Міністерство освіти і науки України

Національний технічний університет України „КПІ”

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки

інформації та управління

**ЗВІТ**

з комп’ютерного практикуму № 4

на тему :

„ ОПЕРАТОРНЫЕ И ДРУЖЕСТВЕННЫЕ ФУНКЦИИ С++”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Виконав**  **студент** |  | *ІП-63 Дроздович В.О.* |  |  |
|  |  | (№ групи, прізвище, ім’я, по батькові ) |  |  |
|  |  |  |  |  |
| **Прийняв** |  | *ас. Головченко М.М.* |  |  |
|  |  | (посада, прізвище, ім’я, по батькові ) |  |  |

Київ 2017

ЗМІСТ

[1 Мета роботи 3](#_Toc501593559)

[2 Постановка задачі 4](#_Toc501593560)

[3 Аналітичні викладки 5](#_Toc501593561)

[4 UML Діаграмма Класу 6](#_Toc501593562)

[5 Тексти програмного коду 7](#_Toc501593563)

[6 Результат роботи програми 10](#_Toc501593564)

[Висновки 11](#_Toc501593565)

# Мета роботи

*Мета роботи* – придбати навички складання елементарних програм із застосуванням операторних та дружніх функцій.

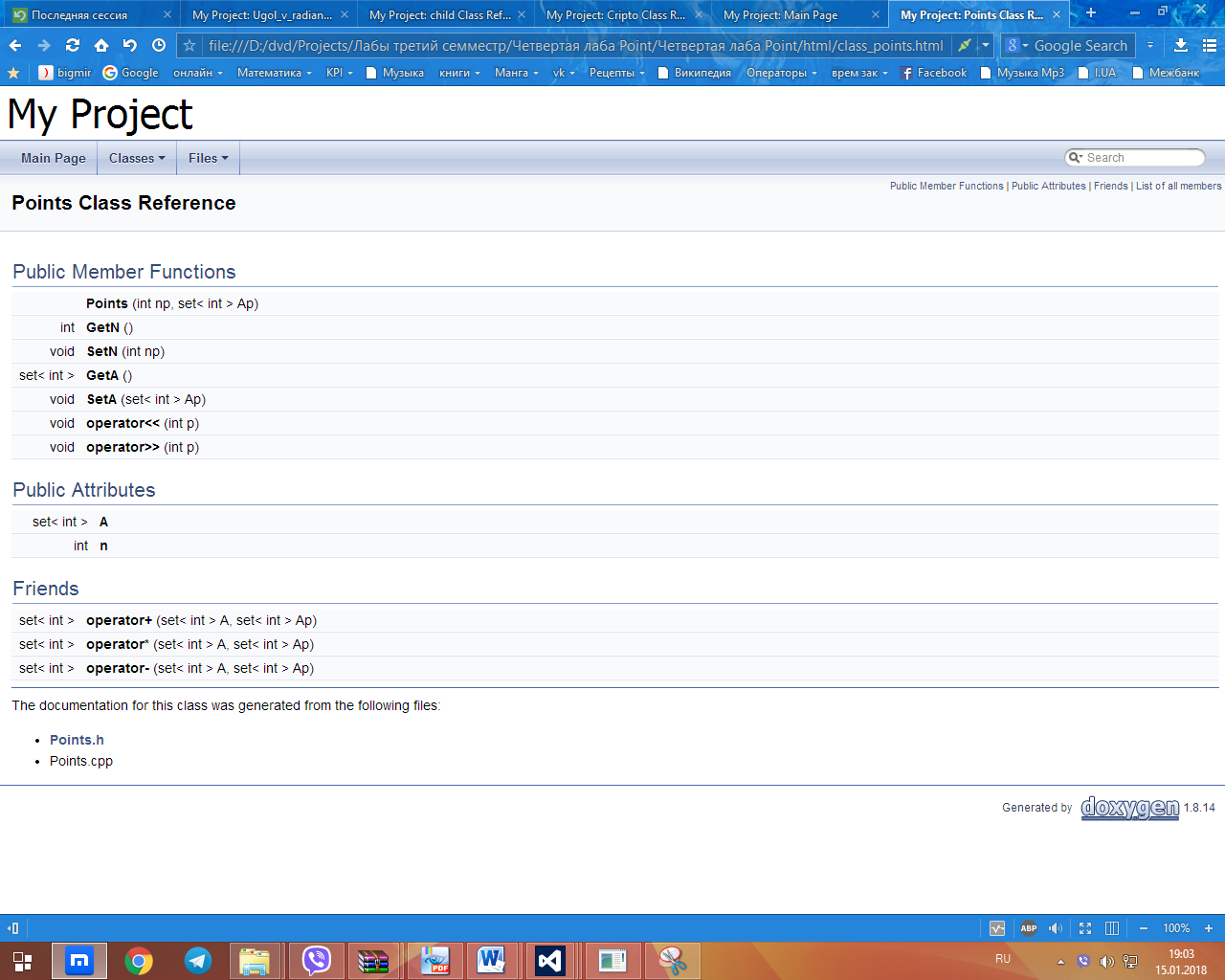
# Постановка задачі

Спроектировать класс «Point», который содержит точку на плоскости и методы доступа к ней. Спроектировать класс «Points», который содержит: множество точек на плоскости – A (объектов класса Point), количество точек – n. Для него определить: операцию добавления новой точки «<<» и операцию удаления точки «>>». Для него определить: операцию объединения «+», ее же в сокращенной форме, которая находит объединение точек без дубликатов; операцию пересечения «\*», ее же в сокращенной форме, которая находит пересечение точек без дубликатов; операцию разности «-», ее же в сокращенной форме, которая находит разность точек без дубликатов. При необходимости разрешается определять другие операции (например «=») и методы (например, getter, setter и прочее). Продемонстрировать каждую операцию.

# Аналітичні викладки

В ході виконання роботи ми використали дружні функції для перевантаження операторів.

# UML Діаграмма Класу



# Тексти програмного коду

#include <set>

#include <iterator>

#include <algorithm>

#include"Point.h"

class Points

{

public:

set<int> A;

int n;

Points ();

Points (int np, set<int> Ap);

int GetN ();

void SetN (int np);

set<int> GetA ();

void SetA (set<int> Ap);

