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 № 5 з дисципліни

«Основи програмування – 2.

Методології програмування»

«Перевантаження операторів»

Варіант 32

Виконав студент ІП-11 Фукс Вікторія Ігорівна

Перевірив \_\_\_\_\_Вітковська Ірина Павлівна\_\_\_\_\_\_\_\_\_

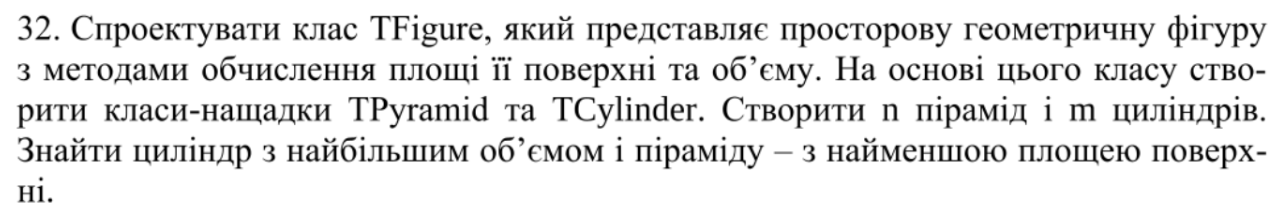
Київ 2022

**Лабораторна робота 5**

**УСПАДКУВАННЯ ТА ПОЛІМОРФІЗМ**

**Мета** – вивчити механізми створення і використання класів та об’єктів.

**Постановка задачі**:

****

**Код:**

**C#**

**Program.cs:**

using System;  
  
namespace csharp\_lab5  
{  
 class Program  
 {  
 static void Main(string[] args)  
 {  
 Console.WriteLine("Enter number of pyramids: ");  
 int n = Convert.ToInt32(Console.ReadLine());  
 Console.WriteLine("Enter number of cylinders: ");  
 int m = Convert.ToInt32(Console.ReadLine());  
  
 TPyramid[] pyr = new TPyramid[n];  
 TCylinder[] cyl = new TCylinder[m];  
  
 double[] Square = new double[n];  
 double[] Volume = new double[m];  
  
 for (int i = 0; i < Square.Length; i++)  
 {  
 pyr[i] = new TPyramid(Rnd(), Rnd());  
 Square[i] = pyr[i].S();  
 Console.WriteLine(pyr[i]);  
 }  
 Console.WriteLine("\n");  
 for (int i = 0; i < Volume.Length; i++)  
 {  
 cyl[i] = new TCylinder(Rnd(), Rnd());  
 Volume[i] = cyl[i].V();  
 Console.WriteLine(cyl[i]);  
 }  
  
 Console.WriteLine("\n");  
   
 double min = minS(Square);  
 double max = maxV(Volume);  
 Console.WriteLine($"Minimum Pyramid Square = {Math.Round(min, 3)}");  
 Console.WriteLine($"Maximum Cylinder Volume = {Math.Round(max, 3)}");  
  
 }  
   
 public static double minS(double[] arr)  
 {  
 double minS = arr[0];  
 for (int i = 0; i < arr.Length; i++)  
 {  
 if (arr[i] < minS) minS = arr[i];  
 }  
 return minS;  
 }  
 public static double maxV(double[] arr)  
 {  
 double maxV = arr[0];  
 for (int i = 0; i < arr.Length; i++)  
 {  
 if (arr[i] > maxV) maxV = arr[i];  
 }  
 return maxV;  
 }  
 public static int Rnd()  
 {  
 Random rnd = new Random();  
 return rnd.Next(1, 21);  
 }  
 }  
}

**TFigure.cs:**

namespace csharp\_lab5  
{  
 public abstract class TFigure  
 {  
 public abstract double S();  
 public abstract double V();  
 }  
}

**TCylinder.cs:**

using System;  
namespace csharp\_lab5  
{  
 internal class TCylinder : TFigure  
 {  
 private double r, h;  
 public TCylinder(double r, double h)  
 {  
 this.r = r;  
 this.h = h;  
 }  
   
 public override double V()  
 {  
 return Math.**PI** \* h \* Math.Pow(r, 2);  
 }  
 public override double S()  
 {  
 return 2 \* Math.**PI** \* r \* (h + r);  
 }  
   
 public double R { get => r; set => r = value;}  
 public double H{ get => h; set => h = value;}  
   
 public override string ToString()  
 {  
 return "Cylinder: " +  
 $"radius = {r} " +  
 $"height = {h} " +  
 $"Volume = {Math.Round(V(),3)}";  
 }  
 }  
}

**TPyramid.cs:**

using System;  
namespace csharp\_lab5  
{  
 internal class TPyramid : TFigure  
 {  
 private double a, p, h;  
 public TPyramid(double a, double p)  
 {  
 this.a = a;  
 this.p = p;  
 }  
 public override double S()  
 {  
 return (p \* a) / 2;  
 }  
   
 public override double V()  
 {  
 return (p \* a) / 2 \* (1/3) \* h;  
 }  
  
 public double A { get => a; set => a = value;}  
 public double P { get => p; set => p = value;}  
   
 public override string ToString()  
 {  
 return "Pyramid: " +  
 $"apothem = {a} " +  
 $"perimeter = {p} " +  
 $"Square = {Math.Round(S(),3)}";  
 }  
 }  
}

