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. Модульне програмування»

«Успадкування та поліморфізм»

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 №5

Тема: успадкування та поліморфізм.

Мета: вивчити механізми успадкування та поліморфізму при створенні і використанні класів та об’єктів.

Хід роботи

Задача.

32. Спроектувати клас TFigure, який представляє просторову геометричну фігуру з методами обчислення площі її поверхні та об’єму. На основі цього класу створити класи-нащадки TPyramid та TCylinder. Створити n пірамід і m циліндрів. Знайти циліндр з найбільшим об’ємом і піраміду – з найменшою площею поверхні.

Виконання задачі.

Випробування коду на C++.

Код:

main.cpp

```cpp
#include "module.h"

int main()
{
    srand(time(NULL));

    //введення кількості пірамід і циліндрів
    int n = sizeInput("pyramids");
    int m = sizeInput("cylinders");

    //генерація і виведення масиву об'єктів класу TPyramid
    TFigure** arrPyr = arrGen<TPyramid>(n);
    arrOutput(n, arrPyr);

    //генерація і виведення масиву об'єктів класу TCylinder
    TFigure** arrCyl = arrGen<TCylinder>(m);
    arrOutput(m, arrCyl);

    //знаходження піраміди з найменшою площею поверхні
    cout << endl << "Pyramids' area:\n";
    arrMaxMin(n, arrPyr, "Min area of pyramids", "area", &boolMin);

    //знаходження піраміди з найбільшою площею поверхні
    cout << endl << "Cylinders' volume:\n";
    arrMaxMin(m, arrCyl, "Max volume of cylinders", "volume", &boolMax);

    //видалення масивів з динамічної пам'яті
    arrDelete(n, arrPyr);
    arrDelete(m, arrCyl);

    cout << endl;
    system("pause");
}
```

module.h

```h
#pragma once
#include <iostream>
#include <iomanip>
#include "TCylinder.h"
#include "TPyramid.h"

int sizeInput(string str);  //введення розміру масиву

void arrOutput(int size, TFigure** arr); //виведення масиву об'єктів

//знаходження максимального або мінімального значення у масиві
void arrMaxMin(int size, TFigure** arr, string str, string method, bool(*func)(float, float));

bool boolMax(float a, float b); //чи є перший параметр більший за другий

bool boolMin(float a, float b); //чи є перший параметр менший за другий

void arrDelete(int size, TFigure** arr); //видадення масиву з динамічної пам'яті

//шаблонна функція для генерації масиву об'єктів
template <class T>
TFigure** arrGen(int size)
{
    TFigure** arr = new TFigure * [size];
    for (int i = 0; i < size; i++)
        arr[i] = new T();
    return arr;
}
```

module.cpp

```cpp
#include "module.h"

//введення розміру масиву
int sizeInput(string str)
{
    int n;
    cout << "Enter the size of array for " << str << ": ";
    cin >> n;
    return n;
}

//виведення масиву об'єктів
void arrOutput(int size, TFigure** arr)
{
    for (int i = 0; i < size; i++)
        arr[i]->output();
    cout << endl;
}
```

```cpp
//знаходження максимального або мінімального значення площі поверхні або об'єму у масиві об'єктів
void arrMaxMin(int size, TFigure** arr, string str, string method, bool(*func)(float, float))
{
    float value = (*arr)->funcChoice(method); //значення, яке шукається
    int pos = 1;    //позиція значення, яке шукається
    float temp;     //значення площі або об'єму поточного об'єкта
    for (int i = 0; i < size; i++)
    {
        temp = arr[i]->funcChoice(method);
        cout << "#" << setw(3) << left << i+1 << internal << ' ' << fixed << setw(10) << setprecision(2) << temp << endl;
        if (func(temp, value))  //func може приймати значення boolMax та boolMin
        {
            value = temp;
            pos = i;
        }
    }
    cout << str << ": " << setprecision(2) << value << " at position " << pos + 1 << endl;
}
```

```cpp
//чи є перший параметр більший за другий
bool boolMax(float a, float b)
{
    return a > b ? true : false;
}

//чи є перший параметр менший за другий
bool boolMin(float a, float b)
{
    return a < b ? true : false;
}

//видадення масиву з динамічної пам'яті
void arrDelete(int size, TFigure** arr)
{
    for (int i = 0; i < size; i++)
    {
        delete arr[i];
    }
    delete[size] arr;
}
```

TFigure.h

```h
#pragma once
#define _USE_MATH_DEFINES
#include <iostream>
#include <math.h>
#include <iomanip>
using namespace std;

class TFigure
{
public:
    virtual float area() = 0;        //площа поверхні фігури
    virtual float volume() = 0;      //об'єм фігури
    virtual void randomValue() = 0; //випадкове значення фігури
    virtual void output() = 0;       //виведення інформації про фігуру
    float funcChoice(string str)     //вибір між знаходженням площі і об'єму
    { return str == "volume" ? volume() :
        (str == "area" ? area() : 0); }
};
```

