National University of Computer and Emerging Sciences



Assignment 06,07

Object Oriented Programming

|  |  |
| --- | --- |
| Name | Muhammad Zain |
| Roll No. | 19F-0228 |
| INSTRUCTOR | Dr Danish |
| Semester | Spring 2020 |

Task 1:

# **Source Code:**

#include<iostream>

#include<string>

#include<string.h>

using namespace std;

class Person

{

protected:

string name;

string bloodGroup;

public:

Person()

{

name = "-";

bloodGroup = "-";

}

virtual void setData() = 0;

virtual void display() = 0;

};

class A :public Person

{

public:

void setData()

{

cout << "Enter the name and blood group of the person " << endl;

cout << "Name = "; cin >> name;

cout << "Blood Group = "; cin >> bloodGroup;

}

void display()

{

cout << "The Name of the Person is " << name << endl;

cout << "The Blood Group of the Person is " << bloodGroup << endl;

}

};

class B :public Person

{

public:

void setData()

{

cout << "Enter the name and blood group of the person " << endl;

cout << "Name = "; cin >> name;

cout << "Blood Group = "; cin >> bloodGroup;

}

void display()

{

cout << "The Name of the Person is " << name << endl;

cout << "The Blood Group of the Person is " << bloodGroup << endl;

}

};

class C :public Person

{

public:

void setData()

{

cout << "Enter the name and blood group of the person " << endl;

cout << "Name = "; cin >> name;

cout << "Blood Group = "; cin >> bloodGroup;

}

void display()

{

cout << "The Name of the Person is " << name << endl;

cout << "The Blood Group of the Person is " << bloodGroup << endl;

}

};

int main()

{

Person \*obj\_person;

A obj\_A;

obj\_person = &obj\_A;

obj\_person->setData();

obj\_person->display();

B obj\_B;

obj\_person = &obj\_B;

obj\_person->setData();

obj\_person->display();

C obj\_C;

obj\_person = &obj\_C;

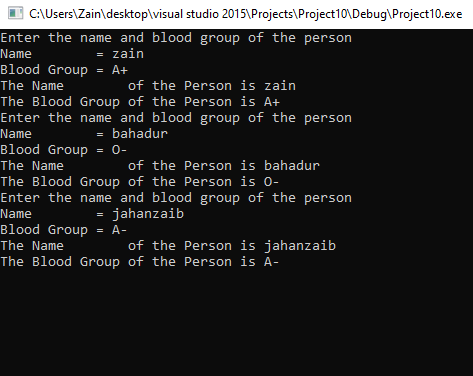
obj\_person->setData();

obj\_person->display();

system("pause>0");

}

# **Snip:**



Task 2:

# **Source Code:**

#include<iostream>

using namespace std;

class Shape {

protected:

float length;

float width;

float height;

public:

Shape() {}

virtual float setArea() = 0;

virtual float setPerimeter() = 0;

virtual void display() = 0;

};

class Rectangle :public Shape {

public:

Rectangle() {}

Rectangle(float x, float y)

{

length = x;

width = y;

}

float setArea()

{

return (length\*width);

}

float setPerimeter()

{

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

}

void display()

{

cout << "The Area Of Rectangle is " << setArea() << " cm^2" << endl;

cout << "The Perimeter Of Rectangle is " << setPerimeter() << " cm " << endl;

}

};

class Triangle :public Shape {

public:

Triangle() {}//Default constructor

Triangle(float x, float y, float z)//parametirazied constructor

{

length = x;

width = y;

height = z;

}

float setArea() {

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

}

float setPerimeter() {

return (height + length + width);

}

void display()

{

cout << "The Area Of Triangle is " << setArea() << " cm^2" << endl;

cout << "The Perimeter Of Triangle is " << setPerimeter() << "cm " << endl;

}

};

class Square :public Shape

{

public:

Square() {}//Defqault constructor

Square(float a)

{

length = a;

}

float setArea()

{

return length\*length;

}

float setPerimeter() {

return 4 \* length;

}

void display()

{

cout << "The Area Of Square is " << setArea() << " cm^2" << endl;

cout << "The Perimeter Of Square is " << setPerimeter() << " cm " << endl;

}

};

int main()

{

int choice;

Shape \*Bptr;

while (int x = 1)

