**qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm**

|  |
| --- |
| Assignment 1(OOP)  MCA SEM-II, LAB-III(OOP)  Sandeep Bhatt, MCA051 |

1. **Define a Class Largest to obtain the largest of three given numbers.**

**Sol:**

#include<iostream>

using namespace std;

class Largest

{

private:

int First\_no,Second\_no,Third\_no;

public:

void input()

{

cout<<"\nEnter First no. :";

cin>>First\_no;

cout<<"\nEnter Second no. :";

cin>>Second\_no;

cout<<"\nEnter Third no. :";

cin>>Third\_no;

}

void calc()

{

int largest;

largest=((First\_no>=Second\_no)&&(First\_no>=Third\_no)?

First\_no:(Second\_no>=First\_no)&&(Second\_no>=Third\_no)?

Second\_no:Third\_no);

cout<<"\nLargest number :"<<largest;

}

};

int main()

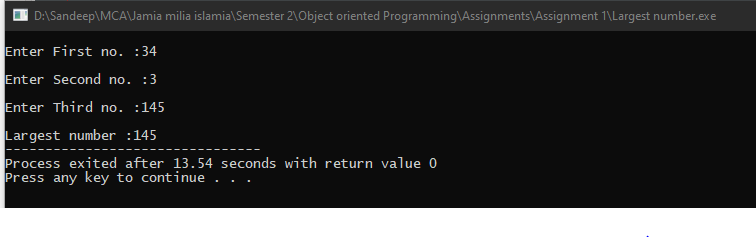
{

Largest l;

l.input();

l.calc(); }

OUTPUT:



1. **Define a class Bank\_Account to represent a bank account. It contains**

**Data Members:**

**Name of the depositor**

**Account Number**

**Type of account**

**Balance**

**Member Functions:**

**To assign initial values**

**To deposit an amount**

**To withdraw an amount < amount available**

**Display the name and balance.**

**Sol:**

#include<iostream>

#include<stdio.h>

#include<string.h>

using namespace std;

**cl**ass bank\_account

{

int acno;

char nm[100], acctype[100];

float bal;

public:

bank\_account(int acc\_no, char \*name, char \*acc\_type, float balance) //Parameterized Constructor

{

acno=acc\_no;

strcpy(nm, name);

strcpy(acctype, acc\_type);

bal=balance;

}

void deposit();

void withdraw();

void display();

};

void bank\_account::deposit() //depositing an amount

{

int deposit\_amount;

cout<<"\n Enter Deposit Amount = ";

cin>>deposit\_amount;

bal+=deposit\_amount;

}

void bank\_account::withdraw() //withdrawing an amount

{

int withdraw\_amount;

cout<<"\n Enter Withdraw Amount = ";

cin>>withdraw\_amount;

if(withdraw\_amount>bal)

cout<<"\n Cannot Withdraw Amount";

bal-=withdraw\_amount;

}

void bank\_account::display() //displaying the details

{

cout<<"\n ----------------------";

// cout<<"\n Accout No. : "<<acno;

cout<<"\n Name : "<<nm;

// cout<<"\n Account Type : "<<acctype;

cout<<"\n Balance : "<<bal;

}

int main()

{

int acc\_no;

char name[100], acc\_type[100];

float balance;

cout<<"\n Enter Details: \n";

cout<<"-----------------------";

cout<<"\n Accout No. ";

cin>>acc\_no;

cout<<"\n Name : ";

cin>>name;

cout<<"\n Account Type : ";

cin>>acc\_type;

cout<<"\n Balance : ";

cin>>balance;

bank\_account b1(acc\_no, name, acc\_type, balance); //object is created

b1.deposit(); //calling member functions

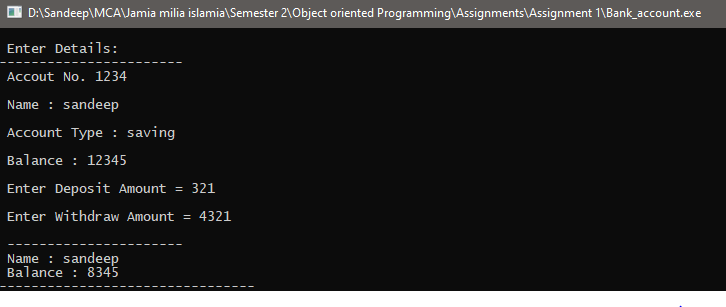
b1.withdraw(); // calling member functions

b1.display(); //displaying the outcome

return 0;

}

Output:



3. **Using the above class Bank\_account and by supplying a user id and password allow**

**users to Login using their id and password. Now if login was successful the user will be able**

**to do the following:**

**Withdraw money.**

**Deposit money.**

**Display Balance.**

**Quit the program.**

**If login was not successful (for example the id or password did not match) then the user will be**

**taken back to the introduction menu.**

**Sol:**

#include<iostream>

using namespace std;

class Bank\_Account

{

private:

string name;

int a\_number;

string type;

float balance;

public:

Bank\_Account(string name,int a\_number,string type,float balance)

{

this->name=name;

this->a\_number=a\_number;

this->type=type;

this->balance=balance;

}

void deposit(int d\_amount)

{

this->balance=this->balance+d\_amount;

cout<<"Amount sucessfully deposited"<<endl;

}

void withdraw(int w\_amount)

{

if(this->balance-w\_amount>5000)

this->balance=this->balance-w\_amount;

else

cout<<"Current balance not sufficient"<<endl;

}

void display()

{

cout<<"Name:"<<this->name<<endl;

cout<<"Current balance:"<<this->balance<<endl;

}

};

int main()