TPyramid.h

```h
#pragma once
#include "TFigure.h"

enum class figure;  //перелічувальний тип для позначення основи піраміди
string figureCheck(figure f);   //перевірка форми основи піраміди

//правильна піраміда (в основі лежить правильний багатокутник)
class TPyramid : public TFigure
{
    figure basis;           //форма основи піраміди
    float side, height;     //довжина сторони основи та висота
public:
    //конструктор за замовчуванням, який заповнює поля випадковими значеннями
    TPyramid() { randomValue(); }

    //конструктор зі списком ініціалізації
    TPyramid(figure f, float s, float h) : basis(f), side(s), height(h) {};

    //геттери
    figure getBasis() { return basis; }
    float getSide() { return side; }
    float getHeight() { return height; }

    float area();       //площа поверхні піраміди
    float volume();     //об'єм піраміди
    void randomValue(); //випадкове значення піраміди
    void output();      //виведення інформації про піраміду
};
```

TPyramid.cpp

```cpp
#include "TPyramid.h"

//перелічувальний тип для позначення основи піраміди
enum class figure
{
    triangle = 3,   //основа - рівносторонній трикутник
    square = 4      //основа - квадрат
};

//перевірка форми основи піраміди
string figureCheck(figure f)
{
    return f == figure::triangle ? "triangle" : "square";
}

//площа поверхні піраміди
float TPyramid::area()
{
    switch (basis)
    {
    case figure::triangle:
        return pow(side, 2) * sqrt(3) / 4 + 1.5 * side * sqrt(pow((side / (2 * sqrt(3))), 2) + pow(height, 2));
    case figure::square:
        return side * side + side * sqrt(pow((side / 2), 2) * 2 + pow(height, 2));
    }
}
```

```cpp
//об'єм піраміди
float TPyramid::volume()
{
    switch (basis)
    {
    case figure::triangle:
        return pow(side, 2) * sqrt(3) / 4 * height / 3;
    case figure::square:
        return side * side * height / 3;
    }
}

//випадкове значення піраміди
void TPyramid::randomValue()
{
    basis = figure(rand() % 2 + 3);
    side = rand() % 10 + 1;
    height = rand() % 10 + 1;
}

//виведення інформації про піраміду
void TPyramid::output()
{
    cout << "\nPyramid: " << setw(2) << height << " h, " << setw(2) << side << " s, " << figureCheck(basis);
}
```

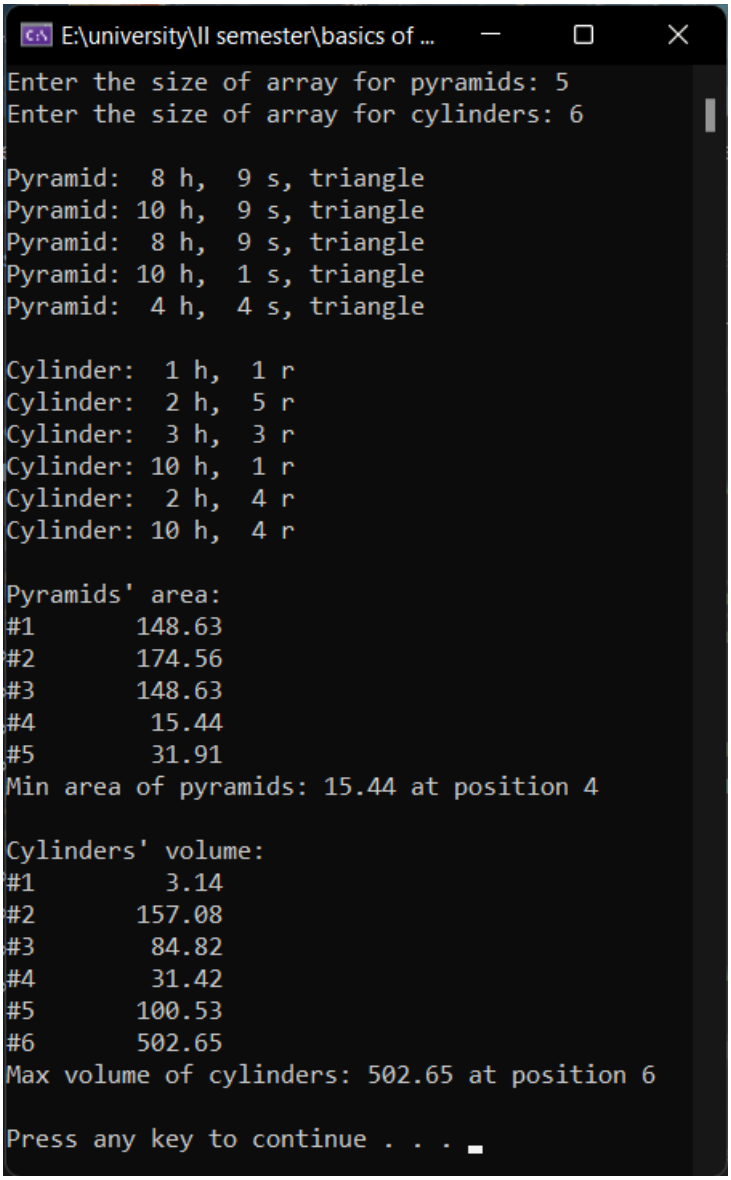
TCylinder.h

```h
#pragma once
#include "TFigure.h"

class TCylinder : public TFigure
{
    float radius, height; //радіус та висота
public:
    //конструктор за замовчуванням, який заповнює поля випадковими значеннями
    TCylinder() { randomValue(); }

    //конструктор зі списком ініціалізації
    TCylinder(float r, float h) : radius(r), height(h) {};

    //геттери
    float getRadius() { return radius; }
    float getHeight() { return height; }

    float area();       //площа поверхні циліндра
    float volume();     //об'єм циліндра
    void randomValue(); //випадкове значення циліндра
    void output();      //виведення інформації про циліндр
};
```

