# Объектно-ориентированное программирование

## Оформление решений

Важно, чтобы решённая задача была правильно оформлена. Это предполагает:

- использование несущих смысловую нагрузку имен переменных, констант и функций;
- применение отступов при записи инструкций программы;
- использование комментариев.

Правильно оформленную программу легче отлаживать, кроме того, она производит хорошее впечатление.

## Пример программы

```
#include "stdio.h"
#include "conio.h"
#include <iostream>

class Student
{
public:
     int semesterHours;  // свойства (или поля)
     float gpa;
     void addCourse (int hours, float grade)  // метод
     {
          semesterHours = hours;
          gpa = grade;
     }
};

int main(void)
{
     Student s,t;
     using namespace std;
     setlocale(LC_ALL, "Russian");
     s.addCourse (10, 4.0);
     t.addCourse (12, 3.5);
     cout << s.gpa << endl;
     cout << s.semesterHours << endl;
     cout << t.gpa << endl;
     cout << t.semesterHours << endl;
     return 0;
}
```

# Лабораторная работа №1

**Цель работы**: Знакомство с объектно-ориентированным программированием в C++. Изучение общих понятий о классах – поля класса, методы класса.

## Задачи

**1**. Создать класс:
Комплексное число в алгебраической форме a = x + iy.
Члены класса: Действительная ***x*** и мнимая ***y*** части числа.
Методы класса: метод вычисления модуля комплексного числа и метод вывода комплексного числа.

Продемонстрировать работу класса на своём примере.

**2**. Создать класс:
Обыкновенная дробь.
Члены класса: Числитель и знаменатель.
Методы класса: метод сокращения дроби, метод вывода дроби в виде обыкновенной и десятичной дроби (округление до 5 разрядов).

Продемонстрировать работу класса на своём примере.

**3**. Создать класс:
Вектор.
Члены класса: три прямоугольные декартовы координаты.
Методы класса: метод вывода вектора, метод вычисления длины вектора.

Продемонстрировать работу класса на своём примере.

**4**. Создать класс:
Матрица.
Члены класса: Размерность матрицы, элементы матрицы.
Методы класса: метод вывода матрицы, метод вычисления определителя матрицы, проверка, является ли матрица диагональной, нулевой, единичной, проверка, является ли матрица симметричной ($A^T=A$)

Продемонстрировать работу класса на своём примере.

**5**. Создать класс:
Прямая.
Члены класса: Координаты двух точек (x1,y1) и (x2,y2).
Методы класса: метод вывода уравнения прямой.

Продемонстрировать работу класса на своём примере.

**6**. Создать класс:
Парабола $y = ax^2 + bx + c$.
Члены класса: Коэффициенты a, b, c.
Методы класса: методы вывода уравнения параболы, вычисления экстремума функции (min или max).

Продемонстрировать работу класса на своём примере.

**7**. Создать класс:
Комплексное число в тригонометрической форме $a = \rho \cdot (\cos\varphi + i \cdot \sin\varphi)$.
Члены класса: Модуль ρ и аргумент φ числа.
Методы класса: метод вывода комплексного числа в тригонометрической и алгебраической формах

Продемонстрировать работу класса на своём примере.

**8**. Создать класс:
Время.
Члены класса: Часы, минуты, секунды.
Методы класса: метод вывода времени и составляющей суток (до 6 – ночь, до 12 – утро, до 18 – день, до 24 – вечер)

Продемонстрировать работу класса на своём примере.

# Лабораторная работа №2

## Задачи

Разработать классы для описанных ниже объектов. Включить в класс методы set (…), get (…), show (…). Определить другие методы.

1. **Abiturient**: Фамилия, Имя, Отчество, Адрес, Оценки. Создать массив объектов. Вывести:

а) список абитуриентов, имеющих неудовлетворительные оценки;

б) список абитуриентов, сумма баллов у которых не меньше заданной;

в) выбрать N абитуриентов, имеющих самую высокую сумму баллов, и список абитуриентов, имеющих полупроходной балл.

2. **Aeroflot**: Пункт назначения, Номер рейса, Тип самолета, Время вылета, Дни недели. Создать массив объектов. Вывести:

а) список рейсов для заданного пункта назначения;

б) список рейсов для заданного дня недели;

в) список рейсов для заданного дня недели, время вылета для которых больше заданного.

3. **Book**: Автор, Название, Издательство, Год, Количество страниц. Создать массив объектов. Вывести:

а) список книг заданного автора;

б) список книг, выпущенных заданным издательством;

в) список книг, выпущенных после заданного года.

4. **Worker**: Фамилия и инициалы, Должность, Год поступления на работу, Зарплата. Создать массив объектов. Вывести:

а) список работников, стаж работы которых на данном предприятии превышает заданное число лет;

б) список работников, зарплата которых больше заданной;

в) список работников, занимающих заданную должность.

