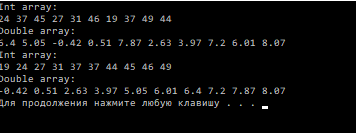


Рис. 17

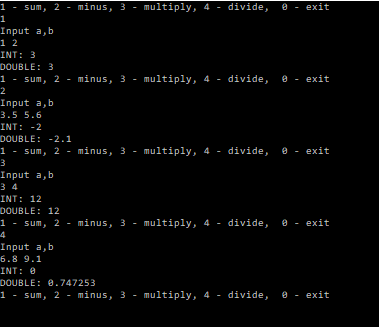


Рис. 18

<https://github.com/dneva/prac-lab/blob/master/5/prac5.1>

<https://github.com/dneva/prac-lab/blob/master/5/prac5.2>

<https://github.com/dneva/prac-lab/blob/master/5/prac5.3>

<https://github.com/dneva/prac-lab/blob/master/5/prac5.4>

# Практическая работа №6

**Задание:**

1. Написать иерархию классов, описывающих имущество

налогоплательщиков. Она должна состоять из абстракт- ного

базового класса Property и производных от него классов Appartment, Car и CountryHouse. Базовый класс должен иметь поле worth(стоимость), конструктор с одним параметром, заполняющий

это поле, и чисто виртуальный метод рассчета налога,

переопределенный в каждом из производных классов. Налог на

квартиру вычисляется как 1/1000 ее стоимости, на машину — 1/200,

на дачу — 1/500. Также, каждый производный класс должен иметь

конструктор с одним параметром, передающий свой параметр

конструктору базового класса. В функции main завести массив из 7

указателей на Property и заполнить его указателями на

динамические объекты производных классов (первые 3 —

Appartment, следующие 2 — Car и последние 2 — CountryHouse).

Вывести на экран величину налога для всех 7 объектов. Не забудьте

также уничтожить динамические объекты перед завершением

программы.

1. Написать набор классов, представляющий выражения. В этом

наборе должен быть один абстрактный базовый тип, а также набор

производных от него типов по видам выражений (константа,

переменная, сумма, разность, произведение, частное, sin, cos, exp,

ln). У каждого из классов должны быть следующие виртуальные

функции: напечатать выражение (без параметров), вычислить

выражение (параметр — значение переменной, результат —

значение выражения), вернуть производную выражения (без

параметров), создать копию выражения (тоже без параметров).

**Ход выполнения:**

Вариант 1

|  |
| --- |
| #include<iostream> |
|  | using namespace std; |
|  | class Property//класс "собственность" |
|  | { |
|  | protected: |
|  | double worth;//стоимость |
|  | public: |
|  | Property(double w): worth(w) {}; |
|  | virtual ~Property() {}; |
|  | virtual double tax() = 0;//виртуальная функция налогов |
|  | }; |
|  | class Appartment : public Property// класс "квартира" |
|  | { |
|  | public: |
|  | Appartment(double w) : Property(w) {}; |
|  | double tax() |
|  | { |
|  | return worth/1000; |
|  | } |
|  | }; |
|  | class Car : public Property// класс "машина" |
|  | { |
|  | public: |
|  | Car(double w) : Property(w) {}; |
|  | double tax() |
|  | { |
|  | return worth / 200; |
|  | } |
|  | }; |
|  | class CountryHouse : public Property// класс "дача" |
|  | { |
|  | public: |
|  | CountryHouse(double w) : Property(w) {}; |
|  | double tax() |
|  | { |
|  | return worth / 500; |
|  | } |
|  |  |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | Property \*properties[7];//массив указателей на Property |
|  | double w; |
|  | for (int i = 0; i < 7; i++) |
|  | { |
|  | cout << "Input worth" << endl; |
|  | cin >> w; |
|  | if (i < 3)// первые 2 - класс "квартира" |
|  | properties[i] = new Appartment(w); |
|  | else if (i == 3 || i == 4)// следующие 2 - класс "машина" |
|  | properties[i] = new Car(w); |
|  | else//последние 2 - класс "дача" |
|  | properties[i] = new CountryHouse(w); |
|  | } |
|  | cout << "Taxes:" << endl; |
|  | for (int i = 0; i < 7; i++) cout << (properties[i]->tax()) << endl;//вывод налогов |
|  | for (int i = 0; i < 7; i++) delete properties[i];//удаление массива |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 2

