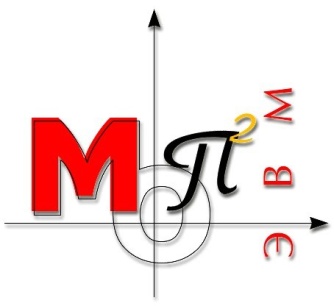
Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждения высшего образования   
«Южный федеральный университет»

Инженерно-технологическая академия

Институт компьютерных технологий и информационной безопасности

Кафедра математического обеспечения и применения ЭВМ

**

Лабораторная работа №1

по дисциплине

"ООП"

на тему

"Классы и объекты в С++"

**Выполнил:**

студент группы КТбо2-1

Сафронов Н. А.

**Проверил:**

Таганрог, 2020

**Вариант №7**

**Задание:**

Определить класс Множество (Set) c методами: добавление элементов в множество и проверкой пересечения двух множеств. Мощность множества задается при ее создании.

**Спецификация классов:**

**Set.h**

#pragma once

class Set

{

private:

int \_power = 0;

int\* \_elements = nullptr;

public:

Set();

Set(int& power);

Set(int\* setElements, int& power);

void CreatedSet(int\* setElement, int& power);

int GetPower() const;

int GetElements(int count) const;

void AddSetElement(int& element);

bool CheckSetCommon(Set& set);

~Set();

};

**Interface.h**

#pragma once

#include "Set.h"

class Interface

{

private:

Set\* set = new Set[2];

void CreatedTwoSet();

void ChangeSet(int& numberSet);

void AddElement(int& numberSet);

void CheckCommon();

void GetPower(int& numberSet);

void GetElements(int& numberSet);

void printMenuText();

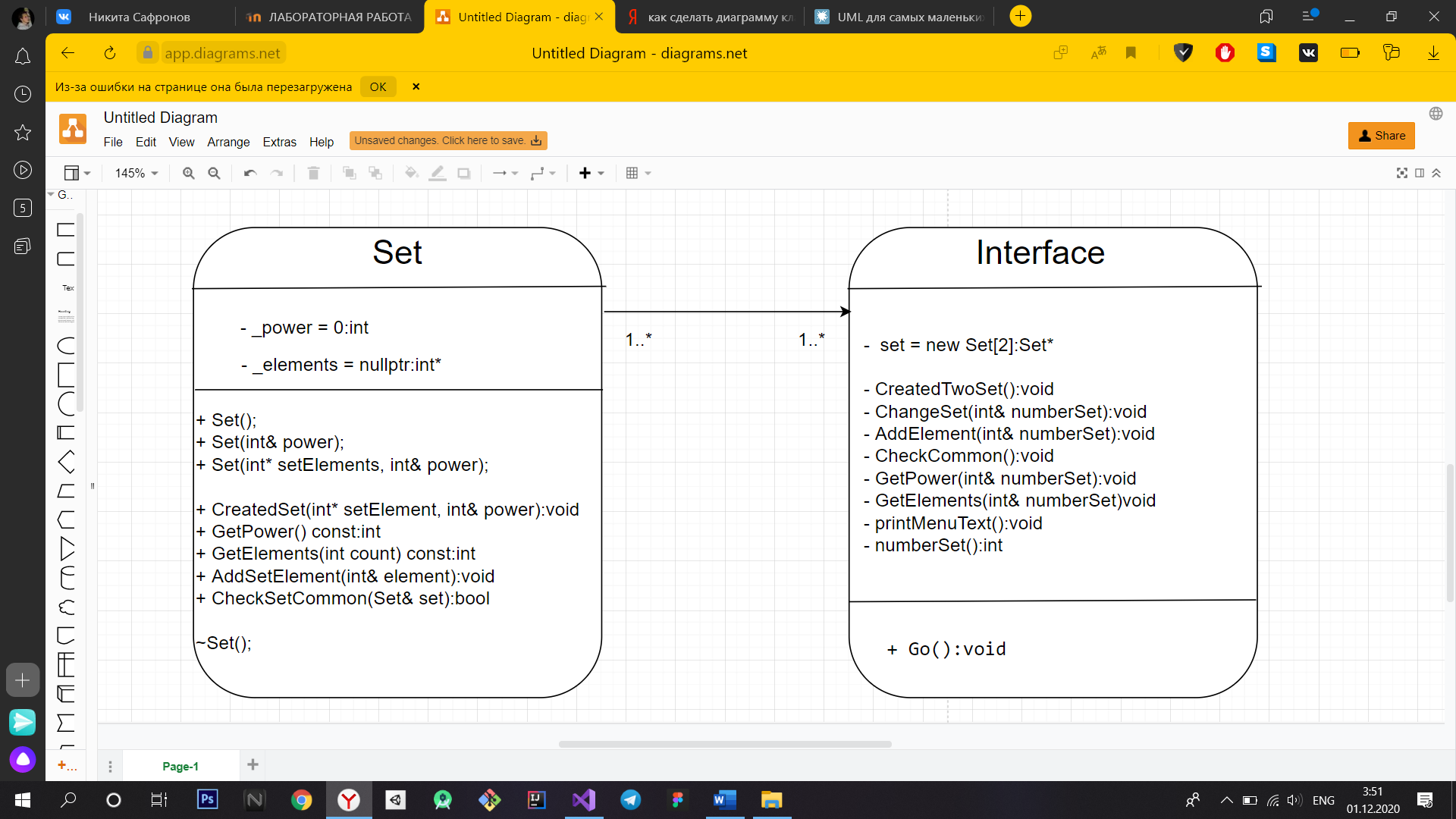
int numberSet();

public:

void Go();