5. **Train**: Пункт назначения, Номер поезда, Время отправления, Число общих мест, Купейных, Плацкартных. Создать массив объектов. Вывести:

а) список поездов, следующих до заданного пункта назначения;

б) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа;

в) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места.

6. **Product**: Наименование, Производитель, Цена, Срок хранения, Количество. Создать массив объектов. Вывести:

а) список товаров для заданного наименования;

б) список товаров для заданного наименования, цена которых не превышает указанной;

в) список товаров, срок хранения которых больше заданного.

7. **Patient**: Фамилия, Имя, Отчество, Адрес, Номер медицинской карты, Диагноз. Создать массив объектов. Вывести:

а) список пациентов, имеющих данный диагноз;

б) список пациентов, номер медицинской карты которых находится в заданном интервале.

8. **Bus**: Фамилия и инициалы водителя, Номер автобуса, Номер маршрута, Марка, Год начала эксплуатации, Пробег. Создать массив объектов. Вывести:

а) список автобусов для заданного номера маршрута;

б) список автобусов, которые эксплуатируются больше 10 лет;

в) список автобусов, пробег у которых больше 10 000 км.

9. **House**: Адрес, Этаж, Количество комнат, Площадь. Создать массив объектов. Вывести:

а) список квартир, имеющих заданное число комнат;

б) список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в определенном промежутке;

в) список квартир, имеющих площадь, превосходящую заданную.

10. **File**: Имя файла, Размер, Дата создания, Количество обращений. Создать массив объектов. Вывести:

а) список файлов, упорядоченный в алфавитном порядке;

б) список файлов, размер которых превышает заданный;

в) список файлов, число обращений к которым превышает заданное.

# Лабораторная работа №3

## *Дружественные функции*

***Дружественные функции*** класса определяются вне области действия этого класса, но имеют право доступа к закрытым элементам **private** данного класса. Функции или класс в целом могут быть объявлены ***другом*** (**friend**) другого класса.

Дружественные функции используются для повышения производительности. Дружественные функции применяются, как правило, для перегрузки операций, используемых классами, и для создания классов итераторов. Объекты класса итератора используются, чтобы последовательно выделять элементы или выполнять операции над элементами в объекте класса контейнера. Объекты классов контейнеров способны хранить множество элементов в форме, подобной массиву.

Чтобы объявить функцию как друга (**friend**) класса, перед ее прототипом в описании класса ставится ключевое слово **friend**.

Спецификаторы доступа к элементам private, protected и public не имеют отношения к объявлениям дружественности, так что эти объявления дружественности могут помещаться в любом месте в описании класса.

**Пример**

```
#include <iostream>
using namespace std;

class Count
{
    friend void setX(Count &, int);
    public:
    Count( ){ x=0; }
    inline void print( ) const {cout<<"x="<<x<<endl;}
    private:
    int x;
};
void setX(Count &c, int val)
{
    c.x=val;
}
main ( )
{
    Count obj;
    cout<<"Obj.x before setX:";
    obj.print( );
    cout<<"Obj.x after setX:";
    setX(obj,10);
    obj.print( );
    system ("Pause");
    return 0;
}
```

## Задачи

**1**. На основе созданного класса для работы с рациональными дробями (вида m/n), создать дружественную функцию. Функция должна менять местами числитель и знаменатель.

**2**. На основе созданного класса для работы с арифметическими комплексными числами создать дружественную функцию. Функция должна менять местами действительную и мнимую части.

**3**. На основе созданного класса для работы с тригонометрическими комплексными числами создать дружественную функцию. Функция должна менять местами модуль и аргумент.

**4**. На основе созданного класса описывающего успеваемость студента создать дружественную функцию. Функция должна изменять оценки по всем трем предметам.

**5**. На основе созданного класса описывающего библиотечную карточку создать дружественную функцию. Функция должна обнулять количество всех взятых книг.

**6**. На основе созданного класса описывающего студенческую группу создать дружественную функцию. Функция должна добавлять в группу заданное количество иностранных студентов.

**7**. На основе созданного класса для работы с массивом создать дружественную функцию. Функция должна сортировать массив по убыванию.

**8**. На основе созданного класса для работы с матрицей создать дружественную функцию. Функция должна сортировать i-ю строку по убыванию.

**9**. На основе созданного класса для работы с датой создать дружественную функцию. Функция должна добавлять к текущей дате определенное количество дней.

**10**. На основе созданного класса описывающего запись в книге учета постояльцев в гостинице создать дружественную функцию. Функция должна добавлять один день к сроку проживания в гостинице.