|  |
| --- |
| #include<iostream> |
|  | using namespace std; |
|  | class Expression// абстрактный класс "выражение" |
|  | { |
|  | protected: |
|  | double a; |
|  | public: |
|  | Expression(double b = 0) : a(b) {};//конструктор |
|  | virtual void print() = 0;//виртуальная функция для печати выражения |
|  | virtual double result(double b) = 0;// виртуальная функция результата |
|  | virtual double derivative() = 0;// виртуальная функция производной |
|  | virtual void copy(double b) = 0;// виртуальная функция для копирования выражения |
|  | virtual ~Expression() {}; |
|  | }; |
|  | class Cons : Expression// класс "константа" |
|  | { |
|  | public: |
|  | Cons(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Constant:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return a; |
|  | } |
|  | double derivative() |
|  | { |
|  | return 0; |
|  | } |
|  | void copy(double b) |
|  | { |
|  |  |
|  | } |
|  | }; |
|  | class Var : Expression// класс "переменная" |
|  | { |
|  | public: |
|  | Var(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Variable:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return b; |
|  | } |
|  | double derivative() |
|  | { |
|  | return 1; |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = b; |
|  | } |
|  | }; |
|  | class Sum : Expression// класс "сумма" |
|  | { |
|  | public: |
|  | Sum(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Sum:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return a+b; |
|  | } |
|  | double derivative() |
|  | { |
|  | return 1; |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = a + b; |
|  | } |
|  | }; |
|  | class Dif : Expression// класс "разность" |
|  | { |
|  | public: |
|  | Dif(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Difference:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return a-b; |
|  | } |
|  | double derivative() |
|  | { |
|  | return 1; |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = a - b; |
|  | } |
|  | }; |
|  | class Mult : Expression// класс "произведение" |
|  | { |
|  | public: |
|  | Mult(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Multiply:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return a \* b; |
|  | } |
|  | double derivative() |
|  | { |
|  | return 1; |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = a \* b; |
|  | } |
|  | }; |
|  | class Pri : Expression// класс "частное" |
|  | { |
|  | public: |
|  | Pri(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Quotient:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | if (b!=0) |
|  | return a / b; |
|  | else return 0; |
|  | } |
|  | double derivative() |
|  | { |
|  | return 1; |
|  | } |
|  | void copy(double b) |
|  | { |
|  | if (b != 0) |
|  | a = a / b; |
|  | else a = 0; |
|  | } |
|  | }; |
|  | class Sin : Expression// класс "синус" |
|  | { |
|  | public: |
|  | Sin(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Sinus:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return sin(a); |
|  | } |
|  | double derivative() |
|  | { |
|  | return cos(a); |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = b; |
|  | } |
|  | }; |
|  | class Cos : Expression// класс "косинус" |
|  | { |
|  | public: |
|  | Cos(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Cosinus:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return cos(b); |
|  | } |
|  | double derivative() |
|  | { |
|  | return -sin(a); |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = b; |
|  | } |
|  | }; |
|  | class Exp : Expression// класс "экспонента" |
|  | { |
|  | public: |
|  | Exp(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Exponent:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return exp(b); |
|  | } |
|  | double derivative() |
|  | { |
|  | return exp(a); |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = b; |
|  | } |
|  | }; |
|  | class Ln : Expression// класс "логарифм" |
|  | { |
|  | public: |
|  | Ln(double b = 0) : Expression(b) {}; |
|  | void print() |
|  | { |
|  | cout << "Logarithm:" << endl; |
|  | } |
|  | double result(double b) |
|  | { |
|  | return log(b); |
|  | } |
|  | double derivative() |
|  | { |
|  | return 1/a; |
|  | } |
|  | void copy(double b) |
|  | { |
|  | a = b; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | double a, b; |
|  | cout << "Input a, b" << endl; |
|  | cin >> a >> b; |
|  | Cons cnst = a; |
|  | cnst.print(); |
|  | cout << "Result: " << cnst.result(b) << " Derivative: " << cnst.derivative() << endl; |
|  | Var v = a; |
|  | v.print(); |
|  | cout << "Result: " << v.result(b) << " Derivative: " << v.derivative() << endl; |
|  | Sum s = a; |
|  | s.print(); |
|  | cout << "Result: " << s.result(b) << " Derivative: " << s.derivative() << endl; |
|  | Dif d = a; |
|  | d.print(); |
|  | cout << "Result: " << d.result(b) << " Derivative: " << d.derivative() << endl; |
|  | Mult m = a; |
|  | m.print(); |
|  | cout << "Result: " << m.result(b) << " Derivative: " << m.derivative() << endl; |
|  | Pri p = a; |
|  | p.print(); |
|  | cout << "Result: " << p.result(b) << " Derivative: " << p.derivative() << endl; |
|  | Sin sn = a; |
|  | sn.print(); |
|  | cout << "Result: " << sn.result(b) << " Derivative: " << sn.derivative() << endl; |
|  | Cos cs = a; |
|  | cs.print(); |
|  | cout << "Result: " << cs.result(b) << " Derivative: " << cs.derivative() << endl; |
|  | Exp e = a; |
|  | e.print(); |
|  | cout << "Result: " << e.result(b) << " Derivative: " << e.derivative() << endl; |
|  | Ln l = a; |
|  | l.print(); |
|  | cout << "Result: " << l.result(b) << " Derivative: " << l.derivative() << endl; |
|  | system("pause"); |
|  | return 0; |
|  | } |

