Task 1:

Main File :

#include <iostream>

#include "PersonData.h"

#include "CustomerData.h"

using namespace std;

void displayCustomer(CustomerData c) {

cout << "Last Name is : " << c.getLastName() << endl;

cout << "First Name is : " << c.getFirstName() << endl;

cout << "Address is : " << c.getAddress() << endl;

cout << "Customer's Number is : " << c.getCustomerNumber() << endl;

cout << "Mailing List is : ";

if (!c.getMailingList())

cout << "No\n\n";

else

cout << "Yes\n\n";

}

int main(){

string name, lastname, address;

int number;

cout << "Name: ";

cin >> name;

cout << "Last Name: ";

cin >> lastname;

cout << "Address: ";

cin >> address;

cout << "Number : ";

cin >> number;

CustomerData customer1(name, lastname, address, number, false);

displayCustomer(customer1);

CustomerData customer2;

cout << "Name: ";

cin >> name;

cout << "Last Name: ";

cin >> lastname;

cout << "Address: ";

cin >> address;

cout << "Number : ";

cin >> number;

customer2.setter4(name);

customer2.setter3(lastname);

customer2.setter2(address);

customer2.setter1(number);

customer2.setter(true);

displayCustomer(customer2);

return 0;

}

**Header File :**

#ifndef PERSONDATA\_H

#define PERSONDATA\_H

#include <string>

using namespace std;

class PersonData{

private:

string lastName;

string firstName;

string address;

public:

PersonData(string ln, string fn, string addr) {

lastName = ln;

firstName = fn;

address = addr;

}

PersonData(){

lastName = "";

firstName = "";

address = "";

}

void setter3(string ln) {

lastName = ln;

}

void setter4(string fn){

firstName = fn;

}

void setter2(string addr){

address = addr;

}

string getLastName() const{

return lastName;

}

string getFirstName() const{

return firstName;

}

string getAddress() const{

return address;

}

};

#endif

**Header File:**

#ifndef CUSTOMERDATA\_H

#define CUSTOMERDATA\_H

#include <string>

using namespace std;

class CustomerData : public PersonData{

private:

int customerNumber; // Customer number

bool mailingList; // Include in mailing list?

public:

CustomerData(string ln, string fn, string addr, int cn, bool ml) :

PersonData(ln, fn, addr){

customerNumber = cn;

mailingList = ml;

}

CustomerData() : PersonData(){

customerNumber = 0;

mailingList = false;

}

void setCustomerNumber(int cn){

customerNumber = cn;

}

void setMailingList(bool ml){

mailingList = ml;

}

int getCustomerNumber(){

return customerNumber;

}

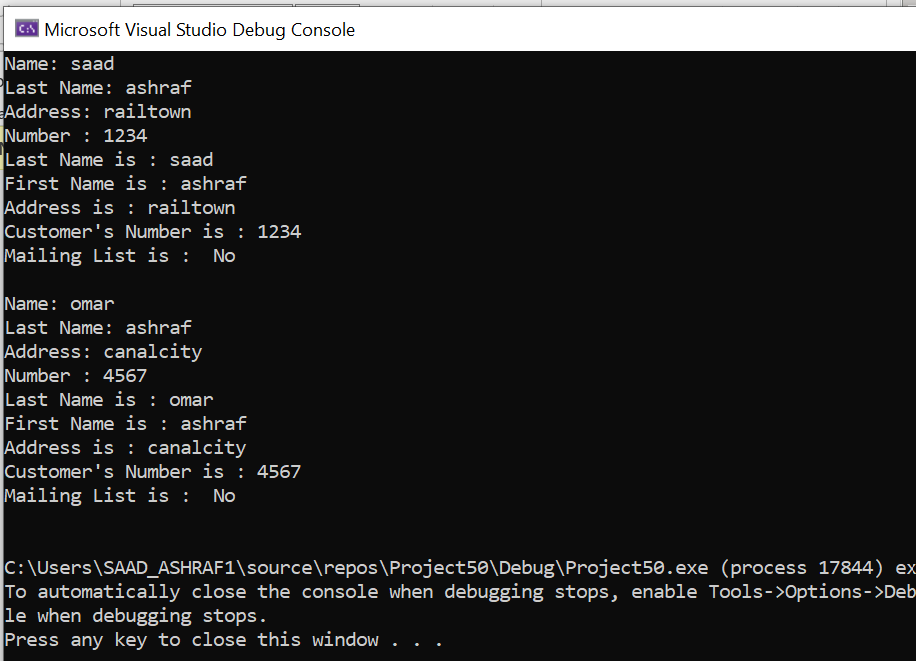
bool getMailingList(){

return mailingList;

}

};

