2016 Fall CIS200 – Lab 7

**Answers**

#include <iostream>

using namespace std;

/\*\*

Make a base class for a Polygon

The private members

- numOfSides : int

- sides[] : int // listed in clockwise order , MAX sides is 100 The attributes

+ Polygon(numSides:int) // default is 0

+ set(sideNum: int, value:int) : void// ensures no negative side values

+ get (sideNum:int):int // returns the value of sides[sideNum]

+ perimeter(): int virtual// returns the perimeter of the polygon

+ area(): double virtual //returns the area of the polygon

+ volume(): double virtual //returns 0

\*/

class Polygon {

public:

Polygon(int n = 0) {

numOfSides = n;

sides = new int[n];

}

virtual void set(int sideNum, int value) {

if (sideNum >= 0 && sideNum < numOfSides && value >= 0)

sides[sideNum] = value;

}

int get(int sideNum) {

if (sideNum >= 0 && sideNum < numOfSides)

return sides[sideNum];

}

virtual int perimeter() {

int perim = 0;

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

perim += sides[i];

return perim;

}

virtual double area() {

return 0;

}

virtual double volume() {

return 0;

}

private:

int numOfSides;

int \*sides;

};

/\*

Make a class for Rectangle derived from Polygon

The attributes + Rectangle()// class Polygon(4);

+ area() // appropriately

+ perimeter()() // appropriately

\*/

class Rectangle : public Polygon {

public:

Rectangle() :Polygon(4) {

}

void set(int index, int value) {

if (index == 0 || index == 2) {

Polygon::set(0, value);

Polygon::set(2, value);

}

else if (index == 1 || index == 3) {

Polygon::set(1, value);

Polygon::set(3, value);

}

}

double area() {

return get(0) \* get(1);

}

private:

};

/\*

Make a class for Square derived from Rectangle

The attributes + Square()// appropriately

+ area() // appropriately

+ perimeter() // appropriately

\*/

class Square : public Rectangle {

public:

Square() {}

void set(int index, int value) {

Rectangle::set(0, value);

Rectangle::set(1, value);

}

private:

};

/\*

Make a class for RightTri derived from Polygon

The attributes + RightTri()// class Polygon(3);

//. Assumes the order side, side, hypotenuse

+ area() // appropriately

+ perimeter()

\*/

class RightTri :public Polygon {

public:

RightTri() : Polygon(3) {

}

double area() {

return get(0)\*get(1) / 2.0;

}

};

/\*

Make a class for RectSolid

+volume() :double // returns the volume

+ getHeight(): int // returns the height

+ setHeight(h:int): void // sets the height to a non-negative value

+ RectSolid(h:int)// initializes a rectangular solid

\*/

class RectSolid : public Rectangle {

public:

RectSolid(int ht = 0) {

setHeight(ht);

}

void setHeight(int value) {

if (value >= 0)

height = value;

}

int getHeight() {

return height;

}

double volume() {

return height \*area();

}

private:

int height;

};

/\*

In your main

Make a an array of pointers to Polygon and add an instance

Of a rectangle with sides of length 4 and 10

A square with sides of length 5

A right triangle with sides 3, 4 and 5

A rectangular solid with sides of length 6 and 3 and height 5.

\*/

int main() {

Polygon \* shapes[5];

shapes[0] = new Rectangle;

shapes[0]->set(0, 4);

shapes[0]->set(1, 10);

shapes[1] = new Square;

shapes[1]->set(0, 5);

shapes[2] = new RightTri;

shapes[2]->set(0, 3);

shapes[2]->set(1, 4);

shapes[2]->set(2, 5);

shapes[3] = new RectSolid(5);

shapes[3]->set(0, 6);

shapes[3]->set(1, 3);

for (int i = 0; i < 3; i++) {

cout << "the perimeter of shape " << i << " is " << shapes[i]->perimeter() << endl;

}

for (int i = 0; i < 3; i++) {

cout << "the area of shape " << i << " is " << shapes[i]->area() << endl;

}

cout << "the volumne of the rectangular solid is " << shapes[3]->volume() << endl;

return 0;

}

}

the perimeter of shape 0 is 28

the perimeter of shape 1 is 20

the perimeter of shape 2 is 12

the area of shape 0 is 40

the area of shape 1 is 25

the area of shape 2 is 6

the volumne of the rectangular solid is 90

Press any key to continue . . .