Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №18.2**

Дисциплина: «Информатика»

Тема: Объектно-ориентированное программирование. Использование конструкторов

Вариант 13

Выполнил:

Студент группы РИС-20-1б

Кузнецов Михаил Сергеевиич

Проверила:

Доцент кафедры ИТАС

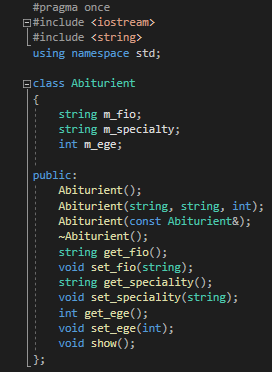
Полякова О. А.

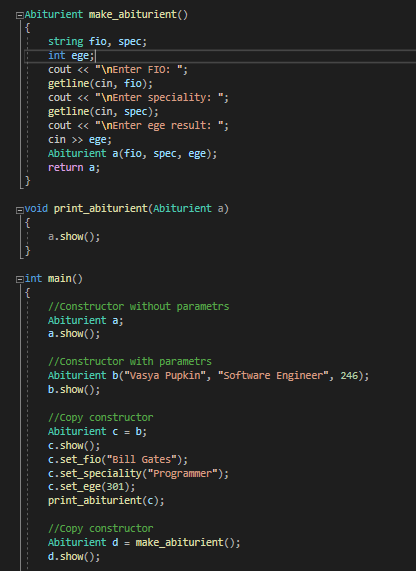
**Постановка задачи**

1. Определить пользовательский класс. Определить в класс следующие конструкторы: без параметров, с параметрами, копирования. Определить в классе деструктор.

**Анализ задачи**

1. Для решения задачи необходимо…
   1. Организовать класс Abiturient, который хранит имя, специальность и результат за экзамены.
   2. Организовать методы для ввода данных, гетторы, сетторы, конструкторы и деструктор.



* 1. В функции main произвести тестирование. 

1. В программе были использованы следующие типы данных:
   1. Тип данных int для хранения результата за экзамен;



* 1. Тип данных string для хранения ФИО и специальности;



**Код**

#include <iostream>

#include <string>

#include "Abiturient.h"

using namespace std;

Abiturient make\_abiturient()

{

string fio, spec;

int ege;

cout << "\nEnter FIO: ";

getline(cin, fio);

cout << "\nEnter speciality: ";

getline(cin, spec);

cout << "\nEnter ege result: ";

cin >> ege;

Abiturient a(fio, spec, ege);

return a;

}

void print\_abiturient(Abiturient a)

{

a.show();

}

int main()

{

//Constructor without parametrs

Abiturient a;

a.show();

//Constructor with parametrs

Abiturient b("Vasya Pupkin", "Software Engineer", 246);

b.show();

//Copy constructor

Abiturient c = b;

c.show();

c.set\_fio("Bill Gates");

c.set\_speciality("Programmer");

c.set\_ege(301);

print\_abiturient(c);

//Copy constructor

Abiturient d = make\_abiturient();

d.show();

}

#include "Abiturient.h"

#include <iostream>

#include <string>

using namespace std;

Abiturient::Abiturient()

{

m\_fio = "";

m\_specialty = "";

m\_ege = 0;

cout << "Constructor without parametrs for object: " << this << endl;

}

Abiturient::Abiturient(string fio, string spec, int ege)

{

m\_fio = fio;

m\_specialty = spec;

m\_ege = ege;

cout << "Constructor with parametrs for object: " << this << endl;

}

Abiturient::Abiturient(const Abiturient &a)

{

m\_fio = a.m\_fio;

m\_specialty = a.m\_specialty;

m\_ege = a.m\_ege;

cout << "Copy Constructor for object: " << this << endl;

}

Abiturient::~Abiturient()

{

cout << "Distructor for object: " << this << endl;

}

string Abiturient::get\_fio()

{

return m\_fio;

}

void Abiturient::set\_fio(string fio)

{

m\_fio = fio;

}

string Abiturient::get\_speciality()

{

return m\_specialty;

}

void Abiturient::set\_speciality(string speciality)

{

m\_specialty = speciality;

}

int Abiturient::get\_ege()

{

return m\_ege;

}

void Abiturient::set\_ege(int ege)

{

m\_ege = ege;

}

void Abiturient::show()

{

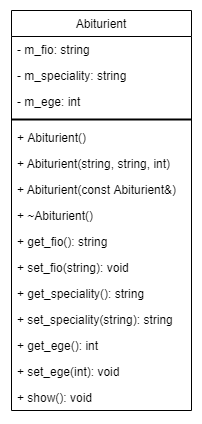
cout << "\nAbiturients FIO: " << m\_fio << endl;

cout << "Abiturients speciality: " << m\_specialty << endl;

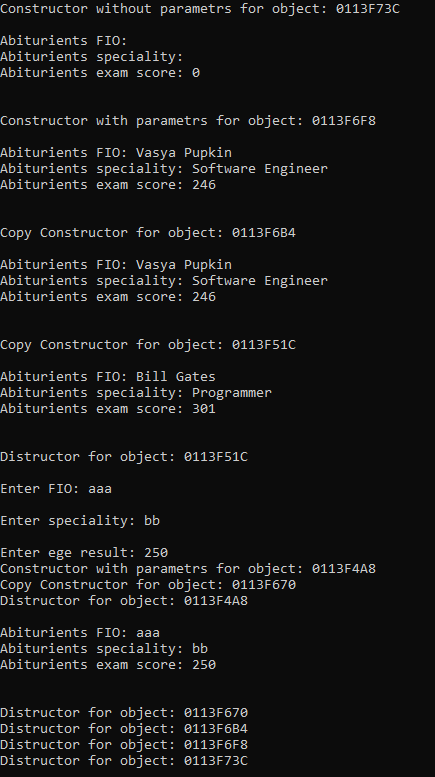
cout << "Abiturients exam score: " << m\_ege << endl << endl << endl;

}

**UML диаграмма**



**Скриншоты тестов**

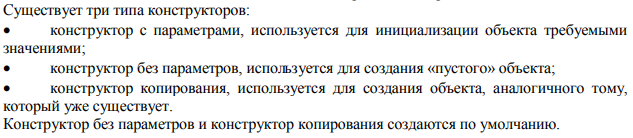
****

**Контрольные вопросы**



1. Конструктор необходим для инициализации объекта.



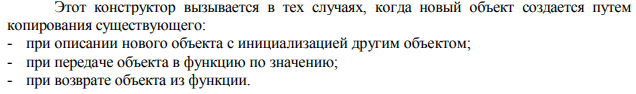




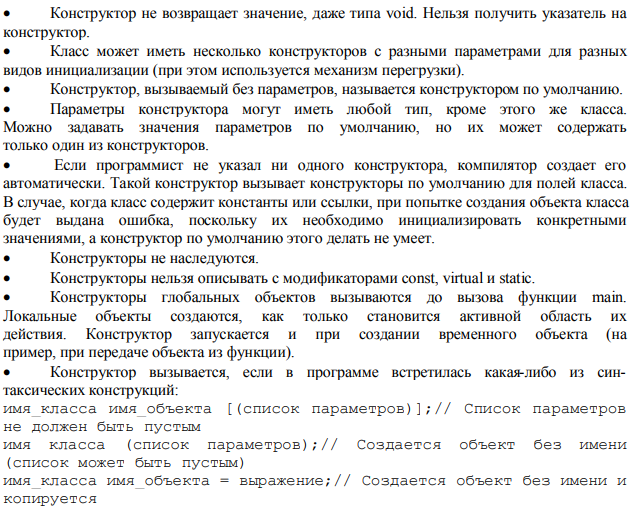
3. Деструктор – это особый вид метода, применяющийся для освобождения ресурсов, выделенных конструктором объекту. Описывать деструктор явным образом требуется в случае, когда объект содержит указатели на память, выделяемую динамически.



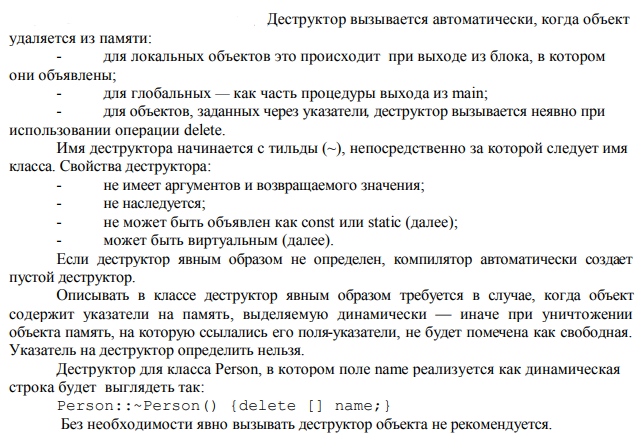
4. Конструктор с параметрами используется для инициализации объекта требуемыми значениями. Конструктор без параметров используется для создания «пустого» объекта. Конструктор копирования – это специальный вид конструктора, получающий в качестве единственного параметра указатель на объект этого же класса.





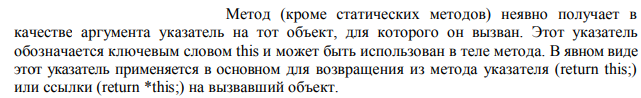


7. 

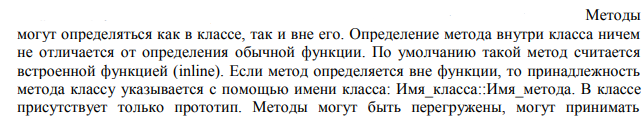


8.



9.



10.



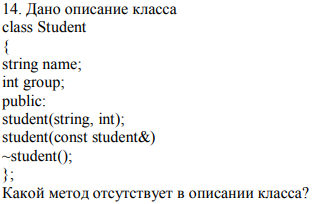
11. Конструктор не возвращает значения.



12. Конструкторы и деструкторы без параметров, если таковые не были прописаны явно.



13. Деструктор не возвращает значения.



14. Отсутствует конструктор без параметров в описании класса.



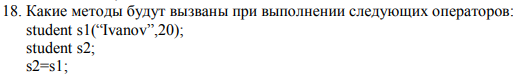
15. Будет вызван конструктор без параметров.



16. Будет вызван конструктор с параметрами.



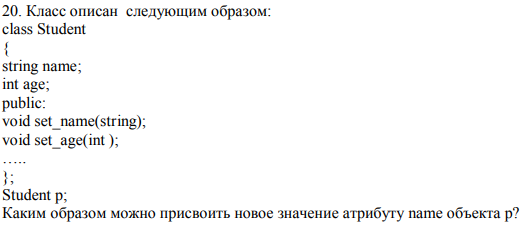
17. Будут вызваны конструктор с параметрами, а затем конструктор копирования.



18. Будут вызваны конструктор с параметрами, затем конструктор без параметров, а затем конструктор копирования.



19. Конструктор без параметров.



20. C помощью метода set\_name().