TCylinder.cpp

```cpp
#include "TCylinder.h"

//площа поверхні циліндра
float TCylinder::area()
{
    return 2 * (float)M_PI* radius* height;
}

//об'єм циліндра
float TCylinder::volume()
{
    return (float)M_PI * (float)pow(radius, 2) * height;
}

//випадкове значення циліндра
void TCylinder::randomValue()
{
    height = float(rand() % 10) + 1;
    radius = float(rand() % 5) + 1;
}

//виведення інформації про циліндр
void TCylinder::output()
{
    cout << "\nCylinder: " << setw(2) << height << " h, " << setw(2) << radius << " r";
}
```

Результат:



Випробування коду на Python.

Код:

main.py

```python
from TPyramid import TPyramid
from TCylinder import TCylinder
import module

#генерація і виведення масиву об'єктів класу TPyramid
arrPyr = module.arrGen("pyramids", TPyramid)

#генерація і виведення масиву об'єктів класу TCylinder
arrCyl = module.arrGen("cylinders", TCylinder)

#знаходження піраміди з найменшою площею поверхні
print("Pyramids' area:")
PyrMinArea, PyrMinAreaPos = module.arrMaxMin(arrPyr, min, "area")
print("Min pyramid area is {:.2f} at position #{}\n".format(PyrMinArea, PyrMinAreaPos))

#знаходження піраміди з найбільшою площею поверхні
print("Cylinders' volume:")
CylMaxVolume, CylMaxVolumePos = module.arrMaxMin(arrCyl, max, "volume")
print("Max cylinder volume is {:.2f} at position #{}\n".format(CylMaxVolume, CylMaxVolumePos))
```

module.py

```python
#генерація масиву об'єктів заданого класу
def arrGen(word, Class):
    arr = [Class() for i in range(int(input("Enter the number of "+word+": ")))]
    [arr[i].output() for i in range(len(arr))]
    print()
    return arr

#знаходження максимального або мінімального значення у масиві
def arrMaxMin(arr, func, method):
    arrValue = [arr[i].funcChoice(method) for i in range(len(arr))]
    [print("#{:<3} {:>6.2f}".format(i+1,arrValue[i])) for i in range(len(arrValue))]
    Value = func(arrValue)
    pos = arrValue.index(Value)
    return Value, pos+1
```

TFigure.py

```python
class TFigure(object):
    def area():          #площа поверхні фігури
        pass
    def volume():        #об'єм фігури
        pass
    def randomValue():   #випадкове значення фігури
        pass
    def output():        #виведення інформації про фігуру
        pass

    #вибір між знаходженням площі і об'єму
    def funcChoice(self, word):
        if word == "volume":
            return self.volume()
        elif word == "area":
            return self.area()
```