**Выводы:**

В ходе выполнения Практической работы №6 были использованы библиотека iostream. Были реализованы методы, возвращающие налоги на собственность и методы для работы с классами математических выражений. Были достигнуты результаты:

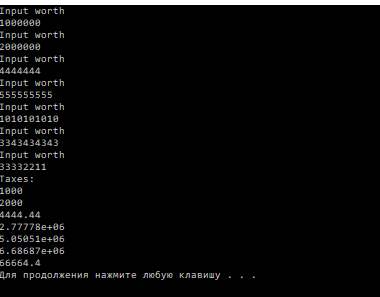


Рис. 19

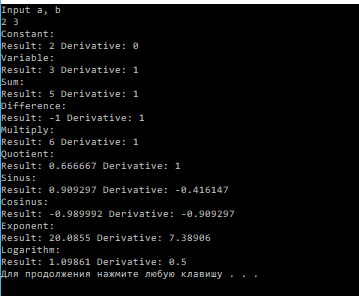


Рис. 20

<https://github.com/dneva/prac-lab/blob/master/6/prac6.1>

<https://github.com/dneva/prac-lab/blob/master/6/prac6.2>

# Практическая работа №7

**Задание:**

1. Реализовать абстрактный класс «Животное» и путём

наследования от него получить классы «Кошка», «Собака»,«Попугай».

1. Реализовать абстрактный класс «Фигура» и путём наследования

от него получить абстрактный класс «Четырёхугольник», и затем

путём наследования получить классы «Ромб», «Прямоугольник».

1. Реализовать абстрактный класс «Транспортное средство» и

путём наследования от него получить классы «Автомобиль», «Автобус», «Велосипед».

1. Реализовать абстрактные классы «Экран» и «Клавиатура», путём

наследования от них получить классы «Ноутбук», «Телефон»,

«Стационарный компьютер».

