Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут”

Кафедра АСОІУ

**ЗВІТ**

про виконання комп’ютерного практикуму № 2

з дисципліни

“ООП”

Тема: «Наслідування С++»

|  |  |  |
| --- | --- | --- |
| Прийняв: |  | Виконав: |
| Головченко Максим Миколайович |  | студент 2-го курсу  гр. ІП-51 ФІОТ  Зарічковий Олександр Анатолійович |

Київ – 2016

**ЗМІСТ:**

[1 Мета роботи 4](#_Toc463208490)

[2 Постановка задачі (варіант 7, рівень В) 5](#_Toc463208491)

[3 Покроковий алгоритм 6](#_Toc463208492)

[4 Діаграма класів 7](#_Toc463208493)

[5 Код програми 8](#_Toc463208494)

[6 Приклади виконання програми 12](#_Toc463208495)

[7 Висновок 13](#_Toc463208496)

# Мета роботи

Мета роботи - вивчити основні концепції об'єктно-орієнтованого програмування. Вивчити особливості успадкування і множинного успадкування.

# Постановка задачі (варіант 7, рівень В)

Визначити наступну ієрархію класів (рисунок 2.1).

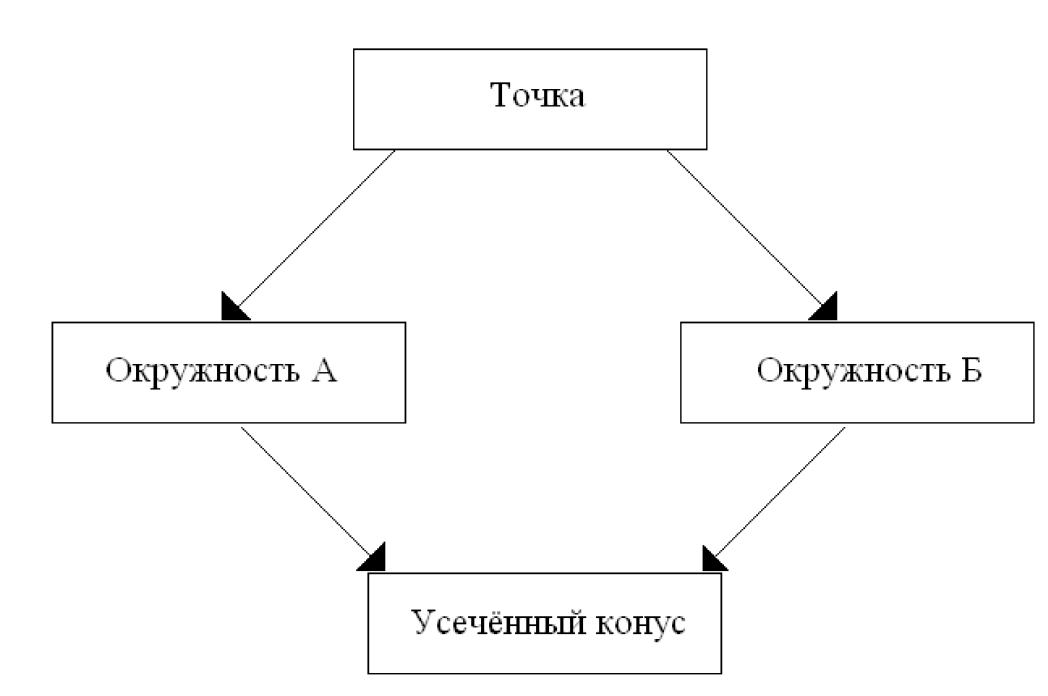


Рисунок 2.1 – Ієрархія класів

Визначити площу кожної фігури (для усіченого конуса – площа бічної поверхні і об’єм). Ініціалізація даних проводиться користувачем.