};

**Диаграмма классов:**



**Листинг кода**

//Set.h

#pragma once

class Set

{

private:

int \_power = 0;

int\* \_elements = nullptr;

public:

Set();

Set(int& power);

Set(int\* setElements, int& power);

void CreatedSet(int\* setElement, int& power);

int GetPower() const;

int GetElements(int count) const;

void AddSetElement(int& element);

bool CheckSetCommon(Set& set);

~Set();

};

//Set.cpp

#include "Set.h"

#include <string>

Set::Set(){}

Set::Set(int& power)

{

\_elements = new int[power];

\_power = power;

}

Set::Set(int\* setElements, int& power)

{

\_elements = new int[power];

\_power = power;

for (int i = 0; i < \_power; i++)

{

\_elements[i] = setElements[i];

}

}

Set::~Set()

{

delete[] \_elements;

}

int Set::GetPower() const

{

return \_power;

}

int Set::GetElements(int count) const

{

return \_elements[count];

}

void Set::CreatedSet(int\* setElements, int& power)

{

\_power = power;

\_elements = new int[power];

for (int i = 0; i < \_power; i++)

{

\_elements[i] = setElements[i];

}

}

void Set::AddSetElement(int& element)

{

int\* tempElements = \_elements;

\_elements = new int[\_power+1];

memcpy(\_elements, tempElements, sizeof(int) \* \_power);

delete[] tempElements;

\_elements[\_power] = element;

\_power++;

}

bool Set::CheckSetCommon(Set& set)

{

for (int i = 0; i < \_power; i++)

{

for (int j = 0; j < set.\_power; j++)

{

if (\_elements[i] == set.\_elements[j])

{

return true;

}

}

}

return false;

}

//Interface.cpp

#pragma once

#include "Set.h"

class Interface

{

private:

Set\* set = new Set[2];

void CreatedTwoSet();

void ChangeSet(int& numberSet);

void AddElement(int& numberSet);

void CheckCommon();

void GetPower(int& numberSet);

void GetElements(int& numberSet);

void printMenuText();

int numberSet();

public:

void Go();

};

**//** Interface.cpp

#include "Interface.h"

#include "Set.h"

#include <iostream>

using std::cin;

using std::cout;

//Определение множеств по номеру

void Interface::ChangeSet(int& numberSet)

{

int power;

cout << "Enter the power for the set: ";

cin >> power;

while (power<0)

{

cout << "Error: enter a non-negative number:\n";

cin >> power;

}

cout << "Enter " << power << " numbers separated by a space: ";

int\* setElement = new int[power];

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

{

cin >> setElement[i];

}

set[numberSet].CreatedSet(setElement, power);

delete[] setElement;

cout << '\n';

}

void Interface::CreatedTwoSet()

{

for (int i = 0; i < 2; i++)

{

ChangeSet(i);

}

}

void Interface::AddElement(int& numberSet)

{

cout << "Введите элемент: ";

int element;

cin >> element;

set[numberSet].AddSetElement(element);

}

void Interface::CheckCommon()

{

if (set[0].CheckSetCommon(set[1]))

{

cout << "Sets have common elements\n";

}

else

{

cout << "Sets haven't common elements\n";

}

}

void Interface::GetPower(int& numberSet)

{

cout << set[numberSet].GetPower();

}

void Interface::GetElements(int& numberSet)

{

for (int i = 0; i < set[numberSet].GetPower(); i++)

{

cout << set[numberSet].GetElements(i)<< ' ';

}

cout << '\n';

}

int Interface::numberSet()

{

bool flag = true;

int number;

while (flag) {

cout << "Select the set to make changes to: 1 or 2\n";

cin >> number;

if (number <= 0 || number >= 3)

{

cout << "Error: Wrong number entered. Try again!\n";

}

else

{

flag = false;

}

}

cout << '\n';

return number - 1;

}

void Interface::printMenuText()

{

cout << "\n1 - Change one set.\n2 - Change both sets.\n"

"3 - Add a new element to the set.\n4 - Check whether there are common elements\n"

"5 - Find out The power from the set.\n6 - Find out which elements the set contains\n"

"7 - Help.\nAny other value - Exit.\n"

"===================================================================================\n";

}

void Interface::Go()

{

bool running = true;

CreatedTwoSet();

printMenuText();

while (running)

{

try

{

int input, number;

cin >> input;

switch (input) {

case 0: {

running = false;

cout << "GoodBuy\n";

break;

}

case 1: {

number = numberSet();

ChangeSet(number);

break;

}

case 2: {

CreatedTwoSet();

break;

}

case 3: {

number = numberSet();

AddElement(number);

break;

}

case 4: {

CheckCommon();

break;

}

case 5: {

number = numberSet();

GetPower(number);

break;

}

case 6: {

number = numberSet();

GetElements(number);

break;

}

case 7: {

printMenuText();

break;

}

default: {

running = false;

break;

}

}

}

catch (int err)

{

cout << "Error" << err << '\n';

}

}

}

//main.cpp

#include "Interface.h"

//#include <vld.H>

int main()

{

Interface interface;

interface.Go();

return 0;

}