{

cout << "\tIf you want to calculate the area of Rectagle Press 1 " << endl;

cout << "\tIf you want to calculate the area of Tritagle Press 2 " << endl;

cout << "\tIf you want to calculate the area of Square Press 3 " << endl;

cout << "\tIf you want to Exit.... Press 4 " << endl;

cout << endl;

cin >> choice;

//system("cls");

//cout<<endl<<endl<<endl;

if (choice == 1)

{

float a, b;

cout << "Enter the Length of Rectangle" << endl;

cin >> a;

cout << "Enter the Width of Rectangle" << endl;

cin >> b;

Rectangle obj\_R(a, b);

Bptr = &obj\_R;

Bptr->display();

cout << endl << endl << endl;

}

else if (choice == 2)

{

float a, b, c;

cout << "Enter the Base of Triangle" << endl;

cin >> a;

cout << "Enter the Height of Triangle" << endl;

cin >> c;

cout << "Enter the Width of Rectangle" << endl;

cin >> b;

Triangle obj\_T(a, b, c);

Bptr = &obj\_T;

Bptr->display();

cout << endl << endl << endl;

}

else if (choice == 3)

{

float a;

cout << "Enter the length of one side of Square" << endl;

cin >> a;

Square obj\_S(a);

Bptr = &obj\_S;

Bptr->display();

cout << endl << endl << endl;

}

else if (choice == 4)

{

break;

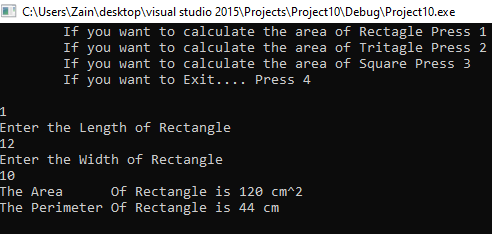
}

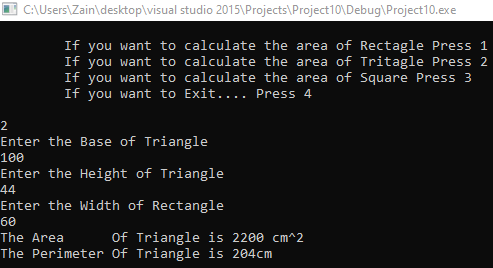
}

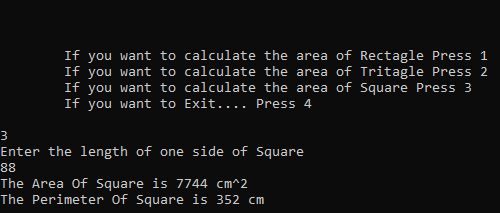
system("pause>0");

}

# **Snip:**







Task 3:

# **Source Code:**

#include<iostream>

#include<string.h>

using namespace std;

class Animal

{

protected:

string name;

public:

Animal()

{

name = "\_";

}

virtual void sound()

{

cout << "Virtual animal Sound Function" << endl;

}

};

class cat :public Animal

{

public:

cat()

{

name = "Cat";

}

void sound()

{

cout << "Cat Sound is Meow-Meow" << endl;

}

};

class dog :public Animal

{

public:

dog()

{

name = "Dog";

}

void sound()

{

cout << "Dog Barks" << endl;

}

};

class Tiger\_family :public Animal

{

public:

void sound()

{

cout << " Sound of Tiger family is ";

}

};

class deer :public Animal

{

public:

deer()

{

name = "deer";

}

void sound()

{

cout << "Deer bellows" << endl;

}

};

class tiger :public Tiger\_family

{

public:

tiger()

{

name = "Tiger";

}

void sound()

{

cout << "Tiger Roars" << endl;

}

};

class lion :public Tiger\_family

{

public:

lion()

{

name = "Lion";

}

void sound()

{

cout << "Sound of lion: Awwrrr" << endl;

}

};

class leopard : public Tiger\_family

{

public:

leopard()

{

name = "Tiger";

}

void sound()

{

cout << "Sound of leopard: Hwwrrr" << endl;

}

};

int main()

