Rajokari institute of technology
Rajokari , Delhi - 110038

(OOPS)Practical submitted.

Name-:*Ayush Kr.*

Enrollment-:*1913111014*

Department-:*ITESM*

Semester-: *3nd*

Practical File Submitted To :-

Antima jain.

(Govt. Of NCT Delhi
Deputy.Of Training and Technology Ed.)

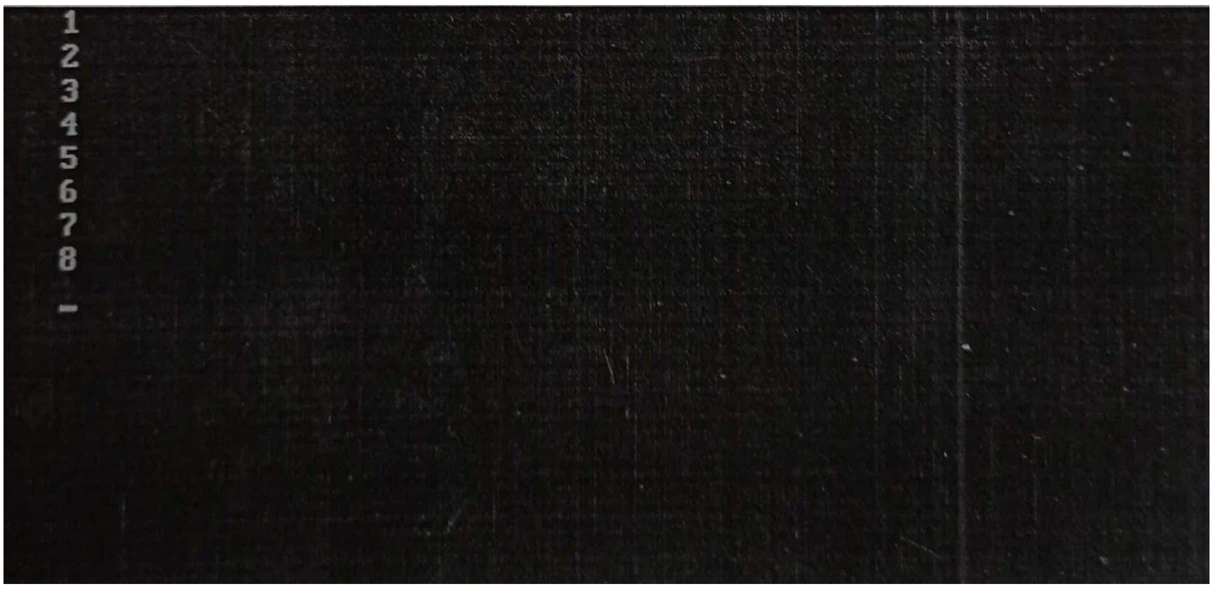
PRACTICAL -: 1

***AIM :-* WRITE A PROGRAM USING CONTROL STRUCTURE.**

INTPUT:-

```
#include<iostream>
void main

{
Int a;
clrscr();
for(a=1;a<=8;a++)
cout<<a<<"\n;
getch():
}
```

OUTPUT:-



```
1
2
3
4
5
6
7
8
_
```

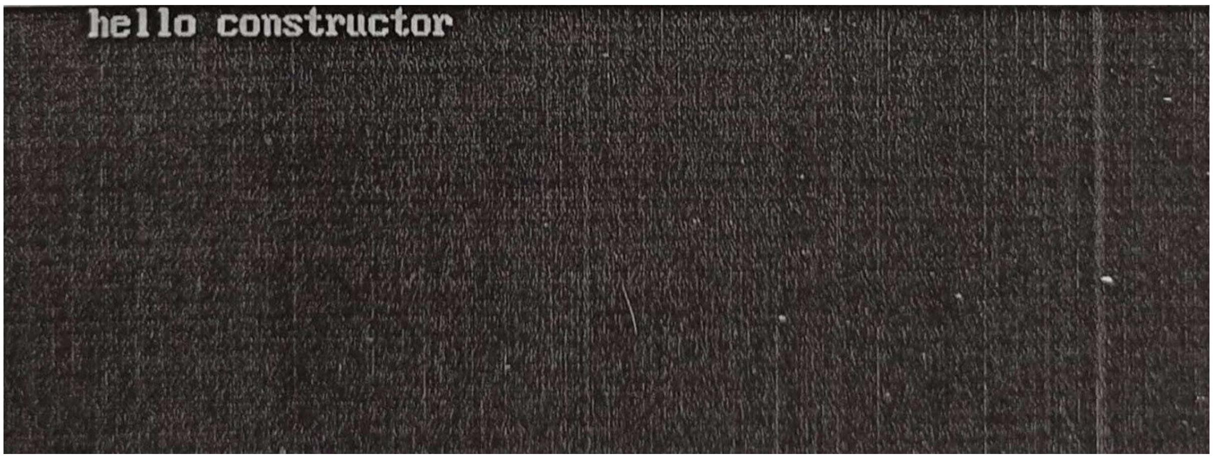
PRACTICAL -: 2(A)