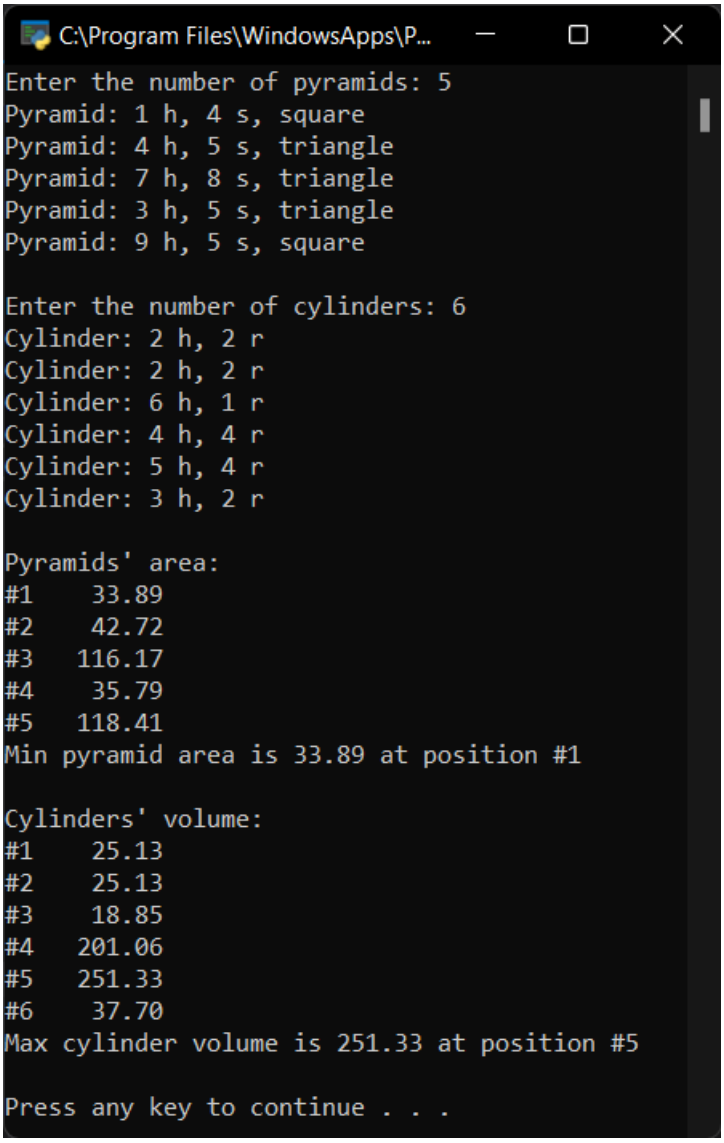
TPyramid.py

```python
from TFigure import TFigure
from math import pow
from math import sqrt
import random
#правильна піраміда (в основі лежить правильний багатокутник)
class TPyramid(TFigure):
    def __init__(self, Basis, Side, Height):
        self.basis = Basis      #форма основи піраміди
        self.side = Side        #довжина сторони основи піраміди
        self.height = Height    #висота піраміди

    #конструктор за замовчуванням, випадкові значення
    def __init__(self):
        self.randomValue()

    #площа поверхні піраміди
    def area(self):
        if self.basis == "triangle":
            return pow(self.side, 2) * sqrt(3) / 4 + 1.5 * self.side * sqrt(pow((self.side / (2 * sqrt(3))), 2) + pow(self.height, 2))
        elif self.basis == "square":
            return self.side * self.side + 2 * self.side * sqrt(pow((self.side / 2), 2) + pow(self.height, 2))

    #об'єм піраміди
    def volume(self):
        if self.basis == "triangle":
            return pow(self.side, 2) * sqrt(3) / 4 * self.height / 3
        elif self.basis == "square":
            return self.side * self.side * self.height / 3

    #випадкове значення піраміди
    def randomValue(self):
        self.basis = random.choice(["triangle", "square"])
        self.side = random.randrange(1, 10)
        self.height = random.randrange(1, 10)

    #виведення інформації про піраміду
    def output(self):
        print("Pyramid: {} h, {} s, {}".format(self.height, self.side, self.basis))
```

TCylinder.py

```python
from TFigure import TFigure
from math import pow
from math import pi
import random
class TCylinder(TFigure):
    def __init__(self, Radius, Height):
        self.radius = Radius    #радіус основи циліндра
        self.height = Height    #висота циліндра

    #конструктор за замовчуванням, випадкові значення
    def __init__(self):
        self.randomValue()

    #площа поверхні циліндра
    def area(self):
        return 2 * pi * self.radius * self.height

    #об'єм циліндра
    def volume(self):
        return pi * pow(self.radius, 2) * self.height

    #випадкове значення циліндра
    def randomValue(self):
        self.radius = random.randrange(1, 5)
        self.height = random.randrange(1, 10)

    #виведення інформації про циліндр
    def output(self):
        print("Cylinder: {} h, {} r".format(self.height, self.radius))
```

Результат:



Висновок: При виконанні лабораторної роботи було вивчено механізми наслідування та поліморфізму та досліджено засоби їх застосування у мовах програмування C++ та Python. На основі вивчених засобів було виконано задачу, в якій класи циліндр та піраміда стали нащадками класу фігура.