**Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования**
**"Уфимский государственный авиационный технический университет"**

**Кафедра** Высокопроизводительных вычислительных технологий и систем

**Дисциплина:** Программирование

# Отчет по лабораторной работе № _6_

**Тема:** «Классы и наследование»

| Группа ПМ-153 | Фамилия И.О. | Подпись | Дата | Оценка |
|---|---|---|---|---|
| Студент | Шамаев И.Р. | | | |
| Принял | Гайнетдинова А.А | | | |

**Уфа 2019**

**Цель:** *Ознакомиться с принципом создания классов и изучить на практике принципы построения иерархии классов*

**Теоретический материал**

**Ответы на контрольные вопросы?**

1) **Что такое класс? Что входит в состав класса?**
   **Класс –** модель еще не существующей сущности(объекта), описываемая на языке терминологии исходного кода. Объект – экземпляр класса.
   **В него входят:** конструктор, деструктор, set и get методы

2) **Из каких двух частей состоит описание класса в C++?**
   Поля данных (необходимые в программе параметры объекта, задающие его состояние) и методы (процедуры и функции, связанные с классом)
3) **Для чего необходим и когда вызывается конструктор класса?**
   Вызов конструктора осуществляется при создании объекта класса. Конструктор класса вызывается компилятором.
   Как правило, конструктор используется для:

   1)выделения памяти под объект класса

   2)начальной инициализации внутренних данных класса

   Конструктор предназначен для формирования экземпляра объекта класса. Имя конструктора класса совпадает с именем класса.

4) **Для чего необходим и когда вызывается деструктор класса?**
   Деструктор автоматически вызывается, когда удаляется объект. Когда объект автоматически выходит из области видимости, вызывается деструктор класса (если он существует) для выполнения необходимой очистки до того, как объект будет удалён из памяти
5) **При помощи чего в классах обеспечивается инкапсуляция внутренней структуры данных?**
   Классический язык C++ позволяет устанавливать доступ к членам класса с помощью трех спецификаторов: private, protected, public.
6) **Опишите назначение модификаторов видимости. Какие у них общие черты и различия?**

Так как все типы и члены типов имеют уровень доступности. Он определеняет возможность использования из другого кода в вашей или в других сборках. Модификаторы доступа:
1)*public: Доступ возможен из любого другого кода в той же сборке или другой сборке, ссылающейся на него
2)*private: Доступ возможен только из кода в том же объекте class или struct
3)*protacted Доступ возможен только из кода в том же объекте class, либо в class, производном от этого класса

7) **Что такое наследование и иерархия?**
**Наследование** – свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется бозовым, родительским или суперклассом. Новый класс – потомком, наследником, дочерним классом.
**Иерархия** наследования представляет собой особый тип объединения сущностей, которые разделяют общие характеристики.

8) **Как и для чего используется служебное слово virtual?**
Служебное слово virtual показывает, что функция может иметь разные версии в разных производных классах. Допустим, чтобы переопределить в классе-потомке метод, присутствующий в классе-предке. Для этого перед инициализации метода, необходимо поставить virtual.

9) **Что такое абстрактный класс?**
При разработке иерархии классов всегда в качестве основы берут нечто, обладающее самым общими атрибутами и методами, но не реализующееся в природе или в рассматриваемой области. Вот такая сущность называется абстрактным классом.

10) **Почему конструктор не может быть виртуальным, а деструктор почти всегда является виртуальным?**
Виртуальный механизм работает на логически завершенном (полностью построенном) объекте. Мы знаем, что мы используем конструкторы для логической инициализации наших объектов. Другими словами, объект не полностью сконструирован, пока конструктор не завершит выполнение. Таким образом, у нас нет виртуальных конструкторов. Если деструктор не будет виртуальным он будет уничтожать класс родителя, а не собственный класс

**11)Что такое композиция?**

Композиция: если поле у нас имеет тип Класс, оно может содержать ссылку на другой объект этого класса, создавая таким образом связь между двумя объектами. То есть один объект предоставляет другому свою функциональность частично или полностью.