***AIM :-* TO WRITE A PROGRAM USING CONSTRUCTOR.**

INPUT

```
#include<iostream>
class complex
{
Int a,b;
public;
complex;
{
cout<<"hello constructor";
}
};
void main()
{
clrscr();
Complex cl;
getch();
}
```

OUTPUT:-



```
hello constructor
```

PRACTICAL -: 2(B)

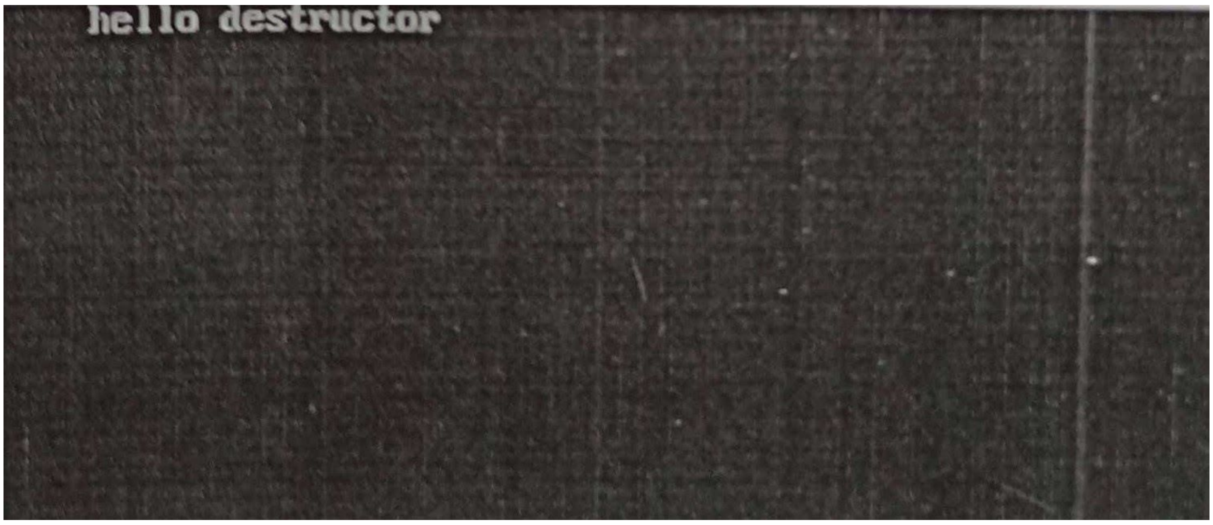
AIM:- TO WRITE A PROGRAM USING DESTRUCTOR.

INPUT:-

```
#include<iostream>
Class integer
{
```

```
Int a,b;
public;
integer()
{
cout<<"hello destructor";
}
};
void main()
{
clrscr;
Integer il;
getch();
}
```