#endif



Task 2:

MAIN:

#include <iostream>

#include <iomanip>

#include "PreferredCustomer.h"

using namespace std;

int main(){

string name, lastname, address;

int number, amount;

cout << "Name: ";

cin >> name;

cout << "Last Name: ";

cin >> lastname;

cout << "Address: ";

cin >> address;

cout << "Number : ";

cin >> number;

cout << "amount : ";

cin >> amount;

PreferredCustomer customer;

double purchase;

customer.setLastName(lastname);

customer.setFirstName(name);

customer.setAddress(address);

customer.setCustomerNumber(number);

customer.setMailingList(true);

customer.setPurchasesAmount(amount);

cout << fixed << showpoint << setprecision(2);

cout << "Last name: " << customer.getLastName() << endl;

cout << "First name: " << customer.getFirstName() << endl;

cout << "Address: " << customer.getAddress() << endl;

cout << "Customer number: "<< customer.getCustomerNumber()<< endl;

if (!customer.getMailingList())

cout << "Mail : No\n";

else

cout << "Mail : YES\n";

cout << "Purchases amount: $"<< customer.getPurchasesAmount()<< endl;

cout << " discount : " << customer.getDiscountLevel()<< endl;

cout << "Enter additional purchase amount: ";

cin >> purchase;

customer.addPurchase(purchase);

cout << "Purchases amount: $"<< customer.getPurchasesAmount()<< endl;

cout << " discount : "<< customer.getDiscountLevel() << endl;

return 0;

}

HEADER FILE

#ifndef PREFERREDCUSTOMER\_H

#define PREFERREDCUSTOMER\_H

#include "CustomerData.h"

class PreferredCustomer : public CustomerData{

private:

double purchasesAmount;

double discountLevel;

void determineDiscountLevel() {

}

public:

PreferredCustomer() : CustomerData(){

purchasesAmount = 0; discountLevel = 0;

}

PreferredCustomer(string ln, string fn, string addr, int cn, bool ml, double pa, double dl) :

CustomerData(ln, fn, addr,cn, ml){

purchasesAmount = pa; discountLevel = dl;

}

void setPurchasesAmount(double p){

purchasesAmount = p;

determineDiscountLevel();

}

double getPurchasesAmount(){

return purchasesAmount;

}

double getDiscountLevel(){

determineDiscountLevel();

return discountLevel;

}

void addPurchase(double p){

purchasesAmount += p;

}

};

#endif

HEADER FILE:

#ifndef CUSTOMERDATA\_H

#define CUSTOMERDATA\_H

#include <string>

#include "PersonData.h"

using namespace std;

class CustomerData : public PersonData{

private:

int customerNumber;

bool mailingList;

public:

CustomerData(string ln, string fn, string addr, int cn, bool ml) :

PersonData(ln, fn, addr){

customerNumber = cn;

mailingList = ml;

}

CustomerData() : PersonData(){

customerNumber = 0;

mailingList = false;

}

void setCustomerNumber(int cn){

customerNumber = cn;

}

void setMailingList(bool ml){

mailingList = ml;

}

int getCustomerNumber(){

return customerNumber;

}

bool getMailingList() {

return mailingList;

}

};

#endif

HEADER :

#ifndef PERSONDATA\_H

#define PERSONDATA\_H

#include <string>

using namespace std;

class PersonData{

private:

string lastName;

string firstName;

string address;

string city;

string state;

string zip;

string phone;

public:

PersonData(string ln, string fn, string addr){

lastName = ln;

firstName = fn;

address = addr;

}

PersonData(){

lastName = "";

firstName = "";

address = "";

}

void setLastName(string ln){

lastName = ln;

}

void setFirstName(string fn){

firstName = fn;

}

void setAddress(string addr){

address = addr;

}

string getLastName(){

return lastName;

}

string getFirstName(){

return firstName;

}

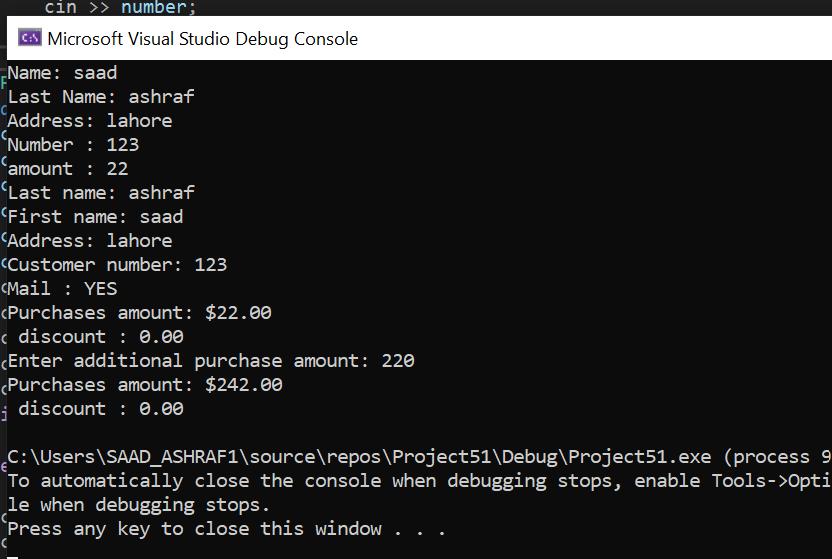
string getAddress(){

return address;