{

string id="",pass="";

int chance=0;

while(id!="Sandeep051"&&pass!="S051")

{

if(chance!=0)

cout<<"Worng id or password!!\nPlease try again"<<endl;

if(chance==3)

{

cout<<"Please try again later"<<endl;

return 0;

}

cout<<"Enter your id:";

cin>>id;

cout<<"Enter your password:";

cin>>pass;

chance++;

}

Bank\_Account obj("Sandeep",1234,"savings",50000);

char c='Y';

int choice,amount;

do

{

cout<<"1.Deposit an amount"<<endl;

cout<<"2.Withdraw an amount"<<endl;

cout<<"3.Display"<<endl;

cout<<"4.Quit"<<endl;

cout<<"Enter you choice:";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Enter the amount to deposit:";

cin>>amount;

obj.deposit(amount);

break;

case 2:

cout<<"Enter the amount to withdraw:";

cin>>amount;

obj.withdraw(amount);

break;

case 3:

obj.display();

break;

case 4:

c='N';

break;

default:

cout<<"Wrong choice"<<endl;

c='N';

break;

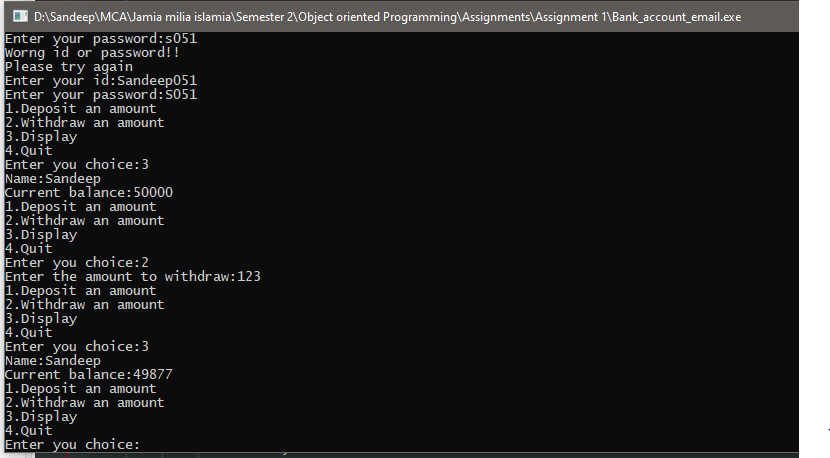
}

}while(c=='Y'||c=='y');

cout<<"Thanks for visiting!"<<endl;

return 0;}

OUTPUT:



**4. Create a class Time to add Two times provided in hour-minute format. Use member**

**functions**

**a)void input() to provide hour and minute.**

**b) void gettime(int,int) to take hour and minute entered by user.**

**c) sum(time <oj> ) to add minutes and hours for two Time objects. If minutes is > 60 add 1 with**

**hour.**

**d) void display() to display the result.**

**Sol:**

#include <iostream>

using namespace std;

class Time

{

private:

int hours;

int minutes;

int seconds;

public:

void getTime(void);

void putTime(void);

void addTime(Time T1,Time T2);

};

void Time::getTime(void)

{

cout << "Enter time:" << endl;

cout << "Hours? "; cin>>hours;

cout << "Minutes? "; cin>>minutes;

cout << "Seconds? "; cin>>seconds;

}

void Time::putTime(void)

{

cout << endl;

cout << "Time after add: ";

cout << hours << ":" << minutes << ":" << seconds << endl;

}

void Time::addTime(Time T1,Time T2)

{

this->seconds=T1.seconds+T2.seconds;

this->minutes=T1.minutes+T2.minutes + this->seconds/60;;

this->hours= T1.hours+T2.hours + (this->minutes/60);

this->minutes %=60;

this->seconds %=60;

}

int main()

{

Time T1,T2,T3;

T1.getTime();

T2.getTime();

//add two times

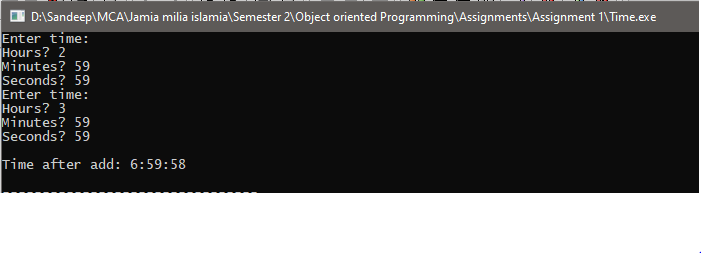
T3.addTime(T1,T2);

T3.putTime();

return 0;

}

**Output:**



**5. Write a C++ program to add two complex numbers using class Complex having real and**

**imag as data members.**

**(i) The class Complex contains three constructors.**

**One with no parameter. (Used for the object for storing result.)**

**With one parameter(Same value for real and imaginary part)**

**With two parameters (Different Values for real and imaginary part)**

**ii) Two friend functions**

**a) One to add two complex numbers by taking two reference variables of class complex**

**and returning another reference.**

**b) To display the result.**

**Sol:**

//adding two complex numbers using class

#include<iostream>

using namespace std;

class complex

{

int real,imag;

public:

void set()

{

cout<<"\nEnter real and imag part : ";

cin>>real>>imag;

}

friend complex sum(complex,complex);

void display();

};

void complex::display()

{

cout<<"\nThe sum of complex num is : "<<real<<" +i "<<imag;

}

complex sum(complex a,complex b)

{

complex t;

t.real=a.real+b.real;

t.imag=a.imag+b.imag;

return t;

}

int main()

{

complex a,b,c;

a.set();

b.set();

c=sum(a,b);

c.display();

return(0);

}

Output:

