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 №6 з дисципліни

«Основи програмування»

Варіант 14

Виконав студент Кашка Максим Сергійович

( прізвище, ім'я, по батькові)

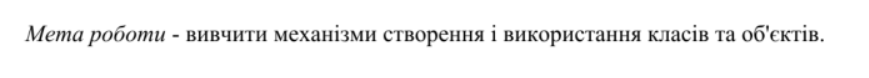
Перевірив викладач Вітковська Ірина Іванівна

( прізвище, ім'я, по батькові)

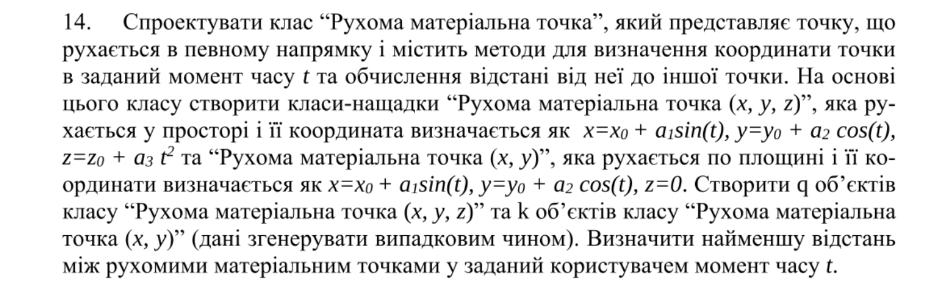
Київ 2022

Лабораторна робота №5

[УСПАДКУВАННЯ ТА ПОЛІМОРФІЗМ](http://ipi.kpi.ua/study/ii-semestr/laboratorni-roboti/laboratorna-robota-4-uspadkuvannya-ta-pooimorfizmn/)



Умова задачі:



Код задачі C++:

Sem2 Lab5C++:

#include "func.h"

int main()

{

srand(time(NULL));

double t = inputT();

int m = input();

PointXYZ\* arrXYZ = XYZ\_Array(m, t);

cout << "--Array of PointXYZ--" << endl;

outputPoint(arrXYZ, m, t);

int n = input();

PointXY\* arrXY = XY\_Array(n, t);

cout << "--Array of PointXY--" << endl;

outputPoint(arrXY, n, t);

double min1 = MinDistance(arrXYZ, m);

cout << "Min distance in array\_xyz is " << min1 << endl;

double min2 = MinDistance(arrXY, n);

cout << "Min distance in array\_xy is " << min2 << endl;

delete[]arrXYZ;

delete[]arrXY;

}

сlass.cpp

#include "class.h"

double Point::GetX()

{

return x;

}

double Point::GetY()

{

return y;

}

double Point::GetZ()

{

return z;

}

string Point::GetStart()

{

return string("(") + to\_string(x0) + string(", ") + to\_string(y0) + string(", ") + to\_string(z0) + string(")");

}

string Point::GetFinal()

{

return string("(") + to\_string(x) + string(", ") + to\_string(y) + string(", ") + to\_string(z) + string(")");

}

double Point::GetDistance(double x2, double y2, double z2)

{

return sqrt(pow(x2 - x, 2) + pow(y2 - y, 2) + pow(z2 - z, 2));

}

PointXYZ::PointXYZ()

{

double min = -100.0;

double max = 100.0;

x0 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

y0 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

z0 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

x = y = z = 0;

}

void PointXYZ::coordinates(int time)

{

double min = -5.0;

double max = 5.0;

double a1 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

double a2 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

double a3 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

x = x0 + a1 \* sin(time);

y = y0 + a2 \* cos(time);

z = z0 + a3 \* pow(time, 2);

}

PointXY::PointXY()

{

double min = -100.0;

double max = 100.0;

x0 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

y0 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

z0 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

}

void PointXY::coordinates(int time)

{

double min = -5.0;

double max = 5.0;

double a1 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

double a2 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

double a3 = (double)(rand()) / RAND\_MAX \* (max - min) + min;

x = x0 + a1 \* sin(time);

y = y0 + a2 \* cos(time);

z = 0;

}

class.h

#pragma once

#include <iostream>

#include <sstream>

using namespace std;

class Point

{

protected:

double x0;

double x;

double y0;

double y;

double z0;

double z;

public:

virtual string GetStart();

virtual string GetFinal();

virtual double GetDistance(double, double, double);

virtual void coordinates(int) = 0;

virtual double GetX();

virtual double GetY();

virtual double GetZ();

};

class PointXYZ : public Point

{

public:

PointXYZ();

void coordinates(int);

};

class PointXY : public Point

{

public:

PointXY();

void coordinates(int);

};

funct.cpp

#include "func.h"

PointXYZ\* XYZ\_Array(int size, int time)

{

PointXYZ\* Array = new PointXYZ[size];

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

{

PointXYZ point;

point.coordinates(time);

Array[i] = point;

}

return Array;

}

PointXY\* XY\_Array(int size, int time)

{

PointXY\* Array = new PointXY[size];

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

{

PointXY point;

point.coordinates(time);

Array[i] = point;

}

return Array;

}

int input()

{

int num = 0;

do

{

cout << "Enter the number of points: "; cin >> num;

} while (num <= 0);

return num;

}

int inputT()

{

cout << "Enter t in seconds: ";

int time;

cin >> time;

while (true)

{

if (time > 0)return time;

else cin >> time;

}

}

void outputPoint(Point\* Array, int size, int time)

{

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

{

cout << "M" << i + 1 << ":" << endl;

cout << "Initial coordinates ofthe point: " << Array[i].GetStart() << endl;

cout << "Coordinates of the point in " << time << " seconds: " << Array[i].GetFinal() << endl;

cout << endl;

}

}

double MinDistance(Point\* Array, int size)

{

double minD = Array[0].GetDistance(Array[1].GetX(), Array[1].GetY(), Array[1].GetZ());

for (int i = 0; i < size - 1; i++)

{

for (int j = i + 1; j < size; j++)

{

double temp = Array[i].GetDistance(Array[j].GetX(), Array[j].GetY(), Array[j].GetZ());

if (temp < minD)minD = temp;

}

}

return minD;

}

func.cpp

#include "GeometricSeries.hpp"

#include <cmath>

GeometricSeries::GeometricSeries(int firstMember, int step)

    :TSeries(firstMember, step)

{

}

int GeometricSeries::FindMember(int memberNum)

{

    return firstMember \* pow(step, memberNum - 1);

}

int GeometricSeries::FindSum(int num)

{

    return firstMember \* (pow(step, num) - 1) / (step - 1);

}

funct.h

#pragma once

#include "class.h"

#include <iostream>

#include <sstream>

using namespace std;

int input();

int inputT();

PointXYZ\* XYZ\_Array(int, int);

PointXY\* XY\_Array(int, int);

void outputPoint(Point\*, int, int);

double MinDistance(Point\*, int);

Код задачі Python:

Sem2 Lab5Pyt:

**from func import \***  
**from point import \***  
  
  
**t = inputT()**  
**k = inputN()**  
**arrXYZ = []**  
**for i in range(k):**  
 **arrXYZ.append(PointXYZ())**  
 **arrXYZ[i].GetCoordinates(t)**  
**print("\nCoordinates of arrXYZ: ")**  
**outputCoordinates(arrXYZ)**  
**print()**  
**n = inputN()**  
**arrXY = []**  
**for i in range(n):**  
 **arrXY.append(PointXY())**  
 **arrXY[i].GetCoordinates(t)**  
**print()**  
**print("\nCoordinates of arrXY: ")**  
**outputCoordinates(arrXY)**  
  