}

};

#endif



Task 3:

MAIN:

#include<iostream>

#include"header.h"

using namespace std;

int main() {

Person p;

UnemployedPerson up;

BusinessMan b;

b.print();

b.EmployedPerson1();

}

HEADER:

#ifndef HEADER\_H

#define HEADER\_H

#include<iostream>

using namespace std;

class Person {

protected:

string name;

private:

int age;

char gender;

public:

Person() {

cout << "Enter Name : ";

cin >> name;

cout << "Enter age : ";

cin >> age;

cout << "ENTER GENDER :";

cin >> gender;

}

};

class EmployedPerson :public Person {

protected:

int nic;

public:

void EmployedPerson1() {

cout << "Enter nic : ";

cin >> nic;

}

};

class UnemployedPerson :public Person {

public:

UnemployedPerson() {

cout << name << endl;

cout << "Hi, I am Unemploy from Unemployed Class" << endl;

}

};

class BusinessMan :public EmployedPerson {

public:

void print() {

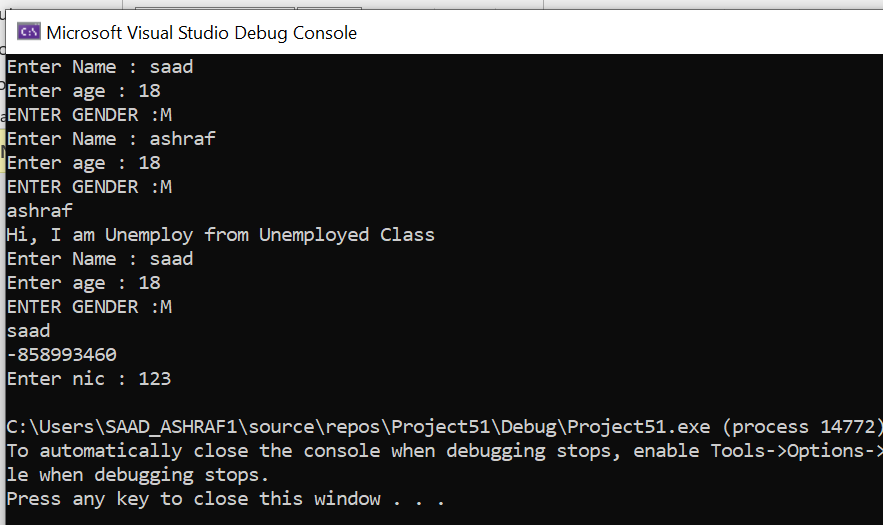
cout << name << endl;

cout << nic << endl;

}

};

#endif



Task 4:

MAIN:

#include<iostream>

#include"PersonData.h"

using namespace std;

int main() {

Vehicle v;

WheelVehicle wv;

WingVehicle wvv;

Truck t;

v.setdistance(20);

cout<<v.getdistance()<<endl;

v.setspeed(20);

cout<<v.getspeed()<<endl;

wv.setwheels(20);

cout<<wv.getwheels()<<endl;

wvv.setwings(20);

cout<<wvv.getwings()<<endl;

t.setcarryingLoad(20);

cout << t.getcarryingLoad() << endl;

}

HEADER:

#include<iostream>

using namespace std;

class Vehicle {

private:

int speed, distance;

public:

void setspeed(int a) {

speed = a;

}

void setdistance(int b) {

distance = b;

}

int getspeed() {

return speed;

}

int getdistance() {

return distance;

}

};

class WheelVehicle :public Vehicle {

private:

int wheels;

public:

void setwheels(int a) {

wheels = a;

}

int getwheels() {

return wheels;

}

};

class WingVehicle :public Vehicle {

private:

int wings;

public:

void setwings(int a) {

wings = a;

}

int getwings() {

return wings;

}

};

class Truck :public WheelVehicle {

private:

int carryingLoad;

public:

void setcarryingLoad(int a) {

carryingLoad = a;

}

int getcarryingLoad() {

return carryingLoad;

}

};

