“Heaven’s Light is Our Guide”

Department of Computer Science & Engineering

RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY

Lab Report-07

Submitted By:

Name: Khandoker Sefayet Alam
Roll:2003121
Department: Computer Science & Engineering
Section-C
Session:2020-21
Course code: CSE 1204

Submitted to:

SUHRID SHAKHAR GHOSH
Assistant Professor
Department of Computer Science & Engineering, RUET

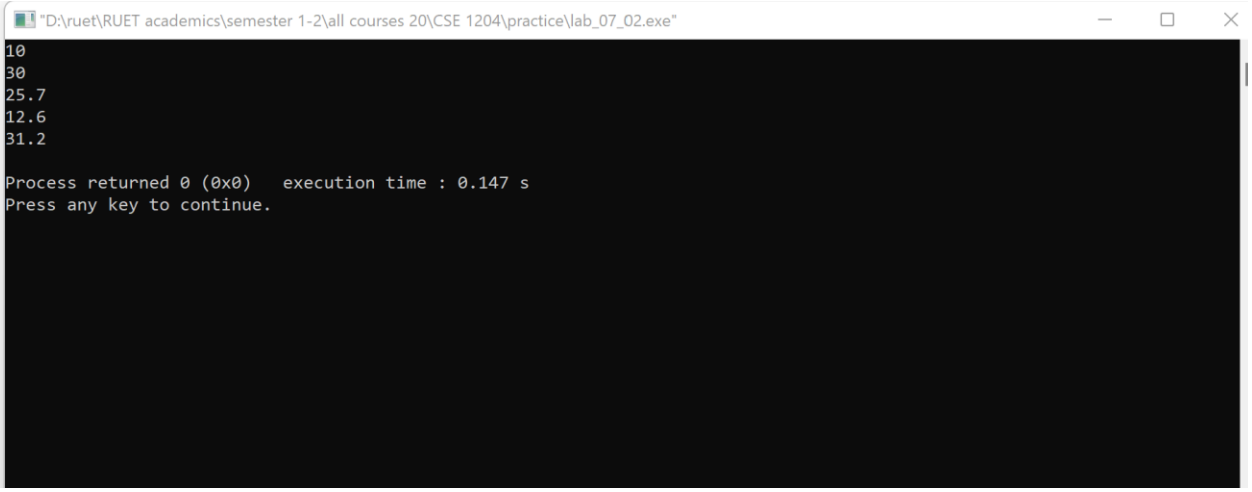
Module 4 [polymorphism]: (for Week 5)
Topic 1[**Method/Function Overloading**]
Problem Statement: Define a class Test where overload a method Sum() to sum numbers sent from main() function.

```
class Test{
public:
//overload Sum() method according to the requirement of main()
};
int main(){
Test t;
t.Sum(10); //returns 10
t.Sum(10,20) //return 30
t.Sum(5.7,20) //return 25.7
t.Sum(10,2.6) //return 12.6
t.Sum(10.5,20.7) //return 21.2
}
```

Solution:

```
#include<iostream>

using namespace std;

class Test
{
public:
   int sum(int a)
   {
      return a;
   }
   int sum(int a,int b)
   {
      return a+b;
   }
   double sum(double a,double b){
   return a+b;
   }
    double sum(double a,int b){
   return a+b;
   }
    double sum(int a,double b){
   return a+b;
   }
};
int main()
{
   Test t;
   cout<<t.sum(10)<<endl;; //returns 10
   cout<<t.sum(10,20)<<endl; //return 30
   cout<<t.sum(5.7,20)<<endl; //return 25.7
   cout<<t.sum(10,2.6)<<endl; //return 12.6
   cout<<t.sum(10.5,20.7)<<endl; //return 21.2
}
```

OUTPUT:


"D:\ruet\RUET academics\semester 1-2\all courses 20\CSE 1204\practice\lab_07_02.exe"

```
10
30
25.7
12.6
31.2

Process returned 0 (0x0)   execution time : 0.147 s
Press any key to continue.
```

Topic 2 [**Operator Overloading**]: Suppose in a AC circuit, there are 3 impedances z1=3+j4, z2=4-j3 and z3=j6 are connected in parallel. Now find the current in the circuit if input voltage is 100+j50. Implement **operator overloading** concept for your calculation. Use class **Circuit** and initialize the impedance values (real & img) by a constructor.

```
class Circuit{
private:
int real;
int img;
public:
//write constructor
//write operator overloaded method
//write a display method to display real and img
};
int main(){
Circuit z1(3,4);
//write required statements to find the current
}
```