**minDXYZ= MinD(arrXYZ)**  
**print(f"\nMin distance in arrXYZ is {round(minDXYZ, 3)}")**  
  
**minDXY= MinD(arrXY)**  
**print(f"\nMin distance in arrXY is {round(minDXY, 3)}")**

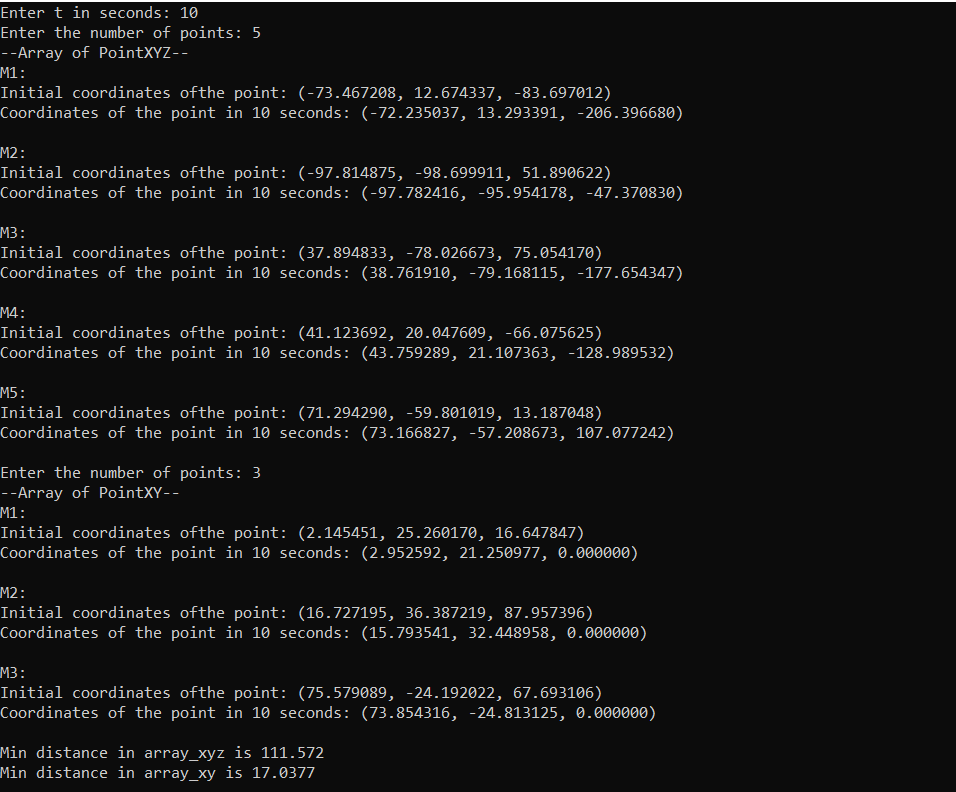
point.py

from abc import ABC, abstractmethod  
import random  
import math  
  
  
class Point(ABC):  
 @abstractmethod  
 def GetCoordinates(self, t):  
 pass  
  
 @abstractmethod  
 def GetDistance(self, x2, y2, z2):  
 pass  
  
 @abstractmethod  
 def InitialCoordinates(self):  
 pass  
  
 @abstractmethod  
 def FinalCoordinates(self):  
 pass  
  
  
class PointXYZ(Point):  
  
 def GetX(self):  
 return self.\_\_x  
  
 def GetX0(self):  
 return self.\_\_x0  
  
 def GetY(self):  
 return self.\_\_y  
  
 def GetY0(self):  
 return self.\_\_y0  
  
 def GetZ(self):  
 return self.\_\_z  
  
 def GetZ0(self):  
 return self.\_\_z0  
  
 def \_\_init\_\_(self):  
 self.\_\_x = 0  
 self.\_\_y = 0  
 self.\_\_z = 0  
 self.\_\_x0 = round(random.uniform(-100, 100), 1)  
 self.\_\_y0 = round(random.uniform(-100, 100), 1)  
 self.\_\_z0 = round(random.uniform(-100, 100), 1)  
  
 def GetCoordinates(self, t):  
 a1 = round(random.uniform(-5, 5), 1)  
 a2 = round(random.uniform(-5, 5), 1)  
 a3 = round(random.uniform(-5, 5), 1)  
  
 self.\_\_x = round(self.\_\_x0 + a1 \* math.sin(t), 2)  
 self.\_\_y = round(self.\_\_y0 + a2 \* math.cos(t), 2)  
 self.\_\_z = round(self.\_\_z0 + a3 \* math.pow(t, 2), 2)  
 return self.\_\_x, self.\_\_y, self.\_\_z,  
  
 def GetDistance(self, x2, y2, z2):  
 return math.sqrt(math.pow(x2 - self.\_\_x, 2) + math.pow(y2 - self.\_\_y, 2) + math.pow(z2 - self.\_\_z, 2))  
  
 def InitialCoordinates(self):  
 print(f"Initial Coordinates(", self.\_\_x0, ',', self.\_\_y0, ',', self.\_\_z0, ')')  
  
 def FinalCoordinates(self):  
 print("Final Coordinates: (", self.\_\_x, ',', self.\_\_y, ',', self.\_\_z, ')')  
  
  
class PointXY(Point):  
  
 def GetX(self):  
 return self.\_\_x  
  
 def GetX0(self):  
 return self.\_\_x0  
  
 def GetY(self):  
 return self.\_\_y  
  
 def GetY0(self):  
 return self.\_\_y0  
  
 def GetZ(self):  
 return self.\_\_z  
  
 def GetZ0(self):  
 return self.\_\_z0  
  
 def \_\_init\_\_(self):  
 self.\_\_x = 0  
 self.\_\_y = 0  
 self.\_\_z = 0  
 self.\_\_x0 = round(random.uniform(-100, 100), 1)  
 self.\_\_y0 = round(random.uniform(-100, 100), 1)  
 self.\_\_z0 = round(random.uniform(-100, 100), 1)  
  