OUTPUT:-



```
hello destructor
```

PRACTICAL -: 3

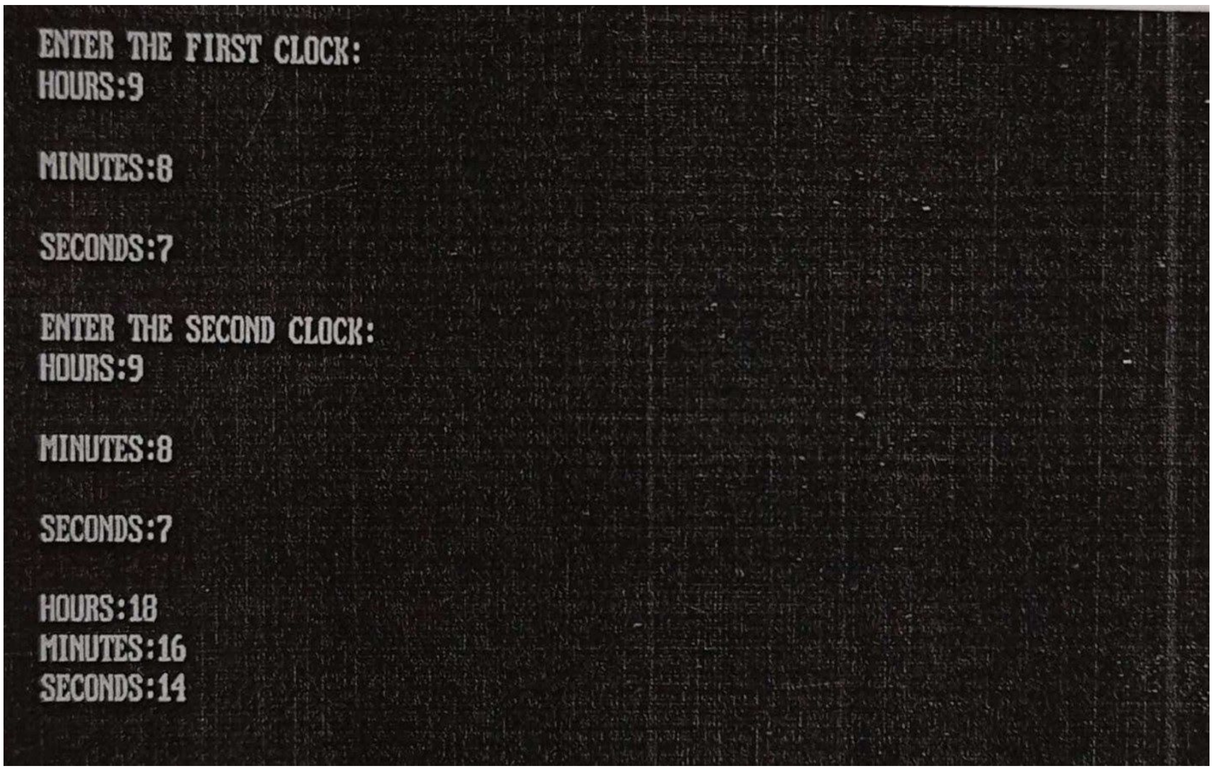
AIM:- USING OBJECT AS A FUNCTION ARGUMENT PERFORM THE ADDITION OF TIME IN HOURS MINUTES AND SECOND FORMAT.

INPUT:-

```
#include<iostream>
```

```
Class time
{
private;
Int h1,m1,s1,h2,m2,s2;
public;
void getdata1()
{
cout<<endl<<”ENTER THE FIRST CLOCK:”<<”HOURS:”;
cin>>h1;
cout<<endl<<”MINUTES:”;
cin>>m1;
cout<<endl<<”SECONDS:”;
cin>>s1;
}
void getdata2(void)
{
cout<<endl<<”ENTER THE SECOND CLOCK:”<<”HOURS:”;
cin>>h2;
cout<<endl<<”MINUTES:”;
cin>>m2;
cout<<endl<<”SECONDS:”;
cin>>s2;
}
void sum(time T1,timeT2)
{
T1.getdata1();
T2.getdata2();
}
void display(void)
{
cout<<endl<<”HOURS:”<<h1+h2<<endl<<”MINUTES:”<<m1+m2<<endl<<”SECON
ONDS:”<<s1+s2;
}
};
void main()
{
clrscr();
Time T;
T.getdata1();
T.getdata2();
T.display();
getch();
}
```

OUTPUT:-



```
ENTER THE FIRST CLOCK:
HOURS:9

MINUTES:8

SECONDS:7

ENTER THE SECOND CLOCK:
HOURS:9

MINUTES:8

SECONDS:7

HOURS:18
MINUTES:16
SECONDS:14
```