void operator <<(int p);

void operator >>(int p);

friend set<int> operator +(set<int> A, set<int> Ap);

friend set<int> operator \*(set<int> A, set<int> Ap);

friend set<int> operator -(set<int> A, set<int> Ap);

~Points();

};

#include "Points.h"

Points::Points ()

{

n=10;

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

{

A.insert(i);

}

}

Points::Points(int np, set<int> Ap)

{

n=np;

A=Ap;

}

int Points::GetN ()

{

return n;

}

void Points::SetN (int np)

{

n=np;

}

set<int> Points::GetA()

{

return A;

}

void Points::SetA (set<int> Ap)

{

A=Ap;

}

void Points::operator<<(int p)

{

A.insert(p);

}

void Points::operator>>(int p)

{

A.erase(p);

}

set<int> operator+(set<int> A, set<int> Ap)

{

set<int> tmp;

std::set\_union(A.begin(),A.end(),Ap.begin(), Ap.end(),std::inserter(tmp, tmp.begin()));

return tmp;

}

set<int> operator\*(set<int> A, set<int> Ap)

{

set<int> tmp;

std::set\_intersection(A.begin(),A.end(),Ap.begin(), Ap.end(),std::inserter(tmp, tmp.begin()));

return tmp;

}

set<int> operator-(set<int> A, set<int> Ap)

{

set<int> tmp;

std::set\_difference(A.begin(),A.end(),Ap.begin(), Ap.end(),std::inserter(tmp, tmp.begin()));

return tmp;

}

Points::~Points()

{

n=NULL;

A.clear();

}

#include<iostream>

#include"Points.h"

int main()

{

setlocale(LC\_ALL, "Russian");

cout<<"1 множество -0 - 9. 2 множество 11. 3 множество 3. \n";

cout<<"1 строка - множество 0 - 9. \n";

cout<<"2 строка - множество 0 - 9 прибавить мнжество 2 (в нем одна переменная 11) \n";

cout<<"3 строка - от полученного во второй строке отнимаем множество 3 (в нем тоже один єлемент - 3). \n";

cout<<"4 строка - к полученном в третьей прибавляем множество 3 ( в нем опять же один элемент - 3). \n";

cout<<"5 строка - от полученного в четвертом делаем разницу на элемент 11 (оно же разница между полученным во 2 строке и 2 множествами). \n";

cout<<"Координаты точек : \n";

Points \*points = new Points();

set<int> Ap = points->A;

set<int>::iterator ik;

for(ik=points->A.begin(); ik!=points->A.end(); ++ik)

cout<<"("<<\*ik<<","<<\*ik-10<<") ";

cout<<endl;

\*points<<11;

cout<<"Добавили элемент 11 \n";

for(ik=points->A.begin(); ik!=points->A.end(); ++ik)

cout<<"("<<\*ik<<","<<\*ik-10<<") ";

cout<<endl;