# Покроковий алгоритм

* Обчислення бічної висота конуса

1. ПОЧАТОК
2. Визначити приріст різницю X координат верхнього та нижнього кіл зрізаного конуса (**deltaX**).
3. Визначити приріст різницю Y координат верхнього та нижнього кіл зрізаного конуса (**deltaY**).
4. Визначити приріст різницю Z координат верхнього та нижнього кіл зрізаного конуса (**deltaZ**).
5. Визначити висоту за формулою:
6. КІНЕЦЬ

* Обчислення бічної поверхні конуса

1. ПОЧАТОК
2. Визначити довжину твірної за формулою:

,

де R та r – радіуси більшого і меншого кіл відповідно.

1. Обчислити бічну площу за формулою:
2. КІНЕЦЬ

# Діаграма класів

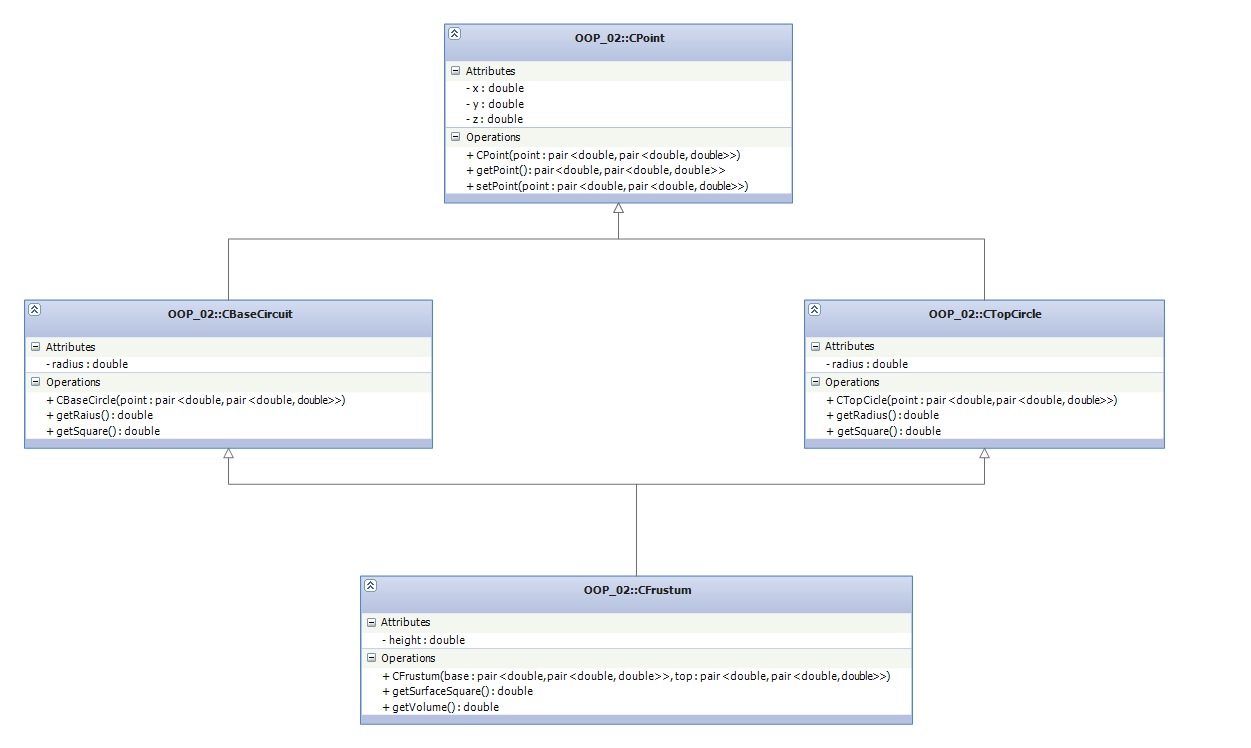
Діаграма класів наведена на рисунку 4.1:

Рисунок 4.1 – Діаграма класів

# Код програми

“main.cpp”

#include "Interface.h"

int main(void) {

while (outputMenu());

}

“Interface.h”