**Python:**

**Main.py**

from random import \*  
from TPyramid import \*  
from TCylinder import \*  
  
  
def minS(arr):  
 min\_S = arr[0]  
 for i in range(len(arr)):  
 if arr[i] < min\_S:  
 min\_S = arr[i]  
 return min\_S  
  
  
def maxV(arr):  
 max\_V = arr[0]  
 for i in range(len(arr)):  
 if arr[i] > max\_V:  
 max\_V = arr[i]  
 return max\_V  
  
  
def main():  
 print("Enter number of pyramids: ")  
 n = int(input())  
 print("Enter number of cylinders: ")  
 m = int(input())  
  
 pyramids = [TPyramid(randint(1, 20), randint(1, 20)) for i in range(n)]  
 cylinders = [TCylinder(randint(1, 20), randint(1, 20)) for i in range(m)]  
  
  
 # pyramids  
 Squares = [Pyramid.S() for Pyramid in pyramids]  
 # cylinders  
 Volumes = [Cylinder.V() for Cylinder in cylinders]  
  
 for pyramid in pyramids:  
 print(pyramid)  
  
 print("\n")  
  
 for cylinder in cylinders:  
 print(cylinder)  
  
 print("\n")  
  
 # min Square of Pyramids and min Volume of Cylinders  
 minS = min(Squares)  
 maxV = max(Volumes)  
 print(f"Minimum Pyramid Square = {minS}")  
 print(f"Maximum Cylinder Volume = {maxV}")  
  
  
main()

**TFigure.py**

from abc import ABC, abstractmethod  
  
class TFigure(ABC):  
  
 @abstractmethod  
 def S(self):  
 pass  
  
 @abstractmethod  
 def V(self):  
 pass

**TCylinder.py**

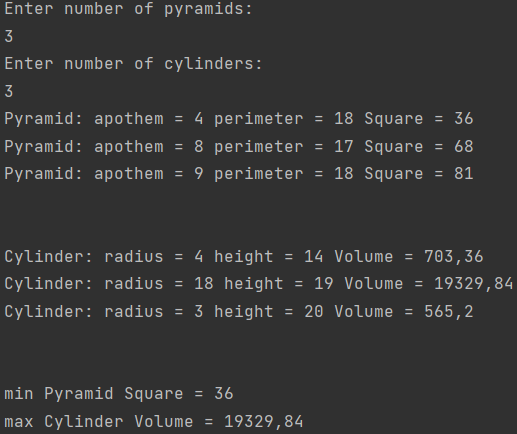
import math  
  
from TFigure import \*  
  
  
class TCylinder(TFigure):  
  
 def V(self):  
 return math.pi \* self.h \* (self.r \*\* 2)  
  
 def S(self):  
 return 2 \* math.pi \* self.r \* (self.h + self.r)  
  
 def \_\_init\_\_(self, h, r):  
 self.h = h  
 self.r = r  
  
 def \_\_str\_\_(self):  
 return "Cylinder: " + f"radius ={self.r} " + f"height = {self.h} " + f"Volume = {round(self.V(), 3)} "

**TPyramid.py**

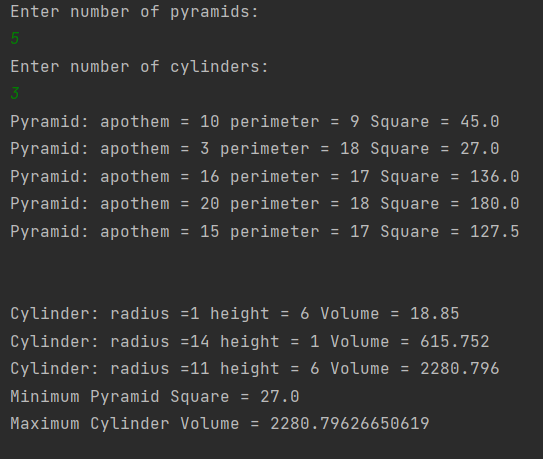
from TFigure import \*  
  
  
class TPyramid(TFigure):  
  
 def S(self):  
 return (self.p \* self.a) / 2  
  
 def V(self):  
 return (self.p \* self.a) / 2 / 2 \* (1/3) \* self.h  
  
 def \_\_init\_\_(self, a, p):  
 self.a = a  
 self.p = p  
  
 def \_\_str\_\_(self):  
 return "Pyramid: " + f"apothem = {self.a} " + f"perimeter = {self.p} " + f"Square = {self.S()} "

**Випробовування алгоритму:**

**C#:**

****

**Python:**

****

**Висновок:**

Ми вивчили механізми створення і використання класів та об’єктів.

У результаті лабораторної роботи ми написали код на двох мовах програмування С# та Python. Зробили випробування алгоритму.