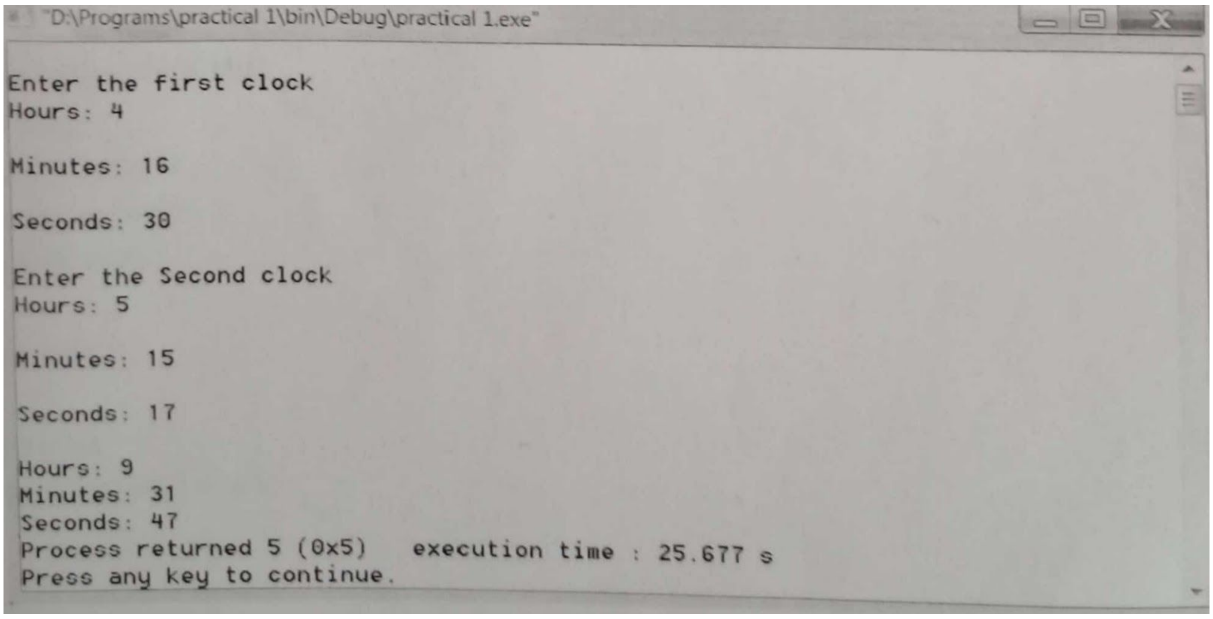
PRACTICAL -: 3(B)

AIM:- Using objects as function arguments perform the addition of time hours,minutes and seconds format.

INPUT:-

```
#include<iostream>

using namespace std;
class time
{
private;
int h1,h2,m1,m2,s1,s2;
public;
void  getdata1()
{
cout<<endl<<”Enter the first clock”<<endl<<”Hours”;
cin>>m1;
cout<<endl<<”Seconds:”;
cin>>s1;
}
void getdata2()
}
cout<<endl<<”Enter the second clock:”<<”Hours:”;
cin>>h2;
cout<<endl<<”Minutes:”;
cin>>m2;
cout<<endl<<”Seconds:”;
cin>>s2;
}
void sum(time 11,timet2)
{
t1.getdata1();
t2.getdata2();
}
void display(void)
{
coût<<"\nHours:"<<h1+h2<<"\nMinutes:";
cout<<m1+m2<<endl<<”Seconds:”<<s1+s2;
}
};
int main()
{
time t;
t.getdata1();
t.getdata2();
t.display();
return 5;
}
```

OUTPUT:-



"D:\Programs\practical 1\bin\Debug\practical 1.exe"

```
Enter the first clock
Hours: 4

Minutes: 16

Seconds: 30

Enter the Second clock
Hours: 5

Minutes: 15

Seconds: 17

Hours: 9
Minutes: 31
Seconds: 47
Process returned 5 (0x5)   execution time : 25.677 s
Press any key to continue.
```

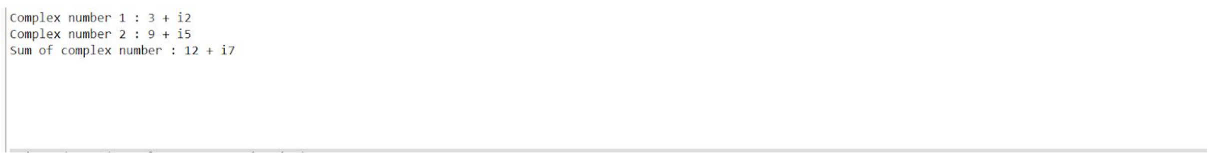
PRACTICAL -:4

AIM:- Perform addition of tom complex numbers using classes.

INPUT:-

```
#include<iostream>
using namespace std;


class Complex {



   public:
      int real, imaginary;


   Complex()
   {
   }


   Complex(int tempReal, int tempImaginary)
   {
        real = tempReal;
        imaginary = tempImaginary;
    }


    Complex addComp(Complex C1, Complex C2)
    {

        Complex temp;


        temp.real = C1.real + C2.real;


        temp.imaginary = C1.imaginary + C2.imaginary;


        return temp;
    }
};


int main()
{


    Complex C1(3, 2);


    cout<<"Complex number 1 : "<< C1.real
            << " + i"<< C1.imaginary<<endl;



    Complex C2(9, 5);


    cout<<"Complex number 2 : "<< C2.real
            << " + i"<< C2.imaginary<<endl;


    Complex C3;
    C3 = C3.addComp(C1, C2);

        cout<<"Sum of complex number : "
                        << C3.real << " + i"
                        << C3.imaginary;
}
```

OUTPUT:-



```
Complex number 1 : 3 + i2
Complex number 2 : 9 + i5
Sum of complex number : 12 + i7
```