Композиция– простейший механизм для создания нового класса путем объединения нескольких объектов существующих классов в единое целое

**Индивидуальное задание №1**

Задание: **1.**

**1.**Создать класс **Файл**, имеющий поля:
-имя,
-расширение,
-размер.

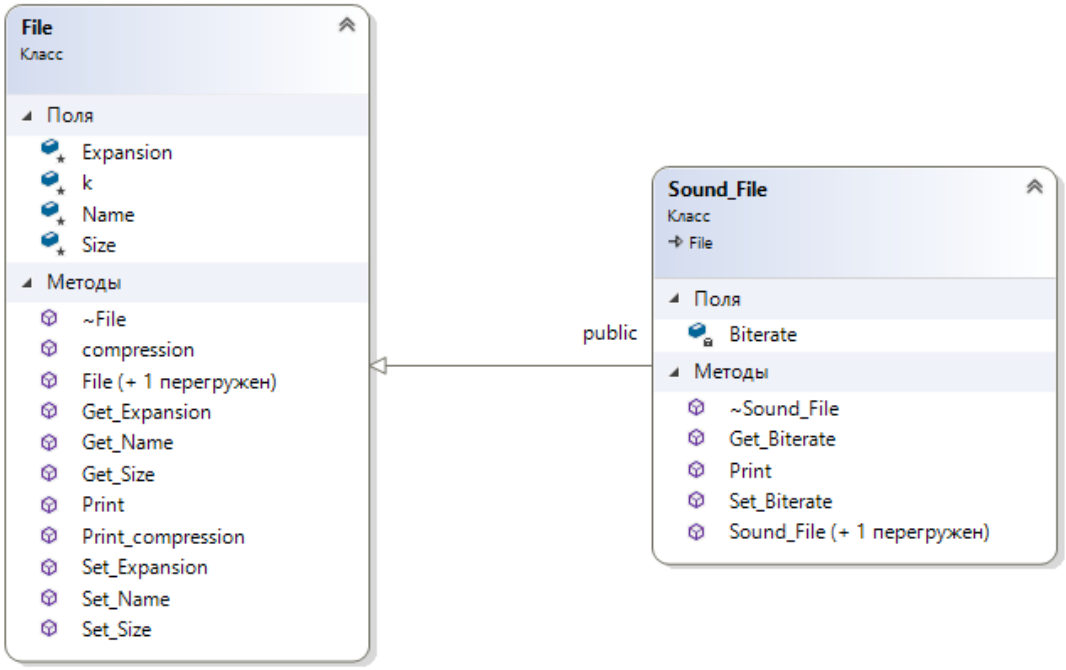
Определить
-пустой конструктор и конструктор с полным заполнением полей,
-деструктор (с очисткой памяти),
-простые методы доступа к полям (set- и get-методы),
-метод вывода полной информации об объекте класса,
-метод расчета размера при сжатии с заданным коэффициентом.
Класс должен быть разработан с соблюдением всех принципов объектноориентированного программирования
Создать объекты данного класса в функции main() и продемонстрировать
работу всех методов класса.

**2.** Создать public-производный класс **звуковой файл**, имеющий
дополнительно поля:
-битрейт.

Определить
- пустой конструктор и конструктор с полным заполнением полей,
-деструктор (с очисткой памяти),
-простые методы доступа к полям (set- и get-методы),
-метод вывода полной информации об объекте класса (перегрузка
метода родителя).
Создать объекты данного класса в функции main() и продемонстрировать
работу всех методов класса (в т.ч. унаследованных)

Описание программы**:** Описываем класс Sound_File и задаем такие поля как Имя, Размер и Расширение файла. Далее описывает конструктор и деконструктор и get и set методы. Задаем функцию вывода класса и расчета размера файла после сжатия. Задаем нужные параметры и проверяем

Блок-схема:




Исходный код программы:

```
#include <iostream>
#include <string>
#include <cstdio>

using namespace std;

class File{
	protected:
	string Name; // Имя
	string Expansion; // Расширение
	int Size; // Размер
	float k = 0.3;
```

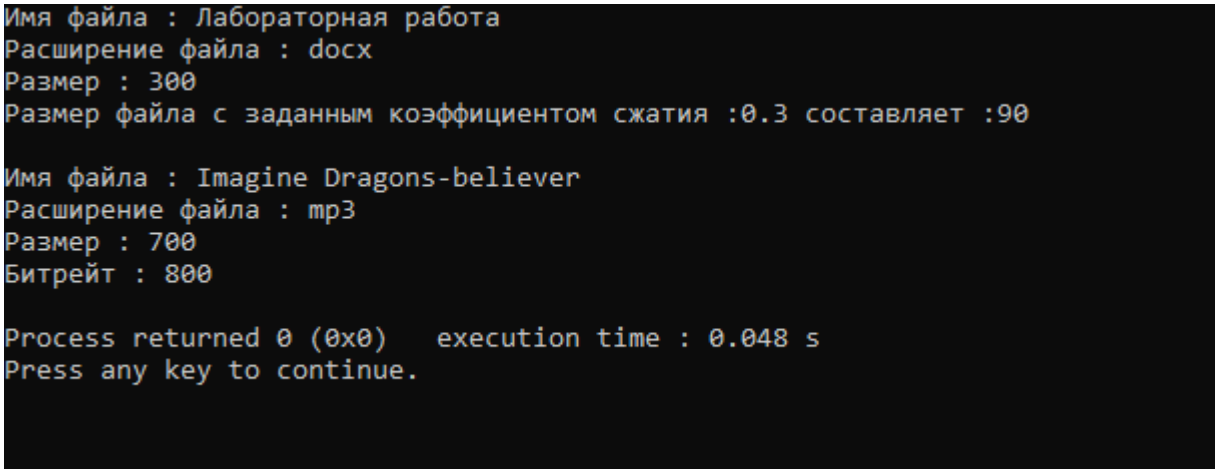
```
        public:
        File();//Пустой конструктор
        File(string NName, string NExpansion, int NSize);
        ~File(); // Деструктор
        int compression();// Функция сжатия

        string Get_Name();
        void Set_Name(string NName);
        string Get_Expansion();
        void Set_Expansion(string NExpansion);
        int Get_Size();
        void Set_Size(int NSize);
        virtual void Print();
        void Print_compression();

};
        string Name;
        string Expansion;
        int Size;
        File :: File()
        {
           Name=" everything ";
           Expansion=" everyone ";
           Size = 99999;
        } //Пустой Конструктор
        File :: File(string NName, string NExpansion, int NSize)
        {
           Name = NName;
           Expansion = NExpansion;
           Size = NSize;
        }//заполненный конструктор

        string File ::Get_Name(){
        return Name;
        }
        void File ::Set_Name(string NName){
        Name= NName;
        }
         string File ::Get_Expansion(){
```

```
return Expansion;
}
void File ::Set_Expansion(string NExpansion){
Expansion= NExpansion;
}
int File ::Get_Size(){
return Size;
}
void File ::Set_Size(int NSize){
Size= NSize;
}
void File ::Print()
{
    cout<< "Имя файла : "<< Name<<"\nРасширение файла : " <<
Expansion<<"\nРазмер : " << Size <<endl;
}
void File::Print_compression()
{
   int R=0;
   R=Size*k;
    cout<<"Размер файла с заданным коэффициентом сжатия :"<<k<<
" составляет :" << R;
}

   class Sound_File : public File
{
   private:
      int Biterate;
   public:
      Sound_File();
      Sound_File(string NName, string NExpansion, int Size, int NBiterate);
      ~Sound_File(); // Деструктор
      int Get_Biterate();
      void Set_Biterate(int NBiterate);
      void Print();
};
      int Biterate;
      Sound_File :: Sound_File()
      {
```

```
        Name="everything";
        Expansion="everyone";
        Size = 99999;
        Biterate = 88888;
    }
     Sound_File ::Sound_File(string NName, string NExpansion, int NSize,
int NBiterate) : File( NName, NExpansion, NSize)
    {
        Biterate = NBiterate;
    }
    int Sound_File :: Get_Biterate()
    {
        return Biterate;
    }
    void Sound_File::Set_Biterate(int NBiterate)
    {
        Biterate = NBiterate;
    }

    void Sound_File :: Print()
    {
    File::Print();
    cout<< "Битрейт : " << Biterate <<endl;
    }



File::~File()
{
}

Sound_File::~Sound_File()
{
}

int main()
{
 setlocale(LC_ALL, "Russian");
```

```
    File Word("Лабораторная работа","docx",300);
    Word.Print();
    Word.Print_compression();
    puts("\n");
    Sound_File Music("Imagine Dragons-believer","mp3",700, 800);
    Music.Print();

    return 0;
}
```

Пример выполнения программы



**Вывод**

*Я ознакомиться с принципом создания классов и изучил на практике принципы построения иерархии классов*