**Ход выполнения:**

Вариант 1

|  |
| --- |
| #include<iostream> |
|  | #include<string> |
|  | using namespace std; |
|  | class Animal// абстрактный класс "животное" |
|  | { |
|  | protected: |
|  | string name; |
|  | public: |
|  | Animal(string n) : name(n) {}; |
|  | virtual void voice() = 0;//виртуальная функция "подать голос" |
|  | }; |
|  | class Cat : Animal// класс "кошка" |
|  | { |
|  | public: |
|  | Cat(string n) : Animal(n) {}; |
|  | void voice() |
|  | { |
|  | cout << name << " - meow!" << endl; |
|  | } |
|  | }; |
|  | class Dog : Animal// класс "собака" |
|  | { |
|  | public: |
|  | Dog(string n) : Animal(n) {}; |
|  | void voice() |
|  | { |
|  | cout << name << " - woof!" << endl; |
|  | } |
|  | }; |
|  | class Parrot : Animal// класс "попугай" |
|  | { |
|  | private: |
|  | string v; |
|  | public: |
|  | Parrot(string n) : Animal(n) { v = "tweet!"; }; |
|  | void teach(string s)// функция "научить попугая" |
|  | { |
|  | v = s; |
|  | } |
|  | void voice() |
|  | { |
|  | cout << name << " - " << v << endl; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | string s; |
|  | cout << "Input cat's name" << endl; |
|  | cin >> s; |
|  | Cat c = s; |
|  | cout << "Input dog's name" << endl; |
|  | cin >> s; |
|  | Dog d = s; |
|  | cout << "Input parrot's name" << endl; |
|  | cin >> s; |
|  | Parrot p = s; |
|  | cout << "Teach parrot" << endl; |
|  | cin >> s; |
|  | p.teach(s); |
|  | c.voice(); |
|  | d.voice(); |
|  | p.voice(); |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 2

|  |
| --- |
| #include<iostream> |
|  | #include<string> |
|  | using namespace std; |
|  | class Shape// абстрактный класс "фигура" |
|  | { |
|  | protected: |
|  | double s; |
|  | public: |
|  | virtual double square() = 0; |
|  | }; |
|  | class Quadrilateral : Shape // абстрактный класс "четырехугольник" |
|  | { |
|  | protected: |
|  | double a, b; |
|  | public: |
|  | Quadrilateral(double a1, double a2) |
|  | { |
|  | a = a1; |
|  | b = a2; |
|  | }; |
|  | }; |
|  | class Rhombus : Quadrilateral// класс "ромб" |
|  | { |
|  | private: |
|  | double sn; |
|  | public: |
|  | Rhombus(double a1, double a2, double s) : Quadrilateral(a1, a2) { sn = s; }; |
|  | double square() |
|  | { |
|  | return a\*b\*sin(sn\*3.14 / 180); |
|  | } |
|  | }; |
|  | class Rectangle : Quadrilateral// класс "прямоугольник" |
|  | { |
|  | public: |
|  | Rectangle(double a1, double a2) : Quadrilateral(a1, a2) {}; |
|  | double square() |
|  | { |
|  | return a\*b; |
|  | } |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | double a, b; |
|  | cout << "Enter the side and angle of the rhombus" << endl; |
|  | cin >> a >> b; |
|  | Rhombus rh(a, a, b); |
|  | cout << "Enter the sides of the rectangle" << endl; |
|  | cin >> a >> b; |
|  | Rectangle re(a, b); |
|  | cout << "Square of the rhombus = " << rh.square() << endl; |
|  | cout << "Square of the rectangle = " << re.square() << endl; |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 3