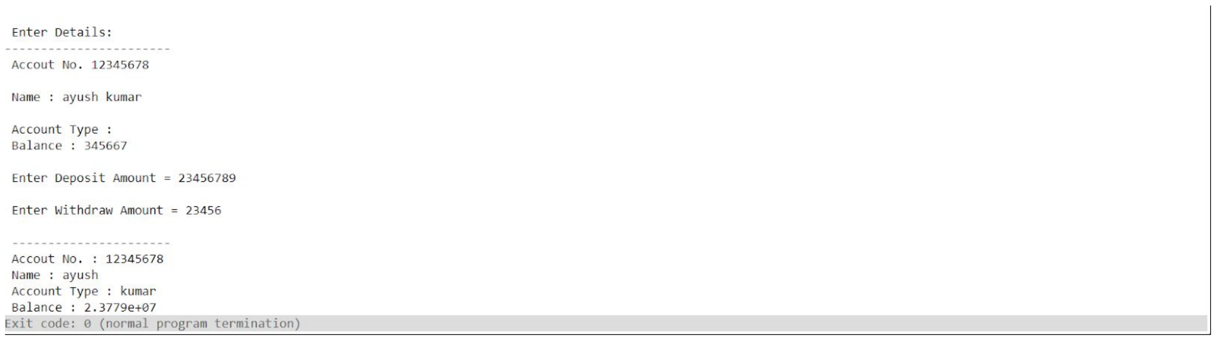
PRACTICAL -: 5

AIM:- DEFINE A CLASS TO REPRESENT BANK ACCOUNT INCLUDE THE FOLLOWING DATA MEMBER: NAME OF THE DEPOSITOR,ACCOUNT NUMBER,TYPE OF ACCOUNT,AND BALANCE ACCOUNT IN THE ACCOUNT,MEMBER FUNCTIONS:TO ASSIGN INITIAL VALUE,TO DEPOSIT AN AMOUNT,TO WITHDRAW AN AMOUNT AFTER CHECKING THE BALANCE ,TO DISPLAY NAME AND BALANCES ,WRITE A MAIN PROGRAM TO TEST THE PROGRAM .

INPUT:-

```
#include<iostream>
#include<stdio.h>
#include<string.h>

using namespace std;

class bank
{
        int acno;
        char nm[100], acctype[100];
        float bal;
   public:
        bank(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::deposit()   //depositing an amount
{
        int damt1;
        cout<<"\n Enter Deposit Amount = ";
        cin>>damt1;
        bal+=damt1;
}
void bank::withdraw()  //withdrawing an amount
{
        int wamt1;
        cout<<"\n Enter Withdraw Amount = ";
        cin>>wamt1;
        if(wamt1>bal)
                cout<<"\n Cannot Withdraw Amount";
        bal-=wamt1;
}
void bank::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 b1(acc_no, name, acc_type, balance);  //object is created
        b1.deposit();
        b1.withdraw();
        b1.display();
        return 0;
}
```

OUTPUT:-



```
 Enter Details:
-----------------------
 Accout No. 12345678

 Name : ayush kumar

 Account Type :
 Balance : 345667

 Enter Deposit Amount = 23456789

 Enter Withdraw Amount = 23456

 ----------------------
 Accout No. : 12345678
 Name : ayush
 Account Type : kumar
 Balance : 2.3779e+07
Exit code: 0 (normal program termination)
```

