Task 1:

// Class.h file (class header file)

#pragma once

//#ifndef interface1.h

#include<string>

#include<iostream>

using namespace std;

//#define CLASS\_H

class ship

{

private:

string name;

string year;

public:

ship(string str="a"):name(str),year(str) {}

void set(string name, string year);

string getname();

string getyear();

virtual void print();

~ship(){}

};

class cruiseship:public ship

{

private:

int max;

public:

cruiseship(int x=0):max(x){}

void setmax(int max);

int getmax();

void print();

~cruiseship(){}

};

class cargoship:public ship

{

private:

int capacity;

public:

cargoship(int x=0) :capacity(x) {}

void setcapacity(int capacity);

int getcapacity();

void print();

~cargoship() {}

};

//implementation

#include "interface1.h"

void ship::set(string name, string year)

{

this->name = name;

this->year = year;

}

string ship::getname()

{

return name;

}

string ship::getyear()

{

return year;

}

void ship::print()

{

cout << " the name is : " << name << endl;

cout << " the year is : " << year << endl;

}

void cruiseship::setmax(int max)

{

this->max = max;

}

int cruiseship::getmax()

{

return max;

}

void cruiseship::print()

{

cout << " the ship's name is : " << this->getname() << endl;

cout << " the max passengers are : " << max<< endl;

}

void cargoship::setcapacity(int capacity)

{

this->capacity = capacity;

}

int cargoship::getcapacity()

{

return capacity;

}

void cargoship::print()

{

cout << " the Cship's name is : " << this->getname() << endl;

cout << " the capacity is : " << capacity << endl;

}

//MAIN file (source.cpp)

#include<iostream>

#include<string>

#include"interface1.h"

using namespace std;

int main()

{

ship\* ship\_ptr = new ship[2],\*shipobj;

cruiseship crship\_ptr[2], crshipobj;

cargoship carship\_ptr[2],carshipobj;

shipobj = &crshipobj;

ship\_ptr->set("american", "2002");

crship\_ptr->set("pakistani", "2004");

crship\_ptr->setmax(12);

carship\_ptr->set("indian", "2020");

carship\_ptr->setcapacity(100);

ship\_ptr->print();

crship\_ptr->print();

carship\_ptr->print();

shipobj->set("british", "2003");

crshipobj.setmax(15);

shipobj->print();

shipobj = &carshipobj;

shipobj->set("russian", "2024");

carshipobj.setcapacity(99);

shipobj->print();

ship\* p[2];

p[0] = new cruiseship(12);

p[1] = new cargoship(6);