Solution:

```
#include<iostream>
```

```
using namespace std;

class Complex{
    float real;
    float imaginary;
public:
    Complex(){
    }
    Complex(float r,float i){
    real=r;
    imaginary=i;
    }

    Complex operator+(Complex x){
    Complex temp;
    temp.real=real+x.real;
    temp.imaginary=imaginary+x.imaginary;
    return temp;
    }
    Complex operator-(Complex x){
    Complex temp;
    temp.real=real-x.real;
    temp.imaginary=imaginary-x.imaginary;
    return temp;
    }
    Complex operator*(Complex x){
    Complex temp;
```

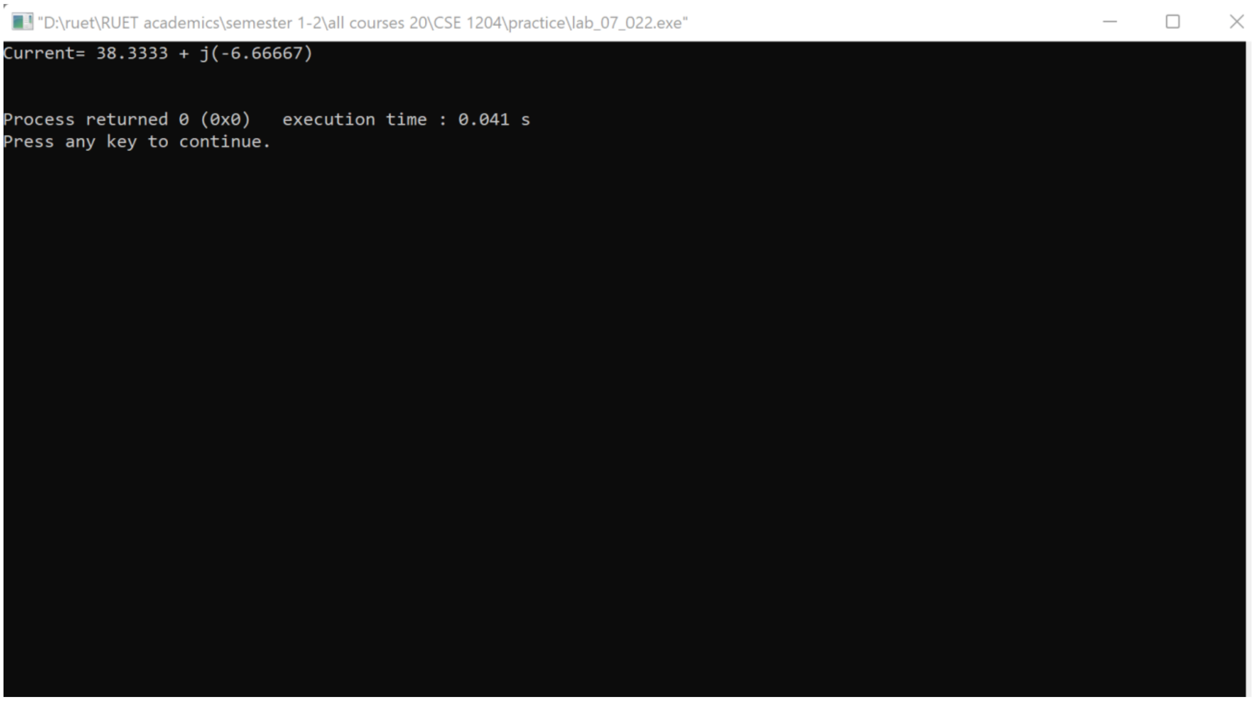
```
temp.real=real*(x.real)-(imaginary)*(x.imaginary);
temp.imaginary=(real)*(x.imaginary)+(imaginary)*(x.real);
return temp;
}
Complex operator/(Complex x){
Complex temp;
float g=(x.real)*(x.real)+(x.imaginary)*(x.imaginary);
temp.real=(real*(x.real)+(imaginary)*(x.imaginary))/g;
temp.imaginary=((x.real)*(imaginary)-(x.imaginary)*(real))/g;
return temp;
}
void print(){
cout<<real<<" + i"<<imaginary<<endl;
}
Complex operator+(float x){
Complex temp;
temp.real=real+x;
temp.imaginary=imaginary;
return temp;
}
Complex operator*(float x){
Complex temp;
temp.real=real*x;
temp.imaginary=imaginary*x;
return temp;
}
 Complex operator/(float x){
```

```
Complex temp;

temp.real=real/x;

temp.imaginary=imaginary/x;

return temp;

}

///postfix increment

Complex operator++(int){

Complex temp=*this;

real++;

return temp;

}

///prefix increment

void operator++(){

real++;

}


 ///postfix decrement

Complex operator--(int){

Complex temp=*this;

real--;

return temp;

}

///prefix decrement

void operator--(){

real--;

}
```

```
    friend Complex operator+(float x,Complex y);
     friend Complex operator-(float x,Complex y);
      friend Complex operator*(float x,Complex y);
       friend Complex operator/(float x,Complex y);

     friend istream& operator>>(istream &in,Complex &a);
       friend ostream& operator<<(ostream &ou,Complex &b);



};

Complex operator+(float x,Complex y){
    Complex temp;
    temp.real=y.real+x;
    temp.imaginary=y.imaginary;
    return temp;
}



Complex operator-(float x,Complex y){
    Complex temp;
    temp.real=y.real-x;
    temp.imaginary=y.imaginary;
    return temp;
}

Complex operator*(float x,Complex y){
    Complex temp;
    temp.real=(y.real)*x;
    temp.imaginary=-(y.imaginary)*x;
    return temp;
}


Complex operator/(float x,Complex y){
    Complex temp;
     float g=(y.real)*(y.real)+(y.imaginary)*(y.imaginary);
    temp.real=(y.real)*x/g;
    temp.imaginary=-(y.imaginary)*x/g;
    return temp;
}

istream& operator>>(istream &in,Complex &a){
    cout<<"Enter the real part: ";
    in>>a.real;
    cout<<"Enter the complex part: ";
    in>>a.imaginary;
    return in;
}

ostream& operator<<(ostream &ou,Complex &b){
    ou<<b.real<<" + j("<<b.imaginary<<") "<<endl;
    return ou;
}
```

```
int main(){


  Complex z1(3,4),z2(4,-3),z3(0,6),V(100,50);
  Complex var=1/z1+1/z2+1/z3;
  Complex Req=1/var;
  Complex I=V/Req;
  cout<<"Current= "<<I<<endl;
}
```

