# TASK 4:

**POLYGON HEADER:**

#pragma once

#include<iostream>

using namespace std;

class polygon

{

public:

float length, width;

virtual float area() = 0;

virtual float perimeter() = 0;

virtual void display() = 0;

void lenghtsetter(float length);

void widthsetter(float width);

float lengthgetter();

float widthgetter();

}; **POLYGON CPP:**

#include"polygon.h"

void polygon::lenghtsetter(float length)

{

this->length = length;

}

void polygon::widthsetter(float width)

{

this->width = width;

}

float polygon::lengthgetter()

{

return length;

}

float polygon::widthgetter()

{

return width;

} **SQUARE HEADER:**

#pragma once

#include<iostream>

#include"polygon.h"

using namespace std;

class square : public polygon

{

public:

square();

float area();

float perimeter();

void display();

}; **SQUARE CPP:**

#include"square.h"

square::square() {}

float square::area()

{

return length \* width;

}

float square::perimeter()

{

return 4 \* length;

}

void square::display()

{

cout << "Area of sqaure is: " << area() << endl;

cout << "Perimeter of square is: " << perimeter() << endl;

}

**TRIANGLE HEADER:**

#pragma once

#include<iostream>

#include"polygon.h"

using namespace std;

class triangle :public polygon

{

public:

triangle();

triangle(float length, float width);

float area();

float perimeter();

void display();

}; **TRIANGLE CPP:**

#include"triangle.h"

triangle::triangle() {}

triangle::triangle(float length, float width)

{

this->length = length;

this->width = width;

}

float triangle::area()

{

return ((length \* width) / 2);

}

float triangle::perimeter()

{

return (length + width);

}

void triangle::display()

{

cout << "Area of triangle is: " << area() << endl;

cout << "Perimeter of triangle is: " << perimeter() << endl;

} **RECTANGLE HEADER:**

#pragma once

#include<iostream>

#include"polygon.h"

using namespace std;

class rectangle :public polygon

{

public:

rectangle();

rectangle(float lenth);

float area();

float perimeter();

void display();

}; **RECTANGLE CPP:**

#include"rectangle.h"

rectangle::rectangle() {}

rectangle::rectangle(float lenth)

{

this->length = length;

}

float rectangle::area()

{

return (2 \* (length \* width));

}

float rectangle::perimeter()

{

return (2 \* (length \* width));

}

void rectangle::display()

{

cout << "Area of rectangle is: " << area() << endl;

cout << "Perimeter of rectangle is: " << perimeter() << endl;

} **SOURCE:**

#include<iostream>

#include"rectangle.h"

#include"square.h"

#include"triangle.h"

using namespace std;

int main()

{

polygon\* p[3];

p[0] = new square;

p[0]->lenghtsetter(3.45);

p[0]->widthsetter(4.56);

p[1] = new rectangle;

p[1]->lenghtsetter(4.785);

p[1]->widthsetter(6.983);

p[2] = new triangle;

p[2]->lenghtsetter(3.849);

p[2]->widthsetter(2.8438);

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

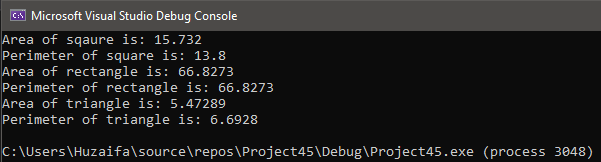
{

p[i]->display();

}

}

# OUTPUT:



# TASK 5:

**BILL HEADER:**

#include<iostream>

using namespace std;

class bill

{

public:

int units;

int per\_unit\_cost;

float calBill;

public:

bill();

virtual void Monthly\_bill(int unit, int perunit);

virtual void Bill\_display();

~bill();

}; **BILL CPP:**

#include"bill.h"

bill:: bill()

{

this->units = 0;

this->per\_unit\_cost = 0;

this->calBill = 0.0;

}

void bill::Monthly\_bill(int unit, int perunit)

{

this->units = unit;

this->per\_unit\_cost = perunit;

this->calBill = units \* per\_unit\_cost;

}

void bill::Bill\_display()

{

cout << "Total Bill is: " << calBill << endl;

}

bill::~bill()

{

cout << "Destructor" << endl;

} **JAN HEADER:**

#pragma once

#include<iostream>

#include"bill.h"

using namespace std;

class jan : public bill

{

void Monthly\_bill(int unit, int perunit);

void Bill\_display();

};

**JAN CPP:**

#include"january.h"

void jan::Monthly\_bill(int unit, int perunit)

{

this->units = unit;

this->per\_unit\_cost = perunit;

this->calBill = units \* per\_unit\_cost;

}

void jan::Bill\_display()

{

cout << "January bill is: " << calBill << endl;

} **FEB HEADER:**

#pragma once

#include<iostream>

#include"bill.h"

using namespace std;

class feb : public bill

{

void Monthly\_bill(int unit, int perunit);

void Bill\_display();

};

**FEB CPP:**

#include"february.h"

void feb::Monthly\_bill(int unit, int perunit)

{

this->units = unit;

this->per\_unit\_cost = perunit;

this->calBill = units \* per\_unit\_cost;

}

void feb::Bill\_display()

{

cout << "Febuary bill is: " << calBill << endl;

}

**MARCH HEADER:**

#pragma once

#include<iostream>

#include"bill.h"

using namespace std;

class march : public bill

{

void Monthly\_bill(int unit, int perunit);

void Bill\_display();

}; **MARCH CPP:**

#include"march.h"

void march::Monthly\_bill(int unit, int perunit)

{

this->units = unit;

this->per\_unit\_cost = perunit;

this->calBill = units \* per\_unit\_cost;

}

void march::Bill\_display()

{

cout << "march bill is: " << calBill << endl;

} **SOURCE:**

#include<iostream>

#include"january.h"

#include"february.h"

#include"march.h"

using namespace std;

int main()

{

bill\* obj1;

jan obj2;

obj1 = &obj2;

obj1->Monthly\_bill(100, 10);

obj1->Bill\_display();

feb obj3;

obj1 = &obj3;

obj1->Monthly\_bill(120, 10);

obj1->Bill\_display();

march obj4;

obj1 = &obj4;

obj1->Monthly\_bill(150, 10);

obj1->Bill\_display();

}