|  |
| --- |
| #include<iostream> |
|  | #include<string> |
|  | using namespace std; |
|  | class Vehicle// абстрактный класс "транспортное средство" |
|  | { |
|  | protected: |
|  | double x, y, dir; |
|  | public: |
|  | Vehicle(double xx, double yy, double d) : x(xx), y(yy), dir(d) {}; |
|  | virtual void move(double len) = 0;//движение транспорта |
|  | void setDir(double nd) //установить направление |
|  | { |
|  | dir = nd; |
|  | } |
|  | void returnPosition()//вывод позиции |
|  | { |
|  | cout << "x = " << x << " ; y = " << y << "; direction = " << dir << endl; |
|  | } |
|  | }; |
|  | class Car : public Vehicle// класс "автомобиль" |
|  | { |
|  | private: |
|  | double fuel, fcons; |
|  | int passengers; |
|  | public: |
|  | Car(double xx, double yy, double d, double fc) : Vehicle(xx, yy, d), passengers(0), fuel(0), fcons(fc) {}; |
|  | void move(double len) |
|  | { |
|  | if (fuel >= len / fcons) |
|  | { |
|  | x += len\*cos(dir); |
|  | y += len\*sin(dir); |
|  | fuel -= len / fcons; |
|  | } |
|  | else cout << "no fuel" << endl; |
|  |  |
|  | } |
|  | void refuel(double f)//заправить бензин |
|  | { |
|  |  |
|  | fuel += f; |
|  | cout << fuel << endl; |
|  | } |
|  | void enter(int n)//пассажиры зашли |
|  | { |
|  | if (n >= 0 && n < 4) |
|  | passengers += n; |
|  | } |
|  | void exit(int n)//пассажиры вышли |
|  | { |
|  | if (n >= 0 && n <= passengers) |
|  | passengers -= n; |
|  | } |
|  |  |
|  |  |
|  | }; |
|  | class Bus : public Vehicle// класс "автобус" |
|  | { |
|  | private: |
|  | double money, fuel, fcons; |
|  | int passengers; |
|  | public: |
|  | Bus(double xx, double yy, double d, double fc) : Vehicle(xx, yy, d), passengers(0), money(0), fuel(0), fcons(fc) {}; |
|  | void move(double len) |
|  | { |
|  | if (fuel >= len / fcons) |
|  | { |
|  | x += len\*cos(dir); |
|  | y += len\*sin(dir); |
|  | fuel -= len / fcons; |
|  | money += passengers\*len; |
|  | } |
|  | else cout << "no fuel" << endl; |
|  | } |
|  | void refuel(double f) |
|  | { |
|  | fuel += f; |
|  | } |
|  | void enter(int n) |
|  | { |
|  | if (n >= 0 && n <= 65) |
|  | passengers += n; |
|  | } |
|  | void exit(int n) |
|  | { |
|  | if (n >= 0 && n <= passengers) |
|  | passengers -= n; |
|  | } |
|  | }; |
|  | class Bike : public Vehicle// класс "велосипед" |
|  | { |
|  | public: |
|  | Bike(double xx, double yy, double d) : Vehicle(xx, yy, d) {}; |
|  | void move(double len) |
|  | { |
|  | x += len\*cos(dir); |
|  | y += len\*sin(dir); |
|  | } |
|  |  |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | Car c(0, 0, 0, 10); |
|  | Bus b(0, 0, 0, 44); |
|  | Bike bi(0, 0, 0); |
|  | c.refuel(10000); |
|  | c.setDir(90); |
|  | c.move(100); |
|  | b.setDir(90); |
|  | b.move(10); |
|  | bi.setDir(30); |
|  | bi.move(10); |
|  | cout << "Car position: " << endl; |
|  | c.returnPosition(); |
|  | cout << "Bus position: " << endl; |
|  | b.returnPosition(); |
|  | cout << "Bike position: " << endl; |
|  | bi.returnPosition(); |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 4

|  |
| --- |
| #include<iostream> |
|  | #include<string> |
|  | using namespace std; |
|  | class Screen |
|  | { |
|  | protected: |
|  | bool power; |
|  | public: |
|  | Screen() : power(0) {}; |
|  | virtual void turnOn() = 0; |
|  | virtual void turnOff() = 0; |
|  | }; |
|  | class Keyboard |
|  | { |
|  | protected: |
|  | string text; |
|  | public: |
|  | Keyboard(): text("Hello!") {}; |
|  | virtual void typeText(string t) = 0; |
|  | }; |
|  | class Laptop: public Screen, public Keyboard |
|  | { |
|  | public: |
|  | Laptop() {}; |
|  | void turnOn() { power = 1; } |
|  | void turnOff() { power = 0; } |
|  | void typeText(string t) { text = t; } |
|  | void state() |
|  | { |
|  | if (power) cout << "Laptop on" << endl << "Text:" << endl << text << endl; |
|  | else cout << "Laptop off" << endl; |
|  | } |
|  |  |
|  | }; |
|  | class Mobile : public Screen, public Keyboard |
|  | { |
|  | public: |
|  | Mobile() {}; |
|  | void turnOn() { power = 1; } |
|  | void turnOff() { power = 0; } |
|  | void typeText(string t) { text = t; } |
|  | void state() |
|  | { |
|  | if (power) cout << "Mobile on" << endl << "Text:" << endl << text << endl; |
|  | else cout << "Movile off" << endl; |
|  | } |
|  | }; |
|  | class PC : public Screen, public Keyboard |
|  | { |
|  | private: |
|  | bool PCpower; |
|  | public: |
|  | PC() : PCpower(0) {}; |
|  | void turnOn() { PCpower = 1; power = 1; } |
|  | void turnOff() { PCpower = 0; power = 0; } |
|  | void typeText(string t) { text = t; } |
|  | void state() |
|  | { |
|  | if (power && PCpower) cout << "PC on" << endl << "Text:" << endl << text << endl; |
|  | else cout << "PC off" << endl; |
|  | } |
|  | }; |
|  | int main() |
|  | { |
|  | Laptop l; |
|  | Mobile m; |
|  | PC p; |
|  | l.turnOn(); |
|  | l.state(); |
|  | m.turnOn(); |
|  | m.state(); |
|  | p.turnOn(); |
|  | p.state(); |
|  | system("pause"); |
|  | return 0; |
|  | } |