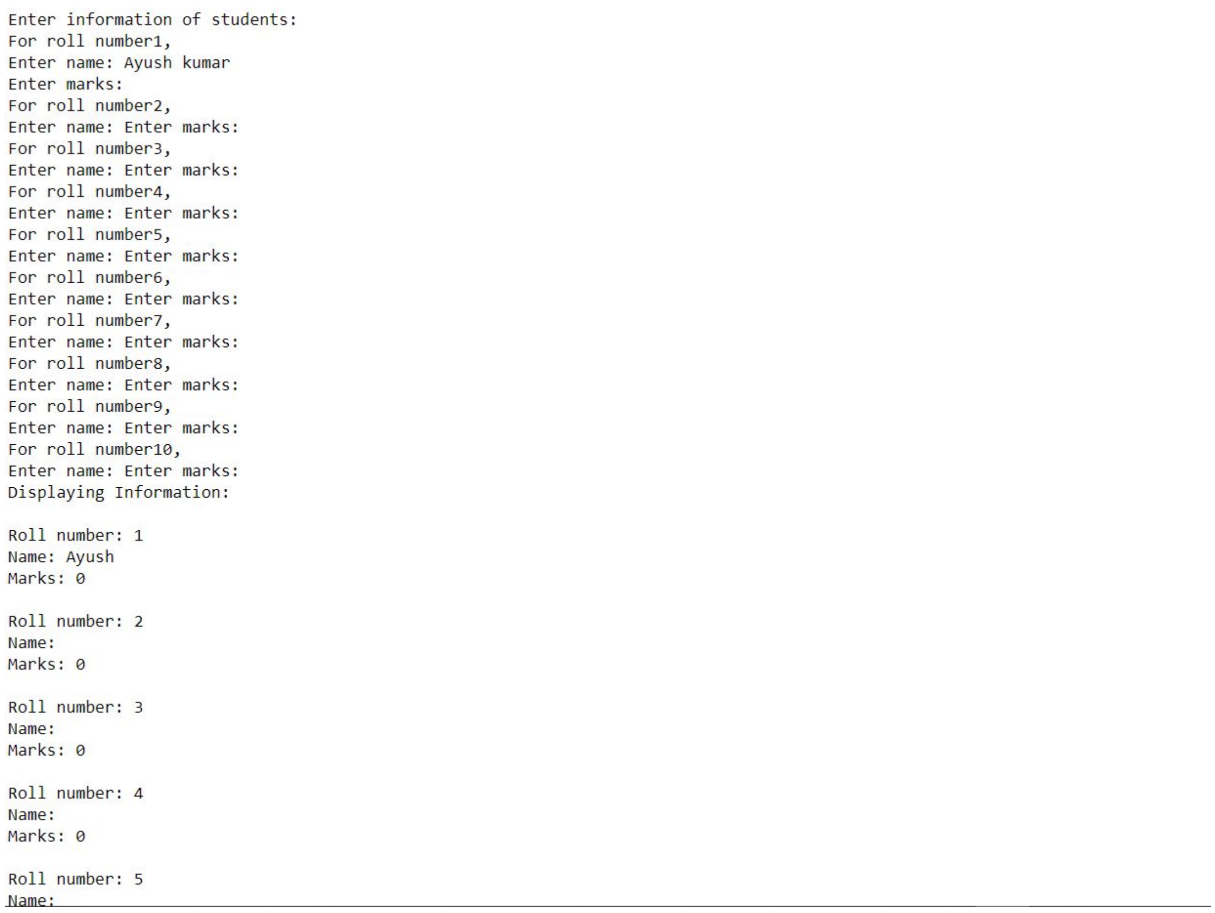
<u>PRACTICAL -: 6</u>

AIM:-MODIFY THE PROGRAM FOR HANDLING 10 CUSTOMER USING ARRAY OF OBJECT.

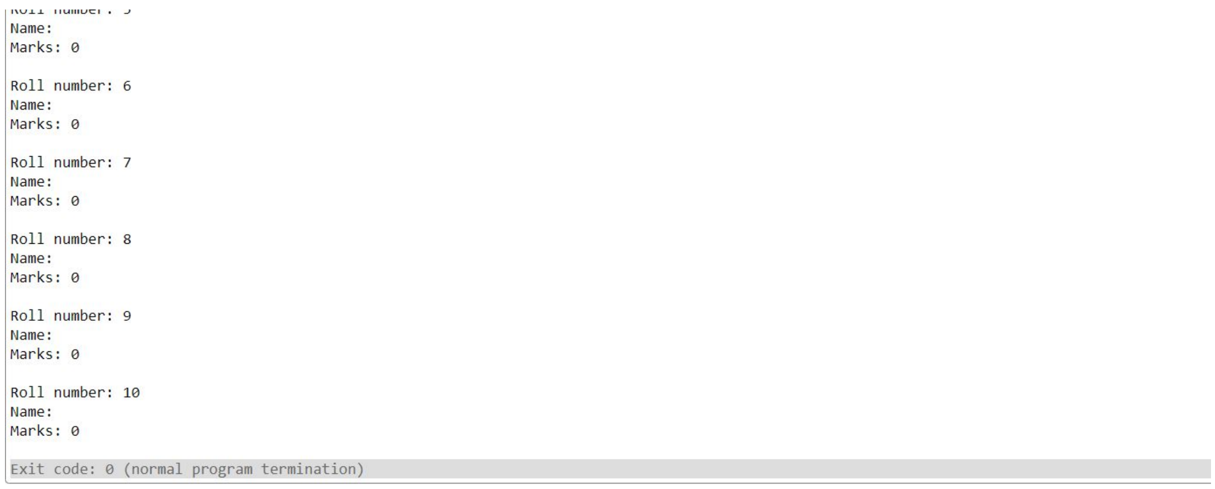
INPUT:-

```
#include <iostream>
using namespace std;

struct student
{
    char name[50];
    int roll;
    float marks;
} s[10];

int main()
{
    cout << "Enter information of students: " << endl;

    // storing information
    for(int i = 0; i < 10; ++i)
    {
        s[i].roll = i+1;
        cout << "For roll number" << s[i].roll << "," << endl;

        cout << "Enter name: ";
        cin >> s[i].name;

        cout << "Enter marks: ";
        cin >> s[i].marks;

        cout << endl;
    }

    cout << "Displaying Information: " << endl;

    // Displaying information
    for(int i = 0; i < 10; ++i)
    {
        cout << "\nRoll number: " << i+1 << endl;
        cout << "Name: " << s[i].name << endl;
        cout << "Marks: " << s[i].marks << endl;
    }

    return 0;
}
```