#pragma once

#include <utility>

using std::pair;

bool outputMenu(void); // Меню

pair <double, pair <double, double>> inputPoint(char \* str); // Введення точки

“Interface.cpp”

#include "Interface.h"

#include "СFrustum.h"

#include <cstdio>

#include <cstdlib>

// Виведення меню

bool outputMenu(void) {

system("cls");

printf("OOP lab work #2\nVariant - 7, Level - C\nTask: Define class frustum using multiple imitation\nWriter: Alexander ALEXKIRNAS Zarichkovyi\n\n");

printf("Menu:\n1. Start program\n2. Exit\n\nMake your choice ");

int t;

scanf("%d", &t);

switch (t) {

case 1: {

pair <double, pair <double, double>> base = inputPoint("base");

pair <double, pair <double, double>> top = inputPoint("top");

CFrustum obj(base, top);

printf("\nSquare of base circle: %Lf\n", obj.CBaseCircle::getSquare());

printf("Square of top circle: %Lf\n\n", obj.CTopCircle::getSquare());

printf("Volume of frustum: %Lf\n", obj.getVolume());

printf("Surface square of frustum: %Lf\n\n", obj.getSufaceSquare());

system("pause");

break;

}

case 2: return false;

}

return true;

}

// Введення точки

pair <double, pair <double, double>> inputPoint(char \* str) {

double x, y, z;

std::cout << "Enter X coordinate of " << str << " circle: ";

std::cin >> x;

std::cout << "Enter Y coordinate of " << str << " circle: ";

std::cin >> y;

std::cout << "Enter Z coordinate of " << str << " circle: ";

std::cin >> z;

std::cout << "\n";

return std::make\_pair(x, std::make\_pair(y, z));

}

“CPoint.h”

#pragma once

#include <utility>

#define M\_PI 3.14159265358979323846 // pi

using std::pair;

// Клас точки

class CPoint {

private:

double x, y, z; // Координати точки

public:

CPoint(pair <double, pair <double, double>> point); // Конструктор

pair <double, pair <double, double>> getPoint(void); // Гетер точки

void setPoint(pair <double, pair <double, double>> point); // Сетер точки

};

“CPoint.cpp”

#include "CPoint.h"

using std::make\_pair;

// Конструктор

CPoint::CPoint(pair <double, pair <double, double>> point) {

setPoint(point);

}

// Гетер точки

pair <double, pair <double, double>> CPoint::getPoint(void) {

return make\_pair(x, make\_pair(y, z));

}

// Сетер точки

void CPoint::setPoint(pair <double, pair <double, double>> point) {

x = point.first;

y = point.second.first;

z = point.second.second;

}

“CBaseCircle.h”

#pragma once

#include "CPoint.h"

#include <iostream>

class CBaseCircle : public CPoint {

private:

double radius; // Радіус основи

public:

CBaseCircle(pair <double, pair <double, double>> point); // Конструктор

double getRadius(void); // Гетер радіуса

double getSquare(void); // Розрахувати площу

};

“CBaseCircle.cpp”

#include "CBaseCircle.h"

using std::cin;

using std::cout;

// Конструктор

CBaseCircle::CBaseCircle(pair <double, pair <double, double>> point) : CPoint(point) {

cout << "Enter radius of base circle: ";

cin >> radius;

}

// Гетер радіуса

double CBaseCircle::getRadius(void) {

return radius;

}

// Розрахувати площу

double CBaseCircle::getSquare(void) {

return radius \* radius \* M\_PI;

}

“CTopCicle.h”

#pragma once

#pragma once

#include "CPoint.h"

#include <iostream>

class CTopCircle : public CPoint {

private:

double radius; // Радіус верхнього кола

public:

CTopCircle(pair <double, pair <double, double>> point); // Конструктор

double getRadius(void); // Гетер радіуса

double getSquare(void); // Розрахувати площу

};

“CTopCicle.cpp”

#pragma once

#pragma once

#include "CPoint.h"