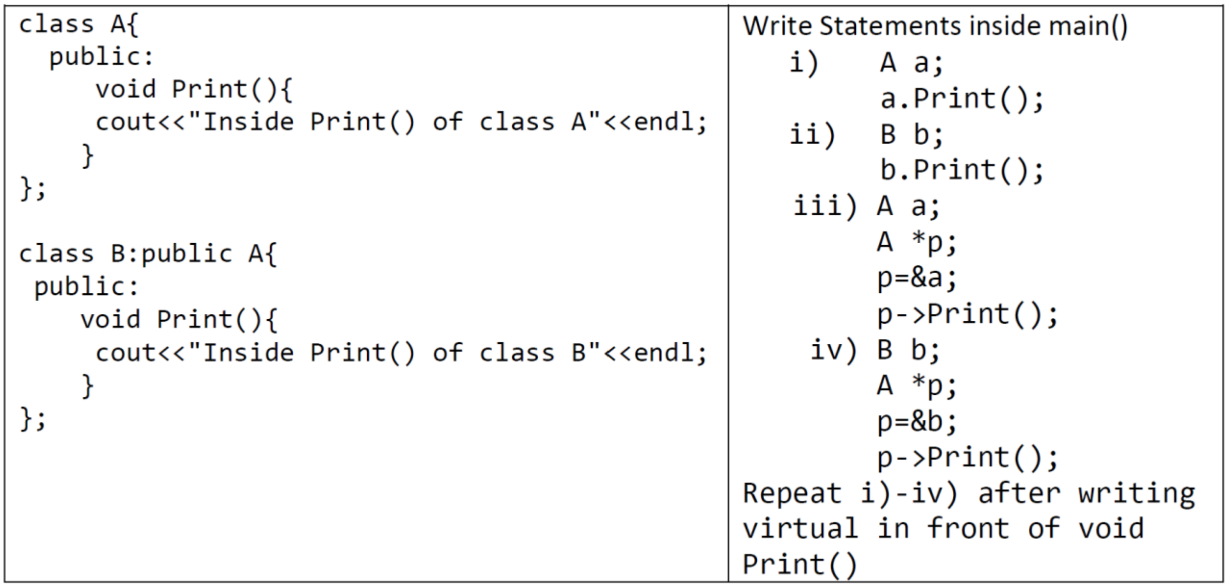
OUTPUT:


"D:\ruet\RUET academics\semester 1-2\all courses 20\CSE 1204\practice\lab_07_022.exe"

```
Current= 38.3333 + j(-6.66667)


Process returned 0 (0x0)   execution time : 0.041 s
Press any key to continue.
```