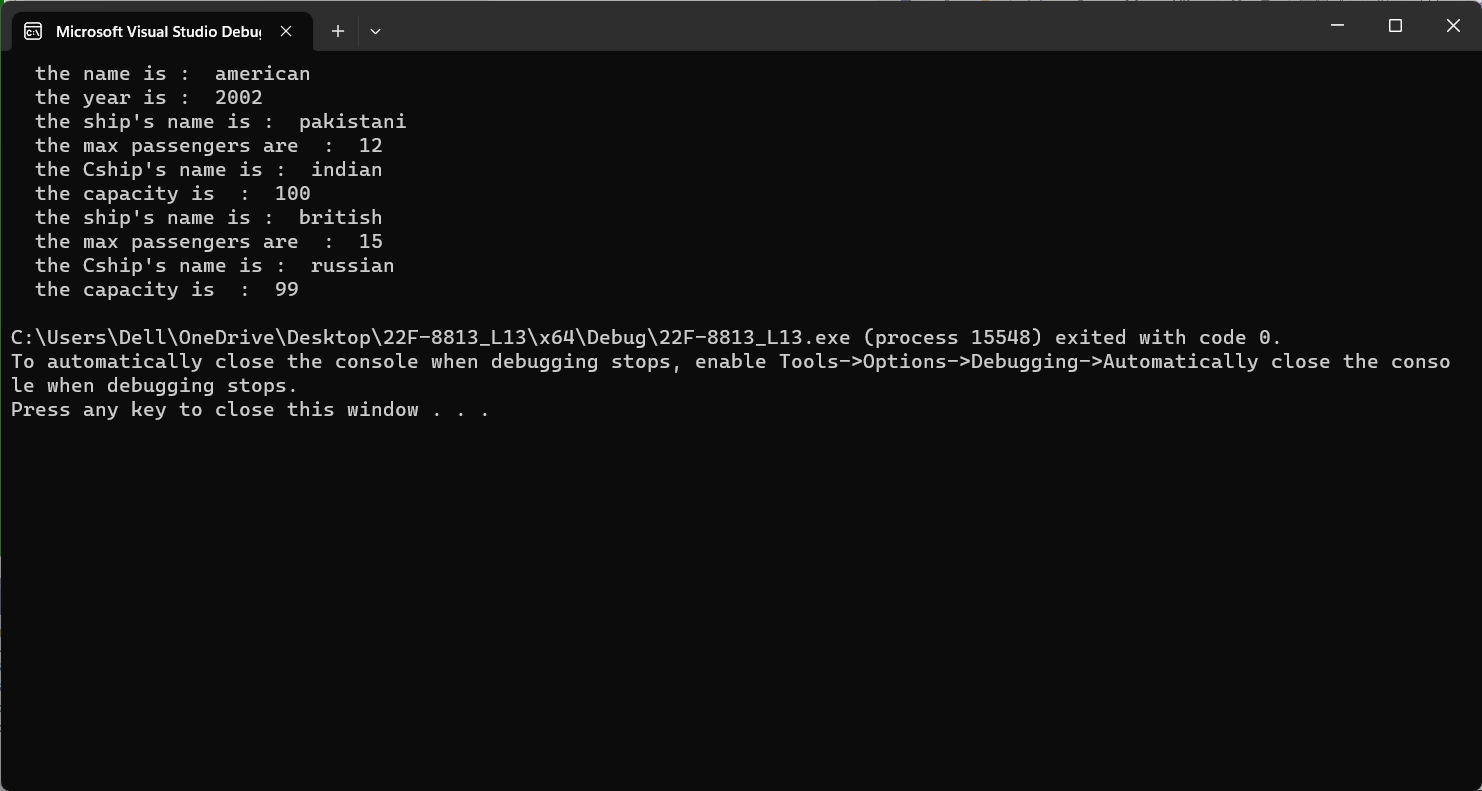
for (int i = 0; i < 2; i++)

{

p[i]->print();

}

delete[] p;}



Task 2:

//MAIN file (source.cpp)

#include<iostream>

#include<string>

#include"interface2.h"

using namespace std;

int main()

{

polygon \*Bptr;

/\*Bptr->polygonset(2, 2);

Bptr->area();

Bptr->perimeter();

Bptr->display();\*/

square sq;

rectangle rec;

triangle tri;

Bptr = &sq;

Bptr->polygonset(2, 2);

Bptr->area();

Bptr->perimeter();

Bptr->display();

Bptr = &rec;

Bptr->polygonset(2, 2);

Bptr->area();

Bptr->perimeter();

Bptr->display();

Bptr = &tri;

Bptr->polygonset(2, 2);

Bptr->area();

Bptr->perimeter();

Bptr->display();

}

// Class.h file (class header file)

#pragma once

//#ifndef interface2.h

#include<string>

#include<iostream>

using namespace std;

//#define CLASS\_H

class polygon

{

protected:

float length;

float width;

public:

polygon(int x, int y);

void polygonset(int x, int y);

float getlen();

float getwid();

virtual float area() const = 0;

virtual float perimeter() const = 0;

virtual void display() const = 0;

~polygon(){}

};

class square:public polygon

{

public:

square();

float area() const;

float perimeter() const;

void display() const;

~square(){}

};

class rectangle :public polygon

{

public:

rectangle();

float area()const;

float perimeter() const;

void display() const;

~rectangle() {}

};

class triangle :public polygon

{

public:

triangle();

float area() const;

float perimeter() const;

void display()const;

~triangle() {}

};

//implementation

#include "interface2.h"

polygon::polygon(int x ,int y)

{

length = x;

width = y;

}

void polygon::polygonset(int x, int y)

{

length = x;

width = y;

}

square::square() :polygon(length, width)

