

Міністерство освіти і науки України  
Національний технічний університет України «Київський політехнічний  
інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 4 з дисципліни  
«Основи програмування 2. Модульне програмування»

«Перевантаження операторів»

Варіант №32

Виконав студент ІП-14 Шляхтун Денис Михайлович  
(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна  
(прізвище, ім'я, по батькові)

Київ 2022

## Лабораторна робота №4

**Тема:** перевантаження операторів.

**Мета:** вивчити механізми створення класів з використанням перевантажених операторів.

### Хід роботи

#### Задача.

32. Визначити клас "Трикутник", членами якого є координати вершин трикутника в просторі. Реалізувати для нього декілька конструкторів, геттери, методи обчислення периметра трикутника. Перевантажити оператори: префіксний "++" / постфіксний "++" - для інкрементування усіх x-координат і усіх y-координат вершин трикутника відповідно, "+=" – для збільшення усіх координат вершин трикутника на вказану величину. Створити три трикутника (T1, T2, T3), використовуючи різні конструктори. Інкрементувати x-координати вершин трикутника T1 і y-координати вершин трикутника T2. Збільшити координати вершин трикутника T3 на вказану величину. Серед трикутників T1, T2, T3 визначити трикутник, що має найбільший периметр.

#### Виконання задачі.

#### Випробування коду на C++.

*Код:*

main.cpp

```
17     #include "module.h"
18     using namespace std;
19
20     int main()
21     {
22         //створення першого трикутника
23         cout << "The first triangle: ";
24         TPoint p[3];
25         pointArrInput(p, 3);
26         TTriangle T1(p[0], p[1], p[2]);
27         T1.output();
28
29         //створення другого трикутника
30         cout << "\nThe second triangle: ";
31         int len[3];
32         lenArrInput(len, 3);
33         TTriangle T2(len[0], len[1], len[2]);
34         T2.output();
35
36         //створення третього трикутника
37         cout << "\nThe third triangle: ";
38         TTriangle T3(T2);
39         T3.output();
40
41         //інкрементування x-координат
42         ++T1;    T1.output();
43
44         //інкрементування y-координат
45         T2++;    T2.output();
46
47         //збільшення координат на задану величину
48         T3 += addInput();    T3.output();
49
50         //визначення максимального периметру
51         maxPerimeter(T1, T2, T3);
52
53         system("pause");
54     }
```

## TTriangle.h

```
1  #pragma once
2  #include <iostream>
3  #include <iomanip>
4  #include "TPoint.h"
5
6  class TTriangle
7  {
8      TPoint p1, p2, p3; //вершини трикутника
9      const int num;     //порядковий номер трикутника
10     static int count;   //загальна кількість створених об'єктів класу
11 public:
12     TTriangle(TPoint a, TPoint b, TPoint c); //конструктор зі списком ініціалізації
13     TTriangle(int a, int b, int c);          //конструктор з параметрами
14     TTriangle(TTriangle&);                   //конструктор копіювання
15     TPoint getCertainVertex(int);           //getter певної вершини трикутника
16     void getAllVertices(TPoint&, TPoint&, TPoint&); //getter для всіх вершин трикутника
17     int getNum() { return num; };           //getter для порядкового номера
18     static int getCount() { return TTriangle::count; }; //getter для кількості створених об'єктів
19     float perimeter(); //обчислення периметру трикутника
20     TTriangle operator++();                  //перевантаження префіксного "++"
21     TTriangle operator++(int notused);       //перевантаження постфіксного "++"
22     const TTriangle operator+=(const float add); //перевантаження "+="
23     void output(); //виведення даних про трикутник
24     ~TTriangle(); //деструктор
25 }
```