{

Animal \*obj\_animal;

obj\_animal = new cat();

obj\_animal->sound();

obj\_animal = new dog();

obj\_animal->sound();

obj\_animal = new deer();

obj\_animal->sound();

obj\_animal = new tiger();

obj\_animal->sound();

obj\_animal = new lion();

obj\_animal->sound();

obj\_animal = new leopard();

obj\_animal->sound();

//lion objl;

//objl = obj\_animal;

//objl->sound();

// Severity Code Description Project File Line Suppression State

// Error C2679 binary '=': no operator found which takes a right - hand operand of type 'Animal \*' (or there is no acceptable conversion)

//Severity Code Description Project File Line Suppression State

// Error C2819 type 'lion' does not have an overloaded member 'operator>'

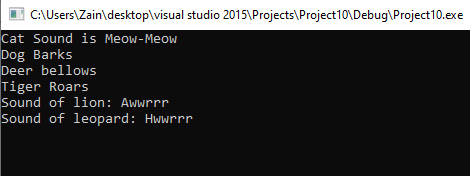
//Severity Code Description Project File Line Suppression State

// Error C2232 '->lion::sound': left operand has 'class' type, use '.'

system("pause>0");

}

# **Snip:**



Task 5:

# **Source Code:**

#include<iostream>

#include<string>

using namespace std;

class BankAccount

{

protected:

string accountHolderName;

double accountNo;

double Balance;

public:

BankAccount()

{

cout << "Enter your name" << endl;

cin.ignore();

getline(cin, accountHolderName);

cout << "Enter the Account Number" << endl;

cin >> accountNo;

while (accountNo<=0)

{

cout << "Please input valid account number" << endl;

cin >> accountNo;

}

cout << "Enter the balance" << endl;

cin >> Balance;

while (Balance<=0)

{

cout << "Please input more amount to open the account" << endl;

cin >> Balance;

}

}

void checkBalance()

{

cout << "Your Balance in Account is " << Balance << endl;

}

virtual void withdraw() = 0;// pure virtual

virtual void deposit() = 0;// pure virtual

};

class SavingAccount :public BankAccount

{

protected:

double inAmount=0;

double outAmount=0;

public:

SavingAccount()

{

cout << "This is Saving Account Class Constructor" << endl;

}

void withdraw()

{

cout << "Enter the Amount to withdraw " << endl;

cin >> outAmount;

}

double update1()

{

return(outAmount+(outAmount\*0.1));

}

void deposit()

{

cout << "Enter the Amount to deposit " << endl;

cin >> inAmount;

}

double update2()

{

return((inAmount\*0.05) + inAmount);

}

void output()

{

double temp1 = update1();

double temp2 = update2();

double newBalance = Balance - temp1 + temp2;

cout << "New Balance = " << newBalance << endl;

}

};

class CheckingAccount :public BankAccount

{

protected:

double inAmount = 0;

double outAmount = 0;

public:

CheckingAccount()

{

cout << "This is Checking Account Class Constructor" << endl;

}

void withdraw()

{

cout << "Enter the Amount to withdraw " << endl;

cin >> outAmount;

}

double update1()

{

return(outAmount);

}

void deposit()

{

cout << "Enter the Amount to deposit " << endl;

cin >> inAmount;

}

double update2()

{

return(inAmount);

}

void output()

{

double temp1 = update1();

double temp2 = update2();

double newBalance = Balance - temp1 + temp2;

cout << "New Balance = " << newBalance << endl;

}

};

class InvestmentAccount :public BankAccount

{

protected:

double inAmount = 0;

double outAmount = 0;

int deduction;

public:

InvestmentAccount()

{

cout << "This is Investment Account Class Constructor" << endl;

}

void withdraw()

{

cout << "Enter the Amount to withdraw " << endl;

cin >> outAmount;

cout << "The Deduction Fee is set to Rs % by the bank" << endl;

deduction = 5;

}

double update1()

{

return(outAmount+deduction);

}

void deposit()

{

cout << "Enter the Amount to deposit " << endl;

cin >> inAmount;

}

double update2()

{

return(inAmount+(inAmount\*0.05));

}

void output()

{

double temp1 = update1();

double temp2 = update2();

double newBalance = Balance - temp1 + temp2;

cout << "New Balance = " << newBalance << endl;

}

};

int main()

{

int choice;

while (1)