{

}

rectangle::rectangle() :polygon(length, width)

{

}

triangle::triangle() :polygon(length, width)

{

}

float polygon:: getlen()

{

return length;

}

float polygon::getwid()

{

return width;

}

float square::area()const

{

return length \* length;

}

float square::perimeter() const

{

return 4 \* length;

}

void square::display() const

{

cout << " area of square is : " << area() << endl;

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

}

float rectangle::area() const

{

return 2 \* (length \* width);

}

float rectangle::perimeter() const

{

return 2 \* (length + width);

}

void rectangle::display() const

{

cout << " area of rectangle is : " << this->area() << endl;

cout << " perimeter of rectangle is : " << this->perimeter() << endl;

}

float triangle::area() const

{

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

}

float triangle::perimeter() const

{

return length + width;

}

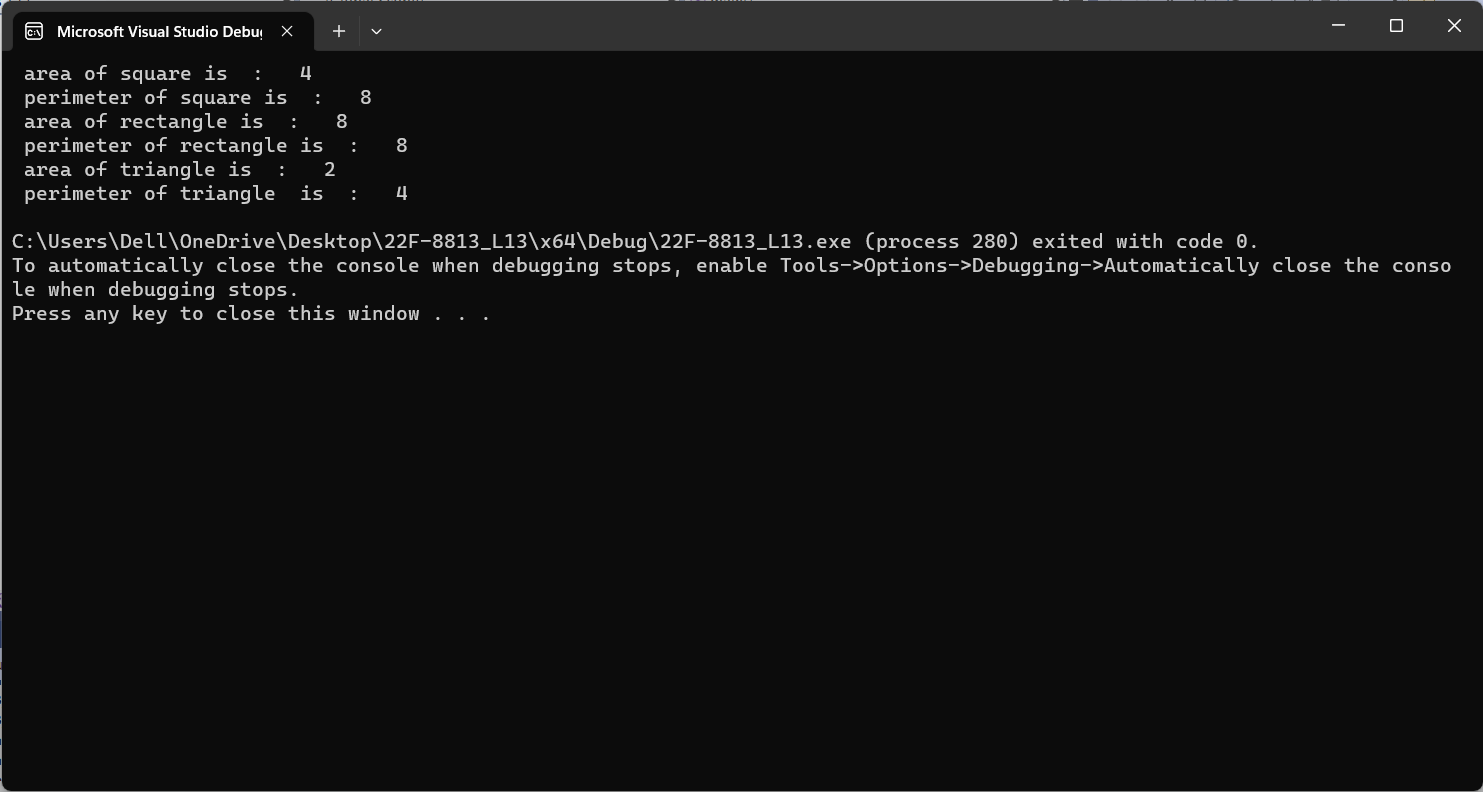
void triangle::display() const

{

cout << " area of triangle is : " << this->area() << endl;

cout << " perimeter of triangle is : " << this->perimeter() << endl;

}



Task 3:

// Class.h file (class header file)

#pragma once

//#ifndef interface3.h

#include<string>

#include<iostream>

using namespace std;

//#define CLASS\_H

class bill

{

protected:

int units;

int per\_unit\_cost;

float calBill;

public:

bill(int x, int y, float z);

virtual ~bill()

{

}

virtual float monthly\_bill()=0;

virtual void bill\_display()=0;

void set(int x,int y);

};

class janbill:public bill

{

public:

janbill();

float monthly\_bill();

void bill\_display();

~janbill()

{

}

};

class febbill:public bill

{

public:

febbill();

float monthly\_bill();

void bill\_display();

~febbill()

{

}

};

class marbill:public bill

{

public:

marbill();

float monthly\_bill();

void bill\_display();

~marbill()

{

}

};

class aprbill:public bill

{

public:

aprbill();

float monthly\_bill();

void bill\_display();

~aprbill()

{

}

};