 def GetCoordinates(self, t):  
 a1 = round(random.uniform(-5, 5), 1)  
 a2 = round(random.uniform(-5, 5), 1)  
 self.\_\_x = round(self.\_\_x0 + a1 \* math.sin(t), 2)  
 self.\_\_y = round(self.\_\_y0 + a2 \* math.cos(t), 2)  
 self.\_\_z = 0  
 return self.\_\_x, self.\_\_y, self.\_\_z  
  
 def GetDistance(self, x2, y2, z2):  
 return math.sqrt(math.pow(x2 - self.\_\_x, 2) + math.pow(y2 - self.\_\_y, 2) + math.pow(z2, 2))  
  
 def InitialCoordinates(self):  
 print(f"Initial Coordinates(", self.\_\_x0, ',', self.\_\_y0, ',', self.\_\_z0, ')')  
  
 def FinalCoordinates(self):  
 print("Final Coordinates: (", self.\_\_x, ',', self.\_\_y, ',', self.\_\_z, ')')

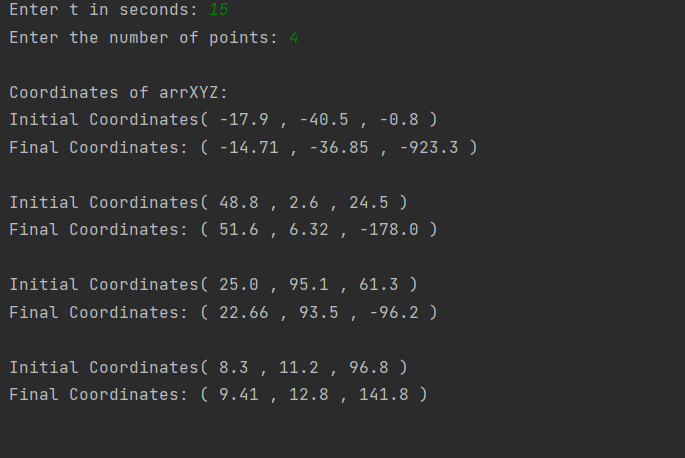
funct.py

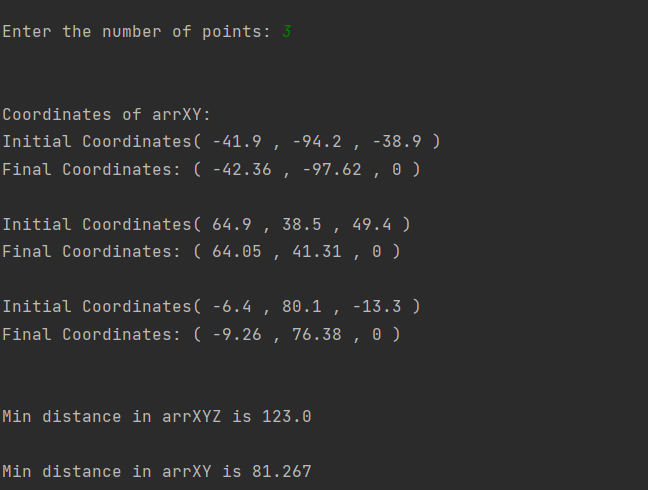
def inputT():  
 time = float(input("Enter t in seconds: "))  
 while True:  
 if time > 0:  
 return time  
 else:  
 time = float(input("Wrong input. Try again: "))  
  
def inputN():  
 count = int(input("Enter the number of points: "))  
 while True:  
 if count > 0:  
 return count  
 else:  
 count = float(input("Wrong input. Try again: "))  
  
  
def outputCoordinates(array):  
 for i in range(len(array)):  
 array[i].InitialCoordinates()  
 array[i].FinalCoordinates()  
 print()  
  
  
  
def MinD(array):  
 minD = array[0].GetDistance(array[1].GetX(), array[1].GetY(), array[1].GetZ())  
 for i in range(len(array)-1):  
 for j in range(i+1,len(array)):  
 temp = array[i].GetDistance(array[j].GetX(), array[j].GetY(), array[j].GetZ())  
 if temp < minD:  
 minD = temp  
 return minD

Скріншоти виконання програми C++:



Скріншоти виконання програми Python:





*Висновок*

В ході лабораторної роботи я вивчив механізми створення класів та об’єктів.