{

cout << "Press 1 for Saving Account Class" << endl;

cout << "Press 2 for Checking Account Class" << endl;

cout << "Press 3 for Investment Account Class" << endl;

cout << "Press 4 to Exit---------------------" << endl << endl;

cout << "Enter option : "; cin >> choice;

if (choice == 1)

{

SavingAccount obj\_S;

obj\_S.withdraw();

obj\_S.output();

obj\_S.deposit();

obj\_S.output();

}

else if (choice == 2)

{

CheckingAccount obj\_C;

obj\_C.withdraw();

obj\_C.output();

obj\_C.deposit();

obj\_C.output();

}

else if (choice == 3)

{

InvestmentAccount obj\_I;

obj\_I.withdraw();

obj\_I.output();

obj\_I.deposit();

obj\_I.output();

}

else if (choice == 4)

{

cout << "\t\tThankyou :-)" << endl;

break;

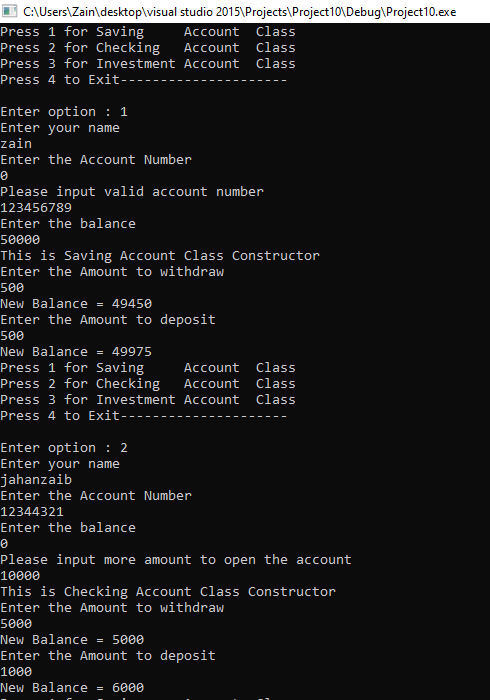
}

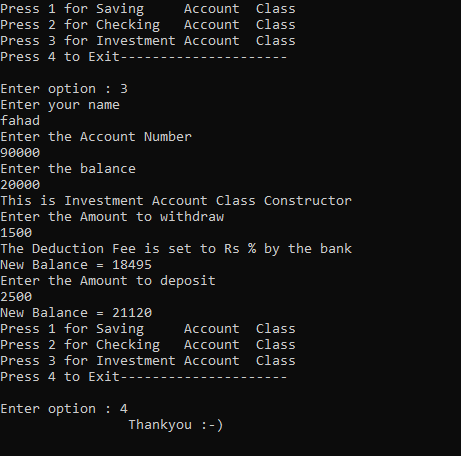
}

system("pause>0");

}

# **Snip:**





Task 6:

# **Source Code:**

#include<iostream>

#include<string>

using namespace std;

class Employee

{

protected:

string Name;

char hireYear[4];

public:

Employee()

{

cout << "Enter your name" << endl;

cin.ignore();

getline(cin, Name);

cout << "Enter the Hiring Year" << endl;

for (int i = 0; i < 4; i++)

{

cin >> hireYear[i];

}

}

virtual void monthlyPay() = 0;// pure virtual

virtual void annualPay() = 0;// pure virtual

};

class SalariedEmployee :public Employee

{

protected:

int monthlySalary;

public:

SalariedEmployee()

{

cout << "Enter your Monthly Salary : "; cin >> monthlySalary;

}

void monthlyPay()

{

cout << "Monthly Pay of Salaried Employee is " << monthlySalary << endl;

}

void annualPay()

{

cout << "Yearly Pay of Salaried Employee is " << 12 \* monthlySalary << endl;

}

};

class HourlyEmployee :public Employee

{

protected:

int hoursPerWeek;

int hourlyWage;

public:

HourlyEmployee()

{

cout << "Enter the Hours Worked per week : "; cin >> hoursPerWeek;

cout << "Enter the Rate of hourly Work : "; cin >> hourlyWage;

}

void monthlyPay()

{

cout << "Monthly Pay of Hourly Employee is " << 4 \* hoursPerWeek\*hourlyWage << endl;

}

void annualPay()

{

cout << "Yearly Pay of Hourly Employee is " << 12 \* 4 \* hoursPerWeek\*hourlyWage << endl;