\*points>>3;

cout<<"Удалили элемент 3 \n";

for(ik=points->A.begin(); ik!=points->A.end(); ++ik)

cout<<"("<<\*ik<<","<<\*ik-10<<") ";

cout<<endl;

points->A=points->A+Ap;

cout<<"Объединили \n";

for(ik=points->A.begin(); ik!=points->A.end(); ++ik)

cout<<"("<<\*ik<<","<<\*ik-10<<") ";

cout<<endl;

points->A=points->A\*Ap;

cout<<"Пересечение \n";

for(ik=points->A.begin(); ik!=points->A.end(); ++ik)

cout<<"("<<\*ik<<","<<\*ik-10<<") ";

cout<<endl;

points->A=points->A-Ap;

cout<<"Разница \n";

for(ik=points->A.begin(); ik!=points->A.end(); ++ik)

cout<<"("<<\*ik<<","<<\*ik-10<<") ";

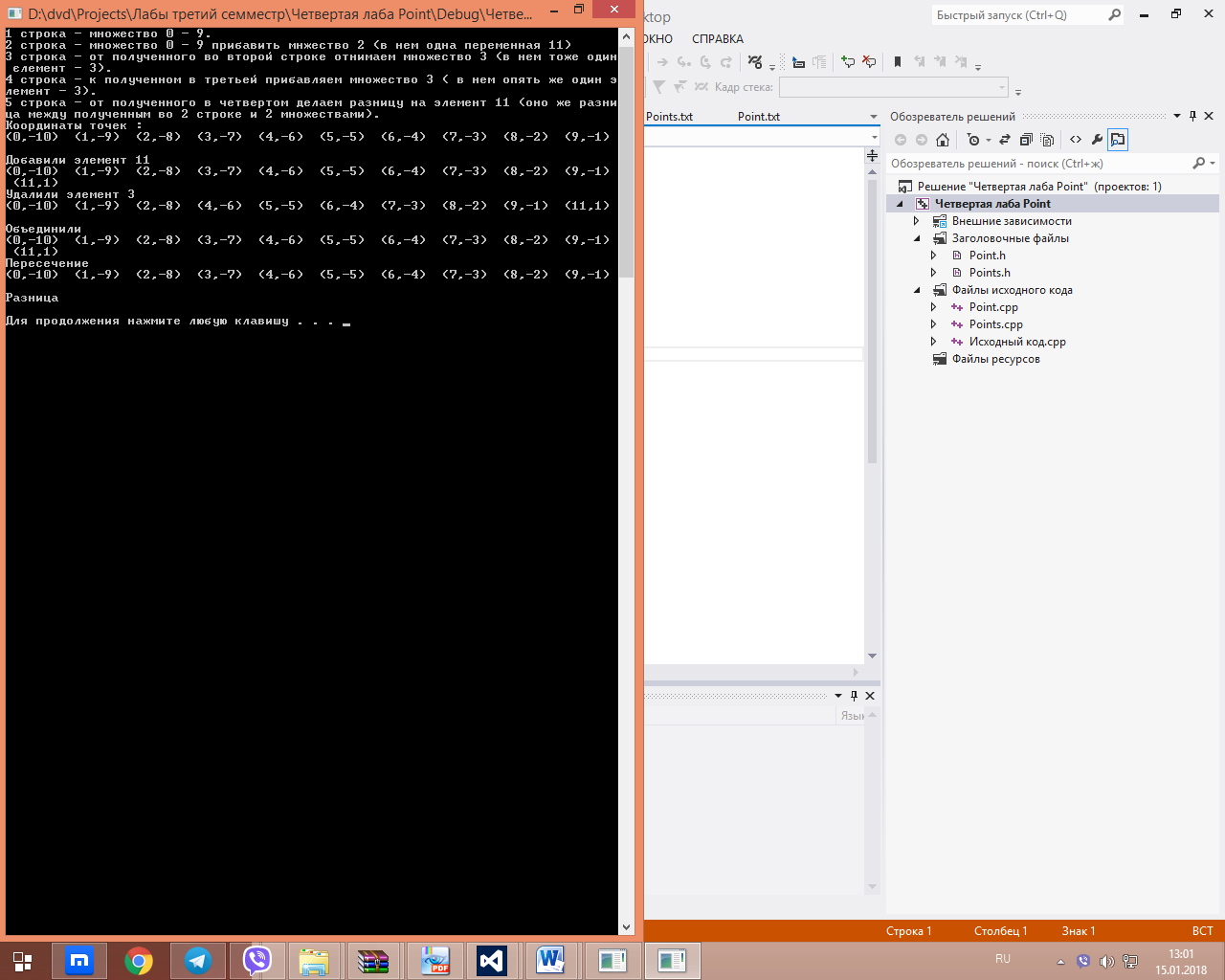
cout<<endl;

system("pause");

return 0;

}

# Результат роботи програми



Висновки

В ході виконання лабораторної роботи я отримав навички застосування дружніх та операторних функцій.