OUTPUT:-



```
Enter information of students:
For roll number1,
Enter name: Ayush kumar
Enter marks:
For roll number2,
Enter name: Enter marks:
For roll number3,
Enter name: Enter marks:
For roll number4,
Enter name: Enter marks:
For roll number5,
Enter name: Enter marks:
For roll number6,
Enter name: Enter marks:
For roll number7,
Enter name: Enter marks:
For roll number8,
Enter name: Enter marks:
For roll number9,
Enter name: Enter marks:
For roll number10,
Enter name: Enter marks:
Displaying Information:

Roll number: 1
Name: Ayush
Marks: 0

Roll number: 2
Name:
Marks: 0

Roll number: 3
Name:
Marks: 0

Roll number: 4
Name:
Marks: 0

Roll number: 5
Name:
```



```
Name:
Marks: 0

Roll number: 6
Name:
Marks: 0

Roll number: 7
Name:
Marks: 0

Roll number: 8
Name:
Marks: 0

Roll number: 9
Name:
Marks: 0

Roll number: 10
Name:
Marks: 0

Exit code: 0 (normal program termination)
```

PRACTICAL -: 7

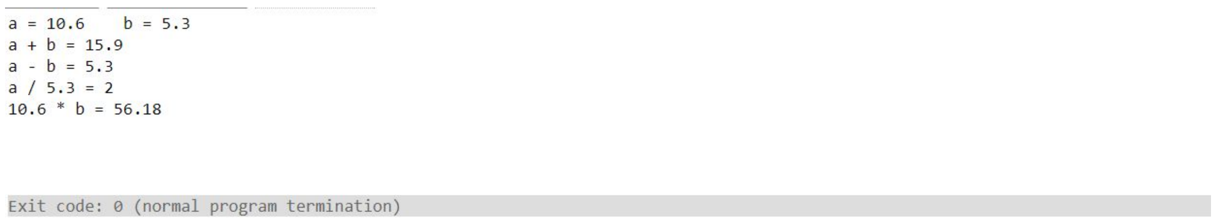
AIM:-Create a class float that contains one float data member overload all the four arithmetic operation so that operate on the object of the folat.

INPUT:-

```
#include<iostream>
using namespace std;
class Float
{
   float i;
   public:
      Float(): i(5) {}
      Float(float x): i(x) {}
      Float operator + (Float a)
      {
         Float temp;
         temp.i = i + a.i;
         return temp;
      }

      Float operator - (Float a)
      {
         Float temp;
         temp.i = i - a.i;
         return temp;
      }

      Float operator / (float a)
      {
         Float temp;
         temp.i = i / a;
         return temp
}

      friend Float operator* (float a,Float b)
      {
         Float temp;
         temp.i = a * b.i;
         return temp;
      }

      void show()
      {
         cout<<i<<endl;
      }
};
```

```
int main()
{
    Float a = 10.6,b = 5.3,c;
    cout<<"a = 10.6    b = 5.3\n";
    c = a + b;
    cout<<"a + b = ";c.show();
    c = a - b;
    cout<<"a - b = ";c.show();
    c = a / 5.3;
    cout<<"a / 5.3 = ";c.show();
    c = 10.6 * b;
    cout<<"10.6 * b = ";c.show();
    return 0;
}
```

OUTPUT:-



```
a = 10.6     b = 5.3
a + b = 15.9
a - b = 5.3
a / 5.3 = 2
10.6 * b = 56.18


Exit code: 0 (normal program termination)
```

PRACTICAL -: 8

AIM:- Define a class string use of overloaded ==operator to compare two strings .

