

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

Кафедра ІІІ

Звіт

з лабораторної роботи № 3 з дисципліни
«Основи програмування 2. Модульне програмування»

„Класи і об'єкти”

Варіант 8

Виконав(ла)

Дацьо Іван Іванович
(шифр, прізвище, ім'я, по батькові)

Перевірів

Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 1

Текстові файли

Мета – вивчити механізм створення і використання класів.

Варіант 8

Задача

8. Розробити клас "конус", який заданий координатами центру основи, координатами вершини та радіусом основи. Створити масив об'єктів даного класу. Визначити конус з найбільшою твірною.

Код:

C++

```
main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
1 #include "functions.h"
2 int main(){
3     int size = inputSize();
4     Cone * cones = input(size);
5     int index = getINdex(cones,size);
6     outAllCones(cones,size);
7     outMax(cones, max index);
8
9 }
10
```

```
main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
1 #include "functions.h"
2 #include "cone.h"
3 int inputSize(){
4     int n;
5     cout <<"Enter number of cones :\n ";
6     cin >> n;
7     cin.ignore();
8     while (n < 1 ){
9         cout <<"Enter the correct number of cones :"<<endl;
10        cin >> n;
11    }
12    return n;
13 }
14 double inputCoo(){
15     string c;
16     cin >>c;
17     cin.ignore();
18     /*while (isdigit(c) == 0 ){
19         cout <<"Enter the correct :";
20         cin >> c;
21     }*/
22     bool flag = verCoo(str:c);
23     while (flag == false){
24         cout <<"Enter the correct :";
25         cin >> c;
26         cin.ignore();
27         flag = verCoo( str: c);
28     }
29
30     return stod( str: c);
31 }
```

```

main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
28     }
29
30     return stod(str.c);
31 }
32 bool isNumb(const string& str){
33     for (char const &c : str){
34         if (isdigit(c) == 0) return false;
35     }
36     return true;
37 }
38 }
39
40 bool verCoo(string str){
41     bool flag = false;
42     for (int i = 0; i < str.size(); i++){
43         if (str[i] == '.') {
44             string bef = str.substr(pos: 0, n: i);
45             string aft = str.substr(pos: i, n: str.size()+1);
46             if (isNumb(str.bef) && isNumb(str.aft)){
47                 flag = false;
48             }
49             else {
50                 flag = true ;
51             }
52         }
53     }
54 }
55 }
56 return flag;
57 }
58 Cone * input(int size){

```

```

main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
55     }
56     return flag;
57 }
58 Cone * input(int size){
59     Cone * cones = new Cone[size];
60     for (int i = 0; i < size; i++){
61         cout << "Cone "<< i+1 << ": " << endl;
62         Point points[2];
63         for (int j = 0; j < 2; j++){
64             if (j == 0){
65                 cout << "Point 0(center of foundation) :\n";
66             }
67             else{
68                 cout << "Point A(the top of the cone) :\n";
69             }
70             cout << "Enter x coordinate : \n";
71             double x = inputCoo();
72             cout << "Enter y coordinate : \n";
73             double y = inputCoo();
74             cout << "Enter z coordinate : \n";
75             double z = inputCoo();
76             points[j] = Point(x,y,z);
77         }
78         cout << "Enter the radius of the base of the cone :\n";
79         double r = inputCoo();
80         cones[i] = Cone(points , r);
81     }
82     return cones;
83 }
84 int getINdex(Cone * cones , int size){
85     int index;

```

```
main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
81     }
82     return cones;
83 }
84 int getINdex(Cone * cones , int size){
85     int index;
86     for (int i = 1; i < size; i++){
87         if (cones[i].getGenerator() > cones[0].getGenerator()){
88             index = i;
89         }
90     }
91     return index;
92 }
93 void outAllCones(Cone* cones ,int size){
94     cout <<"All cones :"<<endl;
95     for (int i = 0; i < size; i++){
96         cout <<"Cone " <<i + 1<<":"<<endl;
97         cones[i].out();
98     }
99 }
100 void outMax(Cone* cones , int max){
101     cout <<"Max generator have this cone : ("<<max + 1<<") : "<<endl;
102     cones[max].out();
103 }
104
105
106
```

```
main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
1  #pragma once
2  #include <string>
3  // #include <ctype.h>
4  #include "cone.h"
5  using namespace std;
6  int inputSize();
7  double inputCoo();
8  bool verCoo(string str);
9  bool isNumb(const string& str);
10 Cone * input(int size);
11 int getINdex(Cone * cones , int size);
12 void outAllCones(Cone* cones ,int size);
13 void outMax(Cone* cones , int max);
14
```

```

1  #include "cone.h"
2  Cone::Cone(){};
3  Cone::Cone(Point Or, Point A, Radius r) {
4      this->Or = Point(Or.getX(), Or.getY(), Or.getZ());
5      this->A = Point(A.getX(), A.getY(), A.getZ());
6      this->r = Radius(r.getR());
7      this->l = gtgenerator();
8
9
10 }
11 Cone::Cone(Point points[2], Radius r) {
12     this->Or = Point(points[0].getX(), points[0].getY(), points[0].getZ());
13     this->A = Point(points[1].getX(), points[1].getY(), points[1].getZ());
14     this->r = Radius(r.getR());
15     this->l = gtgenerator();
16
17
18 }
19 double Cone::gtgenerator() {
20     double gen = pow(x (Or.getX()-A.getX()), y: 2) + pow(x (Or.getY()-A.getY()), y: 2)+ pow(x Or.getZ() - A.getZ(), y: 2)+ pow(x (r
21     return sqrt(x gen);
22 }
23 double Cone::getGenerator() {
24     return l;
25 }
26 void Cone::out(){
27     cout << "Point 0 :";
28     Or.outPoint();
29     cout << "\t";
30     cout << "Point A :";
31     A.outPoint();

```

```

16
17
18 }
19 double Cone::gtgenerator() {
20     double gen = pow(x (Or.getX()-A.getX()), y: 2) + pow(x (Or.getY()-A.getY()), y: 2)+ pow(x Or.getZ() - A.getZ(), y: 2)+ pow(x (r
21     return sqrt(x gen);
22 }
23 double Cone::getGenerator() {
24     return l;
25 }
26 void Cone::out(){
27     cout << "Point 0 :";
28     Or.outPoint();
29     cout << "\t";
30     cout << "Point A :";
31     A.outPoint();
32     cout << "\t";
33     cout << "Radius :";
34     r.outRadius();
35     cout << "\t";
36     cout << "Generator : "<< getGenerator() << endl;
37
38
39 }
40

```

```
main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
1  #pragma once
2  #include "subclass.h"
3  class Cone{
4      Point Or , A; //Or - центр основы ; A - вершина
5      Radius r; // r - радиус основы
6      double l ; //l - высота
7      double getGenerator();
8
9  public:
10     Cone();
11     Cone(Point Or , Point A , Radius r);
12     Cone(Point points[2], Radius r);
13     double getGenerator();
14     void out();
15
16 };
17
```

```
main.cpp x functions.cpp x functions.h x cone.cpp x cone.h x subclass.cpp x subclass.h x
1  #include "subclass.h"
2  Point::Point() {}
3  Point::Point(double x , double y, double z){
4      this->x = x;
5      this->y = y;
6      this->z = z;
7  }
8  void Point::setX(double x) {
9      this->x = x;
10 }
11 void Point::setY(double y) {
12     this->y = y;
13 }
14 void Point::setZ(double z) {
15     this->z = z;
16 }
17 double Point::getX() {
18     return this->x;
19 }
20 double Point::getY() {
21     return this->y;
22 }
23 double Point::getZ() {
24     return this->z ;
25 }
26 Radius::Radius() {}
27 Radius::Radius(double R) {
28     this->R = R;
29 }
30 void Radius::setR(double R) {
31     this->R = R;
32 }
```

```

12   this->y = y;
13 }
14 void Point::setZ(double z) {
15     this->z = z;
16 }
17 double Point::getX() {
18     return this->x;
19 }
20 double Point::getY() {
21     return this->y;
22 }
23 double Point::getZ() {
24     return this->z;
25 }
26 Radius::Radius() {};
27 Radius::Radius(double R) {
28     this->R = R;
29 }
30 void Radius::setR(double R) {
31     this->R = R;
32 }
33 double Radius::getR() {
34     return this->R;
35 }
36
37

```

```

1  #pragma once
2  #include <iostream>
3  #include <cmath>
4  using namespace std;
5  class Point{//класс точка для задания координат вершины та координат центра основы
6      double x;
7      double y;
8      double z;
9  public:
10     Point();
11     Point(double ,double ,double );
12     void setX(double );
13     void setY(double );
14     void setZ(double );
15     double getX();
16     double getY();
17     double getZ();
18     void outPoint(){cout <<"X = "<<x<<"Y = "<<y<<"Z = "<<z;}//вывести координаты точки
19 };
20 class Radius{
21     double R;
22 public:
23     Radius();
24     Radius(double );
25     void setR(double );
26     double getR();
27     void outRadius(){cout <<"Radius = "<<R;}
28 };
29

```

Тестування :

C++

```

C:\Op-1.2\2-semLab\lab3\cmake-build-debug\lab3.exe
Enter number of cones :
2
Cone 1:
Point 0(center of foundation) :
Enter x coordinate :
15.90
Enter y coordinate :
60.90
Enter z coordinate :
40
Enter the correct :40;30
Enter the correct :39.90
Point A(the top of the cone) :
Enter x coordinate :
14.90
Enter y coordinate :
123.0-
Enter z coordinate :
13.90
Enter the radius of the base of the cone :
13.90
Cone 2:
Point 0(center of foundation) :
Enter x coordinate :
12.90
Enter y coordinate :
132.90
Enter z coordinate :
21.89

```

```

Cone 2:
Point 0(center of foundation) :
Enter x coordinate :
12.90
Enter y coordinate :
132.90
Enter z coordinate :
21.89
Point A(the top of the cone) :
Enter x coordinate :
34.90
Enter y coordinate :
234.45
Enter z coordinate :
24.90
Enter the radius of the base of the cone :
5.0
All cones :
Cone 1:
Point 0 :X = 15.9;Y = 60.9;Z = 39.9    Point A :X = 14.9;Y = 123;Z = 13.9    Radius :Radius = 13.9    Generator :68.75
04
Cone 2:
Point 0 :X = 12.9;Y = 132.9;Z = 21.89    Point A :X = 34.9;Y = 234.45;Z = 24.9    Radius :Radius = 5    Generator :104.0
7
Max generator have this cone : 2 :
Point 0 :X = 12.9;Y = 132.9;Z = 21.89    Point A :X = 34.9;Y = 234.45;Z = 24.9    Radius :Radius = 5    Generator :104.0
7

```

Висновок : При виконанні лабораторної роботи було вивчено особливості роботи з класами та об'єктами та використано ці навички під час написання програм. В результаті роботи було отримано програму яка обчислює твірну конуса при відомих координатах вершини , координатам основи , та радіусом основи . Також при декількох конусах програма виводить конус з найбільшою твірною.