}

};

class Manager :public SalariedEmployee

{

public:

Manager()

{

cout << "This is Managers Constructor" << endl;

}

void output()

{

monthlyPay();

annualPay();

}

};

class Staff :public SalariedEmployee

{

public:

Staff()

{

cout << "This is Staff Constructor" << endl;

}

void output()

{

monthlyPay();

annualPay();

}

};

class FullTimeEmployee :public HourlyEmployee

{

public:

FullTimeEmployee()

{

cout << "This is Full Time Employee Constructor" << endl;

}

void output()

{

monthlyPay();

annualPay();

}

};

class PartTimeEmployee :public HourlyEmployee

{

public:

PartTimeEmployee()

{

cout << "This is Part Time Employee Constructor" << endl;

}

void output()

{

monthlyPay();

annualPay();

}

};

int main()

{

int choice;

while (1)

{

cout << "Press 1 for Salaried Employee Class" << endl;

cout << "Press 2 for Hourly Employee Class" << endl;

cout << "Press 3 to Exit---------------------" << endl << endl;

cout << "Enter option : "; cin >> choice;

if (choice == 1)

{

Manager obj\_M;

while (1)

{

cout << "Press 1 to Manager Salary" << endl;

cout << "Press 2 to Staff Salary" << endl;

cout << "Press 3 to Return to main menu " << endl;

int opt;

cin >> opt;

if (opt == 1)

{

obj\_M.output();

}

else if (opt == 2)

{

Staff obj\_S;

obj\_S.output();

}

else if (opt == 3)

{

break;

}

}

}

else if (choice == 2)

{

while (1)

{

cout << "Press 1 for Full-time Employee Class" << endl;

cout << "Press 2 for Part-time Employee Class" << endl;

cout << "Press 3 to Return to main menu " << endl;

int opt;

cin >> opt;

if (opt == 1)

{

FullTimeEmployee obj\_F;

obj\_F.output();

}

else if (opt == 2)

{

PartTimeEmployee obj\_P;

obj\_P.output();

}

else if (opt == 3)

{

break;

}

}

}

else if (choice == 3)

{

cout << "\t\tThankyou :-)" << endl;

break;

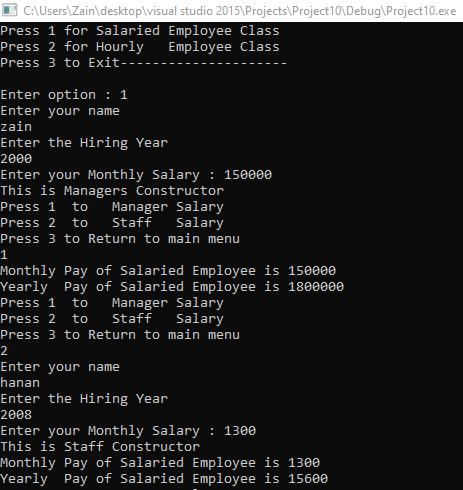
}

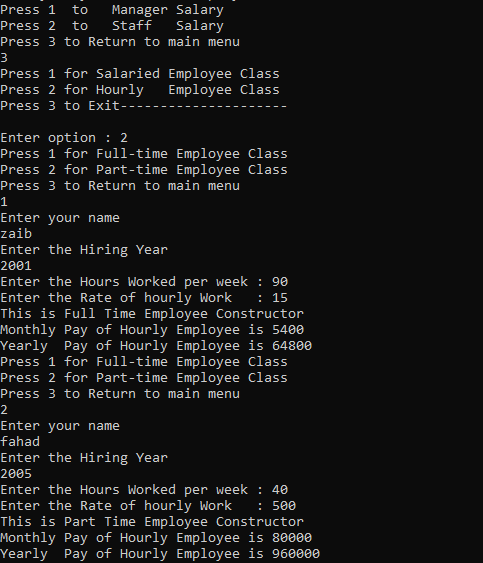
}

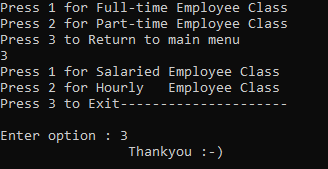
system("pause>0");

}

# **Snip:**







Task 7:

# **Source Code:**

#include<iostream>

using namespace std;

template <class t1>

class Calculator

{

private:

t1 value1;

t1 value2;

public:

Calculator() {}

Calculator(t1 a, t1 b)

{

value1 = a;

value2 = b;

}

t1 Addition()

{

return (value1 + value2);

}

t1 Subtraction()

{

return (value1 - value2);

}

t1 Multiplication()

{

return (value1\*value2);

}

t1 Division();//just to learn the syntax of definig function outside class

t1 Modulus();

};

template <class t1> t1 Calculator<t1>::Division()

{

return value1 / value2;

}

template <class t1> t1 Calculator<t1>::Modulus()

{

t1 temp = static\_cast<int>(value1) % static\_cast<int>(value2);

return temp;

}

int main()

{

cout << "\t\tCalculator" << endl;

double int1;

double int2;

try {

cout << "Input 2 values " << endl;

//cout << "Value 1 : ";

//cin >> int1;

//cout << "Value 2 : ";

//cin >> int2;

while (!(cin >> int1 >> int2))

{

throw int1, int2;

}

}

catch (...)

{

cout << "Your Input contains String of Character" << endl;

cout << endl;

//Clear the previous input ";

cin.clear();

//Discard the previous input";

cin.ignore(123, '\n');

cout << "Enter numbers on both inputs " << endl;

cout << "Number 1 = "; cin >> int1;

cout << "Number 2 = "; cin >> int2;

}

Calculator <double>obj(int1, int2);

cout << endl << endl;

cout << "Addition : " << obj.Addition() << endl;

cout << "Subtraction : " << obj.Subtraction() << endl;

cout << "Multiplication : " << obj.Multiplication() << endl;

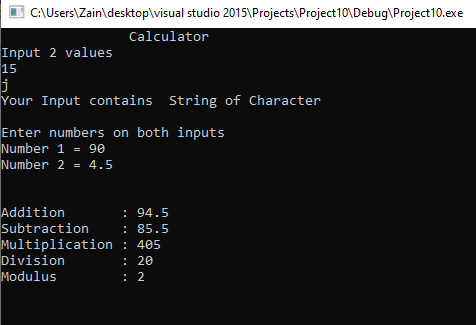
cout << "Division : " << obj.Division() << endl;

cout << "Modulus : " << obj.Modulus() << endl;

system("pause>0");

}

# **Snip:**



Task 8:

# **Source Code:**

#include <iostream>

using namespace std;

const int Max = 10;

template <class Temp>

class Stack

{

private:

Temp arr[10]; // Stack apna

public:

Temp item;

Temp var;

int top; //Lateste jo Top pa data push kea h

Stack()

{

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

{

arr[i] = NULL;

//pahlay 0 to all initialization

}

top = -1;

//Empty stack wala scene kea h

}

void push(Temp a)

{//-1 sa 0 krnay k lea

top++;

if (top < Max)

{

arr[top] = a;

}

else

{

cout << "Full Stack" << endl;

top--;

}

}

Temp isEmpty()

{

if (top == -1)

{

cout << "Empty Stack " << endl;

return 0;

}

}

Temp pop()

{

isEmpty();

Temp data = arr[top];

arr[top] = 0;

top--;

return data;

}

};

int main()

{

int choice;

Stack<int> obj;

while (1)

{

cout << "\tThe Capacity of the Stack is " << ((Max - obj.top) - 1) << endl;

cout << "\tPress 1 to Push Data" << endl;

cout << "\tPress 2 to Pop Data" << endl;

cout << "\tPress 3 to Check Whether the stack is empty or not " << endl;

cout << "\tPress 4 to Exit------------------------------------" << endl << endl;

cout << "Enter your choice :";cin >> choice;

if (choice == 1)

{

cout << "Enter a number to Push at the top of the Stack" << endl;

cin >> obj.item;

obj.push(obj.item);

cout << endl;

cout << endl << endl;

}

else if (choice==2)

{ obj.var = obj.pop();

cout << "Item popped from Stack is:" << obj.var << endl;

cout << endl << endl;

}

else if (choice == 3)

{

obj.isEmpty();

cout << endl << endl;

}

else if (choice == 4)

{

break;

}

}

return 0;

}

# **Snip:**