**Выводы:**

В ходе выполнения Практической работы №7 были использованы библиотеки iostream, string. Была реализованы методы для демонстрации работоспособности производных классов. Были достигнуты результаты:

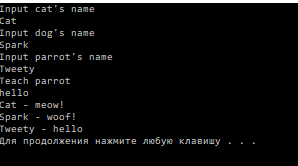


Рис. 21

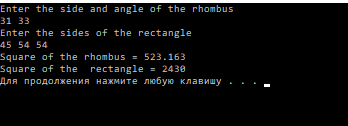


Рис. 22

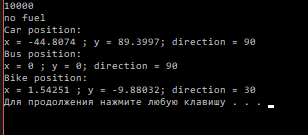


Рис. 23

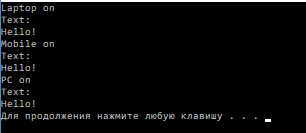


Рис. 24

<https://github.com/dneva/prac-lab/blob/master/7/prac7.1>

<https://github.com/dneva/prac-lab/blob/master/7/prac7.2>

<https://github.com/dneva/prac-lab/blob/master/7/prac7.3>

<https://github.com/dneva/prac-lab/blob/master/7/prac7.4>

# Практическая работа №8

**Задание:**

1. Реализуйте программу, считывающую текст из файла и выводящую

каждое слово с новой строки.

1. Реализуйте программу, считывающую текст с клавиатуры и

записывающую его в файл.

**Ход выполнения:**

Вариант 1

|  |
| --- |
| #include<iostream> |
|  | #include<string> |
|  | #include<fstream> |
|  | using namespace std; |
|  |  |
|  | int main() |
|  | { |
|  | string s; |
|  | ifstream fin("test.txt"); |
|  | while (fin >> s) |
|  | cout << s << endl; |
|  | fin.close(); |
|  | system("pause"); |
|  | return 0; |
|  | } |

Вариант 2

|  |
| --- |
| #include<iostream> |
|  | #include<string> |
|  | #include<fstream> |
|  | using namespace std; |
|  |  |
|  | int main() |
|  | { |
|  | string s; |
|  | cout << "Input text" << endl; |
|  | getline(cin,s); |
|  | ofstream fout("test.txt"); |
|  | fout << s; |
|  | fout.close(); |
|  | system("notepad.exe test.txt"); |
|  | return 0; |
|  | } |

**Выводы:**

В ходе выполнения Практической работы №8 были использованы библиотеки iostream,string,fstream. Были достигнуты результаты:

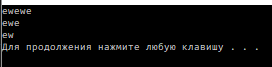


Рис. 25

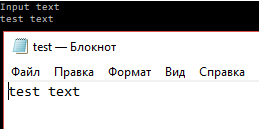


Рис. 26

<https://github.com/dneva/prac-lab/blob/master/8/prac8.1>

<https://github.com/dneva/prac-lab/blob/master/8/prac8.2>