## TTriangle.cpp

```
1  #include "TTriangle.h"
2
3  //статична змінна кількості створених об'єктів класу
4  int TTriangle::count;
5
6  //конструктор зі списком ініціалізації
7  TTriangle::TTriangle(TPoint a, TPoint b, TPoint c)
8  : p1(a), p2(b), p3(c), num(++count) {}
9
10 //конструктор з параметрами за замовчуванням шляхом відкладення відрізків на осях координат
11 TTriangle::TTriangle(int a = 1, int b = 1, int c = 1) : num(++count)
12 {
13     p1.set(a, 0, 0);
14     p2.set(0, b, 0);
15     p3.set(0, 0, c);
16 }
17
18 //конструктор копіювання
19 TTriangle::TTriangle(TTriangle& T) : num(++count)
20 {
21     T.getAllVertices(p1, p2, p3);
22 }
```

```

24     //геттер певної вершини трикутника
25     TPoint TTriangle::getCertainVertex(int n = 1)
26     {
27     switch (n)
28     {
29     case 1:
30         return p1;
31     case 2:
32         return p2;
33     case 3:
34         return p3;
35     default:
36         cout << "\nWrong input detected. Value of first verticle is returned.";
37         return p1;
38     }
39     }
40
41     //геттер для всіх вершин трикутника
42     void TTriangle::getAllVertices(TPoint& a, TPoint& b, TPoint& c)
43     {
44         a = p1; b = p2; c = p3;
45     }
46
47     //обчислення периметру трикутника
48     float TTriangle::perimeter()
49     {
50         return sideLength(p1, p2) + sideLength(p2, p3) + sideLength(p1, p3);
51     }

```

```

53     //перевантаження префіксного "++" для збільшення x-координат
54     TTriangle TTriangle::operator++()
55     {
56         p1.x++; p2.x++; p3.x++;
57         return *this;
58     }
59
60     //перевантаження постфіксного "++" для збільшення y-координат
61     TTriangle TTriangle::operator++(int notused)
62     {
63         p1.y++; p2.y++; p3.y++;
64         return *this;
65     }
66
67     //перевантаження "+=" для збільшення всіх координат на задану величину
68     const TTriangle TTriangle::operator+=(const float add)
69     {
70         p1 += add; p2 += add; p3 += add;
71         return *this;
72     }
73
74     //виведення даних про трикутник
75     void TTriangle::output()
76     {
77         cout << "\nTriangle #" << num <<
78             "\nP1:"; p1.output();
79         cout << "\nP2:"; p2.output();
80         cout << "\nP3:"; p3.output();
81         cout << "\nGeneral number of triangles: " << count << endl;
82     }

```

```

84     //деструктор
85     TTriangle::~TTriangle()
86     {
87         count--;
88     }

```

## TPoint.h

```
1  #pragma once
2  #include <iostream>
3  #include <iomanip>
4  using namespace std;
5
6  class TPoint
7  {
8      float x, y, z; //координати точки
9  public:
10     void set(int a, int b, int c); //задання значення координат точки
11     const TPoint operator+=(const float add); //перевантаження оператора "+="
12     void input(); //введення координат точки
13     void output(); //виведення координат точки
14     friend float sideLength(TPoint a, TPoint b); //визначення довжини між двома точками
15     friend class TTriangle; //дружній клас трикутник
16 };
```

## TPoint.cpp

```
1   #include "TPoint.h"
2
3   //задання значення координат точки
4   void TPoint::set(int a, int b, int c)
5   {
6       x = a; y = b; z = c;
7   }
8
9   /*перевантаження оператора "+" для
10  збільшення всіх координат на задану величину*/
11  const TPoint TPoint::operator+=(const float add)
12  {
13      x += add; y += add; z += add; return *this;
14  }
15
16  //введення координат точки
17  void TPoint::input()
18  {
19      cout << "\nEnter x: ";
20      cin >> x;
21      cout << "Enter y: ";
22      cin >> y;
23      cout << "Enter z: ";
24      cin >> z;
25  }
26
27  //виведення координат точки
28  void TPoint::output()
29  {
30      cout << setw(4) << x
31          << setw(4) << y
32          << setw(4) << z;
33  }
```

## module.h

```
1   #pragma once
2   #include <iostream>
3   #include <iomanip>
4   #include "TTriangle.h"
5
6   float sideLength(TPoint a, TPoint b); //визначення відстані між двома точками
7   void pointArrInput(TPoint* arr, int n); //введення масиву точок
8   void lenArrInput(int* arr, int n); //введення масиву відстаней
9   float addInput(); //введення значення для додавання до координат трикутника
10  int maxPerimeter(TTriangle&, TTriangle&, TTriangle&); //найбільший периметр
```



## module.cpp

```
1  #include "module.h"
2
3  //визначення відстані між двома точками
4  float sideLength(TPoint a, TPoint b)
5  {
6      return sqrt((a.x - b.x) * (a.x - b.x) + (a.y - b.y) * (a.y - b.y) + (a.z - b.z) * (a.z - b.z));
7  }
8
9  //введення масиву точок
10 void pointArrInput(TPoint* arr, int n)
11 {
12     for (int i = 0; i < n; i++)
13     {
14         cout << "\nEnter coordinates for point #" << i + 1;
15         arr[i].input();
16     }
17 }
18
19 //введення масиву відстаней
20 void lenArrInput(int* arr, int n)
21 {
22     for (int i = 0; i < n; i++)
23     {
24         cout << "\nEnter distance for point #" << i + 1 << " : ";
25         cin >> arr[i];
26     }
27 }
28
29 //введення значення для додавання до координат трикутника
30 float addInput()
31 {
32     float add;
33     cout << "\nEnter add value: ";
34     cin >> add;
35     return add;
36 }
```

```

38 //знаходження найбільшого периметру
39 int maxPerimeter(TTriangle& T1, TTriangle& T2, TTriangle& T3)
40 {
41     float per,
42         per1 = T1.perimeter(),
43         per2 = T2.perimeter(),
44         per3 = T3.perimeter();
45     int pos;
46     cout << "\nPerimeter of triangle #" << T1.getNum() << ": " << per1;
47     cout << "\nPerimeter of triangle #" << T2.getNum() << ": " << per2;
48     cout << "\nPerimeter of triangle #" << T3.getNum() << ": " << per3;
49
50     if (per1 >= per2)
51     {
52         per = per1;
53         pos = T1.getNum();
54     }
55     else
56     {
57         per = per2;
58         pos = T2.getNum();
59     }
60
61     if(per3>per)
62     {
63         per = per3;
64         pos = T3.getNum();
65     }
66
67     cout << "\nTriangle #" << pos << " has max perimeter: " << per << endl;
68
69     return pos;
70 }

```

*Результат:*

```
E:\university\II semest...
The first triangle:
Enter coordinates for point #1
Enter x: 1
Enter y: 1
Enter z: 1

Enter coordinates for point #2
Enter x: 2
Enter y: 2
Enter z: 2

Enter coordinates for point #3
Enter x: 3
Enter y: 3
Enter z: 3

Triangle #1
P1: 1 1 1
P2: 2 2 2
P3: 3 3 3
General number of triangles: 1

The second triangle:
Enter distance for point #1 : 1

Enter distance for point #2 : 2

Enter distance for point #3 : 3

Triangle #2
P1: 1 0 0
P2: 0 2 0
P3: 0 0 3
General number of triangles: 2

The third triangle:
Triangle #3
P1: 1 0 0
P2: 0 2 0
P3: 0 0 3
General number of triangles: 3

Select Microsoft Visua...
Triangle #1
P1: 2 1 1
P2: 3 2 2
P3: 4 3 3
General number of triangles: 3

Triangle #2
P1: 1 1 0
P2: 0 3 0
P3: 0 1 3
General number of triangles: 3

Enter add value: 2

Triangle #3
P1: 3 2 2
P2: 2 4 2
P3: 2 2 5
General number of triangles: 3

Perimeter of triangle #1: 6.9282
Perimeter of triangle #2: 9.0039
Perimeter of triangle #3: 9.0039
Triangle #2 has max perimeter: 9.0039
Press any key to continue . . .

E:\university\II semester\basics of pr
og\Lab4\CPPLab4\Debug\CPPLab4.exe (pro
cess 16060) exited with code 0.
To automatically close the console whe
```

**Висновок:** При виконанні лабораторної роботи було вивчено механізми створення класів з використанням перевантажених операторів у мові програмування C++ та на основі набутих навичок було виконано задачу. Під час виконання лабораторної роботи було використано різні типи конструкторів (з параметрами, списком ініціалізацій та копіювання) та було здійснено перевантаження операторів: префіксного «++», суфіксного «++» та «+=».