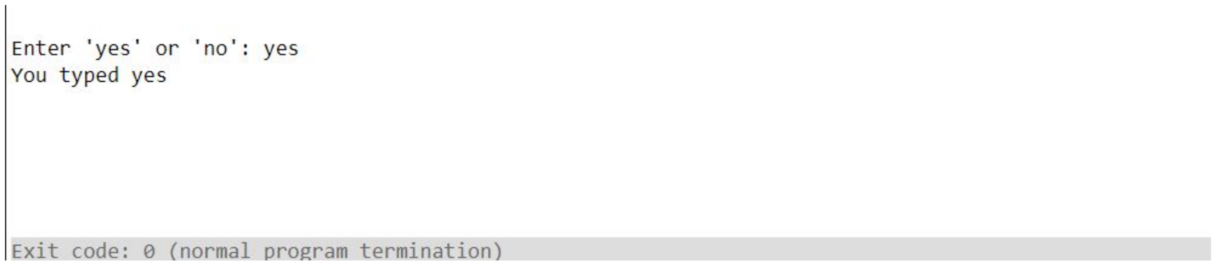
INPUT:-

```
#include <iostream>

using namespace std;

#include <string.h>

class String{

private:
```

```
enum { SZ = 80 };

char str[SZ];

public:

String(){ strcpy(str, ""); }

String( char s[] ){ strcpy(str, s); }

void display() const{ cout << str; }

void getstr(){ cin.get(str, SZ); }

bool operator == (String ss) const{

return ( strcmp(str, ss.str)==0 ) ? true : false;

}

};

int main(){

String s1 = "yes";

String s2 = "no";

String s3;



cout << "\nEnter 'yes' or 'no': ";

s3.getstr();
```

```
if(s3==s1)

cout << "You typed yes\n";

else if(s3==s2)

cout << "You typed no\n";

else

cout << "You didn't follow instructions\n";

return 0;

}
```

OUTPUT:-



```
Enter 'yes' or 'no': yes
You typed yes




Exit code: 0 (normal program termination)
```

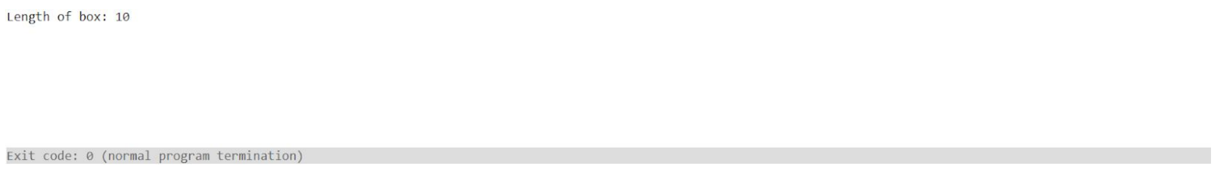
PRACTICAL -: 9

AIM:-_Write a program using friend function

INPUT:-

```
#include <iostream>
using namespace std;
class Box
{
private:
 int length;
public:
Box(): length(0) { }
 friend int printLength(Box); //friend function
};
int printLength(Box b)
{
b.length += 10;
return b.length;
}
int main()
{
Box b;
cout<<"Length of box: "<< printLength(b)<<endl;
return 0;
}
```

OUTPUT:-



```
Length of box: 10




Exit code: 0 (normal program termination)
```

PRACTICAL -: 10

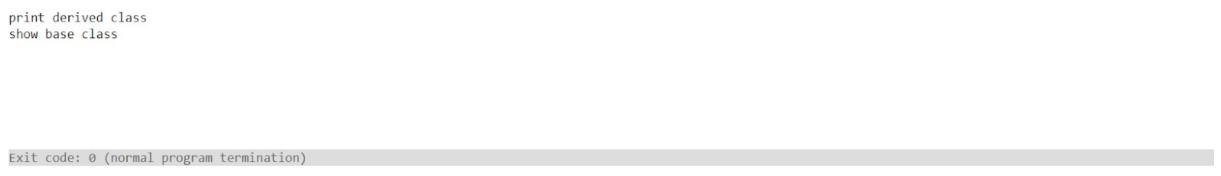
AIM:-Write a program using virtual function?

INPUT:-

```
#include <iostream>

using namespace std;

class base {
```

```
public:

    virtual void print()

    {

        cout << "print base class" << endl;

    }



    void show()

    {

        cout << "show base class" << endl;

    }

};



class derived : public base {

public:

    void print()

    {

        cout << "print derived class" << endl;

    }



    void show()
    {
        cout << "show derived class" << endl;
    }
};

int main()
{
    base* bptr;
    derived d;
    bptr = &d;
  // virtual function, binded at runtime
    bptr->print();
  // Non-virtual function, binded at compile time
    bptr->show();
}
```

OUTPUT:-



```
print derived class
show base class



Exit code: 0 (normal program termination)
```