

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФГБОУ ВО

Московский авиационный институт

(национальный исследовательский университет)

Кафедра 304

Вычислительные машины, системы и сети

Отчет по лабораторной работе №1

по учебной дисциплине «Конструирование ПО»

Выполнил студент группы МЗО-311Б-21:

Давыдов А.П.

Принял:

Грабовский М. Н.

Москва 2023

Оглавление

Задание 1.....3

Задание 2.....5

Задание 3.....7

Задание 1

Код программы:

```
#include <iostream>
#include <string>
#include "MainHeader.h"

int main()
{
    Arbeit::Example* obj = new Arbeit::Example("Cake"); // Создание нового объекта с
    использованием пространства имён
    Arbeit::Example::Print_Unhinged(); // Статический метод

    obj->Print_Objected(); // Обращение к не-статическому методу по указателю

    Arbeit::Example alive = *(new Arbeit::Example("Flower"));
    alive.MakeAlive(); // Обращение к статическому методу с статической переменной
    alive.Print_Objected(); // Обращение по значению
    obj->Print_Objected(); // Ещё для примера использования статической переменной
}

bool Arbeit::Example::isAlive = false;

Arbeit::Example::Example(std::string name)
{
    static bool isAlive = false;
    this->name = name;
    std::cout << "New object of name " << name << " was created\n";
}
Arbeit::Example::~Example()
{
}

void Arbeit::Example::Print_Unhinged()
{
    std::cout << "I'm not linked to any object, and can be called directly\n";
}

void Arbeit::Example::Print_Objected()
{
    std::cout << "I'm " << name << "!\n";
    if (isAlive)
        std::cout << "And I'm alive.\n";
}
```

Файл заголовка:

```
#pragma once
#include <string>
#include <iostream>

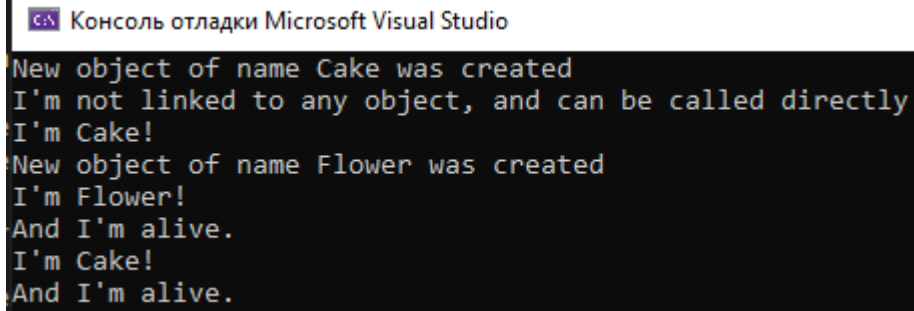
using namespace std;

namespace Arbeit {
    class Example
    {
    private:
        string name;
        static bool isAlive;

    public:
        static void MakeAlive()
        {
            isAlive = true;
        }
    };
}
```

```
}  
  
public: Example(string name);  
public: ~Example();  
  
public: static void Print_Unhinged();  
public: void Print_Objected();  
};  
}
```

Пример:



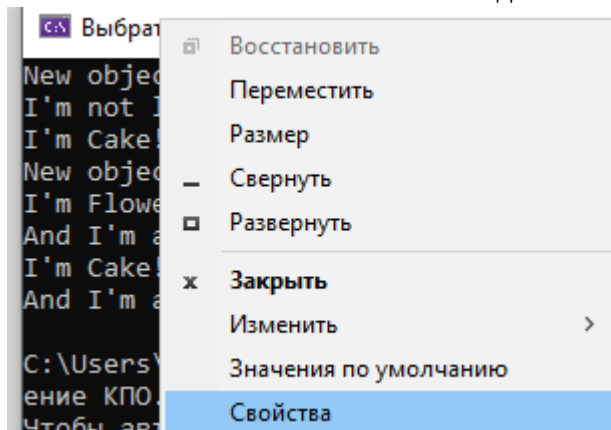
Консоль отладки Microsoft Visual Studio

```
New object of name Cake was created  
I'm not linked to any object, and can be called directly  
I'm Cake!  
New object of name Flower was created  
I'm Flower!  
And I'm alive.  
I'm Cake!  
And I'm alive.
```


Задание 2

Название	Назначение
Windows.h	Основанный на языке C и специализированный под Windows файл-заголовок. Очень важный, и должен находится над другими заголовочными файлами, так как сам является базисом для них.
Stdlib.h	Standart Template Library. Содержит в себе самые основные программистские структуры и функции для работы.
Malloc.h	Используется для ООП.
Последовательность функции	
wWinMain()	Функция main(), с которой работает пользователь, создаёт окно и функции обработки сообщений Windows.
MyRegisterClass()	Класс окна и классы пользователя.
InitInstance()	Создаёт само окно и инициализирует всё.
WndProc()	Обработка сообщений

Все основные изменения в консоли делаются в «Свойствах»:



Там можно поменять шрифт, цвет фона, текста и другие параметры.

 Консоль отладки Microsoft Visual Studio

```
New object of name Cake was created
I'm not linked to any object, and can be called directly
I'm Cake!
New object of name Flower was created
I'm Flower!
And I'm alive.
I'm Cake!
And I'm alive.
```

Задание 3

1)

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

int number = 0;

int counter()
{
    return number++;
}

auto makeCounter()
{
    number = 0;
    // Вернуть нужно функцию.
    return counter;
}

int main()
{
    auto Counter1 = makeCounter();
    std::cout << Counter1() << std::endl;
    std::cout << Counter1() << std::endl;
    auto Counter2 = makeCounter();
    std::cout << Counter2() << std::endl;
    std::cout << Counter2() << std::endl;
    std::cout << Counter2() << std::endl;
    system("pause");
}
```

2)

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

int number = 0;

int counter()
{
    return number++;
}

auto makeCounter()
{
    // Судя по "Например" этого не обязательно делать, но я хочу достичь 100%-го
    // совпадения
    if(number != 0)
        number++;
    return counter;
}

int main()
{
    auto Counter1 = makeCounter();
    std::cout << Counter1() << std::endl;
    std::cout << Counter1() << std::endl;
    auto Counter2 = makeCounter();
    std::cout << Counter2() << std::endl;
    std::cout << Counter2() << std::endl;
    std::cout << Counter2() << std::endl;
    system("pause");
}
```

```

3)
#include <iostream>
#include <string>

int numberCounting = 0;

auto Count()
{
    return numberCounting++;
}

auto MakeCounter()
{
    if(numberCounting != 0)
        numberCounting--; // Чтобы сделать логику последовательности
    return Count;
}

auto Factory()
{
    return MakeCounter;
}

int main()
{
    auto makeCounter = Factory();
    auto Counter1 = makeCounter();
    std::cout << Counter1() << std::endl;
    std::cout << Counter1() << std::endl;
    auto Counter2 = makeCounter();
    std::cout << Counter2() << std::endl;
    std::cout << Counter2() << std::endl;
    auto Counter3 = makeCounter();
    std::cout << Counter3() << std::endl;
    std::cout << Counter3() << std::endl;
    std::cout << Counter3() << std::endl;
    system("pause");
}

```