#include <iostream>

class CTopCircle : public CPoint {

private:

double radius; // Радіус верхнього кола

public:

CTopCircle(pair <double, pair <double, double>> point); // Конструктор

double getRadius(void); // Гетер радіуса

double getSquare(void); // Розрахувати площу

};

#include "CTopCircle.h"

using std::cin;

using std::cout;

// Конструктор

CTopCircle::CTopCircle(pair <double, pair <double, double>> point) : CPoint(point) {

cout << "Enter radius of top circle: ";

cin >> radius;

}

// Гетер радіуса

double CTopCircle::getRadius(void) {

return radius;

}

// Розрахувати площу

double CTopCircle::getSquare(void) {

return radius \* radius \* M\_PI;

}

“CFrustum.h”

#pragma once

#include "CTopCircle.h"

#include "CBaseCircle.h"

class CFrustum : public CTopCircle, public CBaseCircle {

private:

double height; // Висота конуса

public:

CFrustum(pair <double, pair <double, double>> base, pair <double, pair <double, double>> top); // Конструктор

double getVolume(void); // Розрахувати обєм

double getSufaceSquare(void); // Розрахувати площу поверхні

};

“CFrustum.cpp”

#include "СFrustum.h"

#include <cmath>

// Конструктор

CFrustum::CFrustum(pair <double, pair <double, double>> base, pair <double, pair <double, double>> top) : CBaseCircle(base), CTopCircle(top) {

double deltaX, deltaY, deltaZ;

deltaX = base.first - top.first;

deltaY = base.second.first - top.second.first;

deltaZ = base.second.second - top.second.second;

height = sqrt(pow(deltaX, 2) + pow(deltaY, 2) + pow(deltaZ, 2));

}

// Розрахувати обєм

double CFrustum::getVolume(void) {

return static\_cast<double> (1) / 3 \* M\_PI \* height \*

(pow(CBaseCircle::getRadius(), 2) + CBaseCircle::getRadius() \* CTopCircle::getRadius() + pow(CTopCircle::getRadius(), 2));

}

// Розрахувати площу поверхні

double CFrustum::getSufaceSquare(void) {

double generatrix = sqrt(pow(height, 2) + pow((CBaseCircle::getRadius() - CTopCircle::getRadius()), 2));

return M\_PI \* generatrix \* (CBaseCircle::getRadius() + CTopCircle::getRadius());

}

# Приклади виконання програми

Приклад виконання програми наведений на рисунку 5.1:

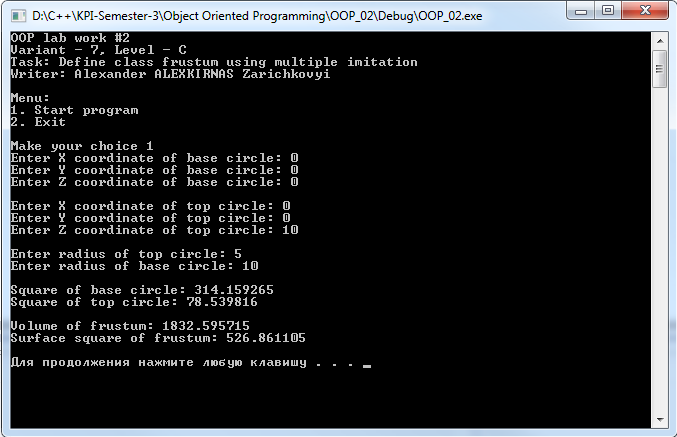


Рисунок 5.1 – Приклад виконання програми

# Висновок

Отже, дана програма створює об’єкт класа CFrustum, який створює зрізаний конус за допомогою двох класів-предків CTopCicle та СBaseCircle, які створюють кола в просторо та в свою чергу наслідуються від класа CPoint, який є класом точки в просторі. Оскільки дана програма реалізована за модульним принципом і кожен модуль програми створювався й налагоджувався автономно, то дана програма буде видавати очікуваний результат на всіх наборах вхідних даних.