class maybill:public bill

{

public:

maybill();

float monthly\_bill();

void bill\_display();

~maybill()

{

}

};

#include "interface3.h"

bill::bill(int x, int y, float z)

{

units = x;

per\_unit\_cost = y;

calBill = z;

}

janbill::janbill() :bill(units, per\_unit\_cost, calBill)

{

}

febbill::febbill() :bill(units, per\_unit\_cost, calBill)

{

}

marbill::marbill() :bill(units, per\_unit\_cost, calBill)

{

}

aprbill::aprbill() :bill(units, per\_unit\_cost, calBill)

{

}

maybill::maybill() :bill(units, per\_unit\_cost, calBill)

{

}

void febbill::bill\_display()

{

cout << "feb bill is : " << this->monthly\_bill() << endl;

}

void marbill::bill\_display()

{

cout << "march bill is : " << this->monthly\_bill() << endl;

}

void aprbill::bill\_display()

{

cout << "april bill is : " << this->monthly\_bill() << endl;

}

void maybill::bill\_display()

{

cout << "may bill is : " << this->monthly\_bill() << endl;

}

void janbill::bill\_display()

{

cout << "january bill is : " << this->monthly\_bill() << endl;

}

float janbill::monthly\_bill()

{

return per\_unit\_cost \* units;

}

float febbill::monthly\_bill()

{

return per\_unit\_cost \* units;

}

float marbill::monthly\_bill()

{

return per\_unit\_cost \* units;

}

float aprbill::monthly\_bill()

{

return per\_unit\_cost \* units;

}

float maybill::monthly\_bill()

{

return per\_unit\_cost \* units;

}

void bill::set(int x, int y)

{

units = x;

per\_unit\_cost = y;

}

#include<iostream>

#include<string>

#include"interface3.h"

using namespace std;

int main()

{

bill\* ptr;

janbill jo;

febbill fo;

marbill mo;

aprbill ao;

maybill mao;

ptr = &jo;

ptr->set(2 ,2);

ptr->monthly\_bill();

ptr->bill\_display();

ptr = &fo;

ptr->set(5, 5);

ptr->monthly\_bill();

ptr->bill\_display();

ptr = &mo;

ptr->set(6, 6);

ptr->monthly\_bill();

ptr->bill\_display();

ptr = &ao;

ptr->set(8, 8);

ptr->monthly\_bill();

ptr->bill\_display();

ptr = &mao;

ptr->set(10, 10);

ptr->monthly\_bill();

ptr->bill\_display();

}