Topic 3[Method/Function Overriding]

Problem statement: Write a class A with a method **Print()** and a derived class B with method **Print()** overloaded. Now observe the output when following statements are written in the **main()** function



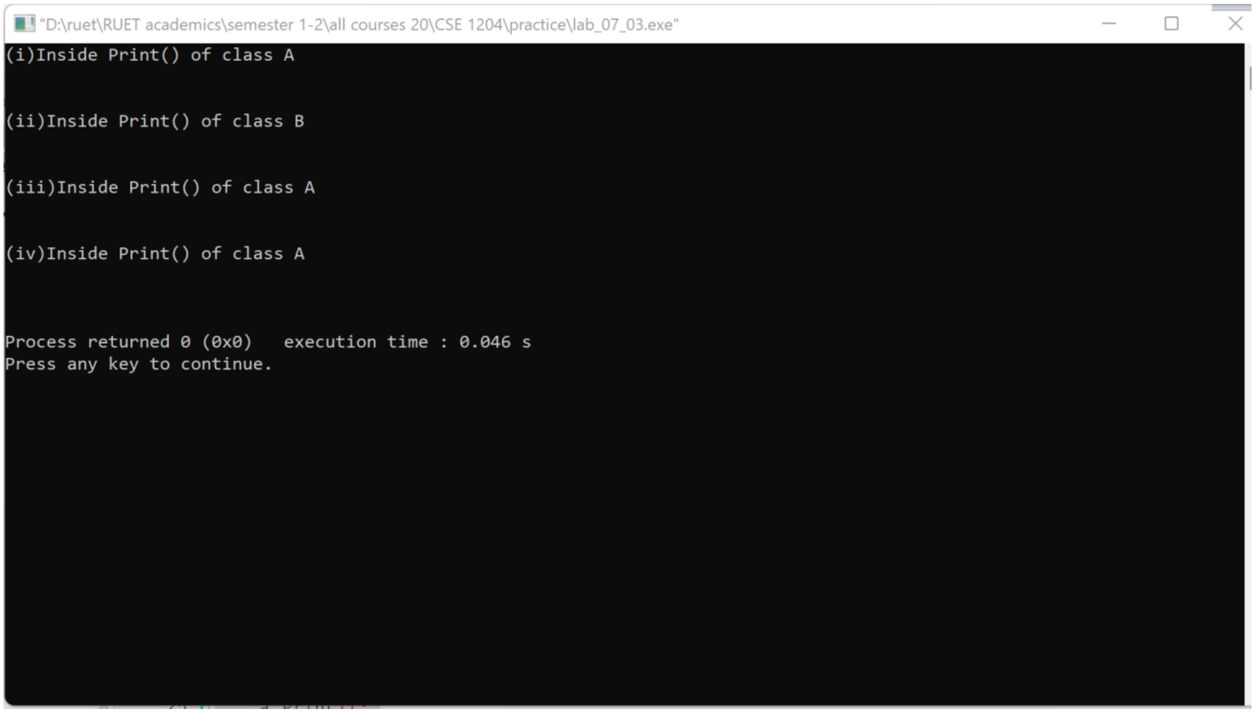
class A{
  public:
     void Print(){
     cout<<"Inside Print() of class A"<<endl;
    }
};
 	
class B:public A{
 public:
    void Print(){
     cout<<"Inside Print() of class B"<<endl;
    }
};
	Write Statements inside main()
	i) A a;
	     a.Print();
	ii) B b;
	     b.Print();
	iii) A a;
	     A *p;
	     p=&a;
	     p->Print();
	iv) B b;
	     A *p;
	     p=&b;
	     p->Print();
	Repeat i)-iv) after writing virtual in front of void Print()

Solution:

```
#include<iostream>


using namespace std;


class A

{

public:

    void Print()

    {

        cout<<"Inside Print() of class A"<<endl;

    }

};
```

```
class B:public A
{
public:
    void Print()
    {
        cout<<"Inside Print() of class B"<<endl;
    }
};

int main(){
    cout<<"(i)";
    A a;
    a.Print();
    cout<<endl<<endl;
     cout<<"(ii)";
    B b;
    b.Print();
     cout<<endl<<endl;
      cout<<"(iii)";
    A *p;
    p=&a;
    p->Print();
     cout<<endl<<endl;
      cout<<"(iv)";
    p=&b;
    p->Print();
     cout<<endl<<endl;
}
```

OUTPUT:

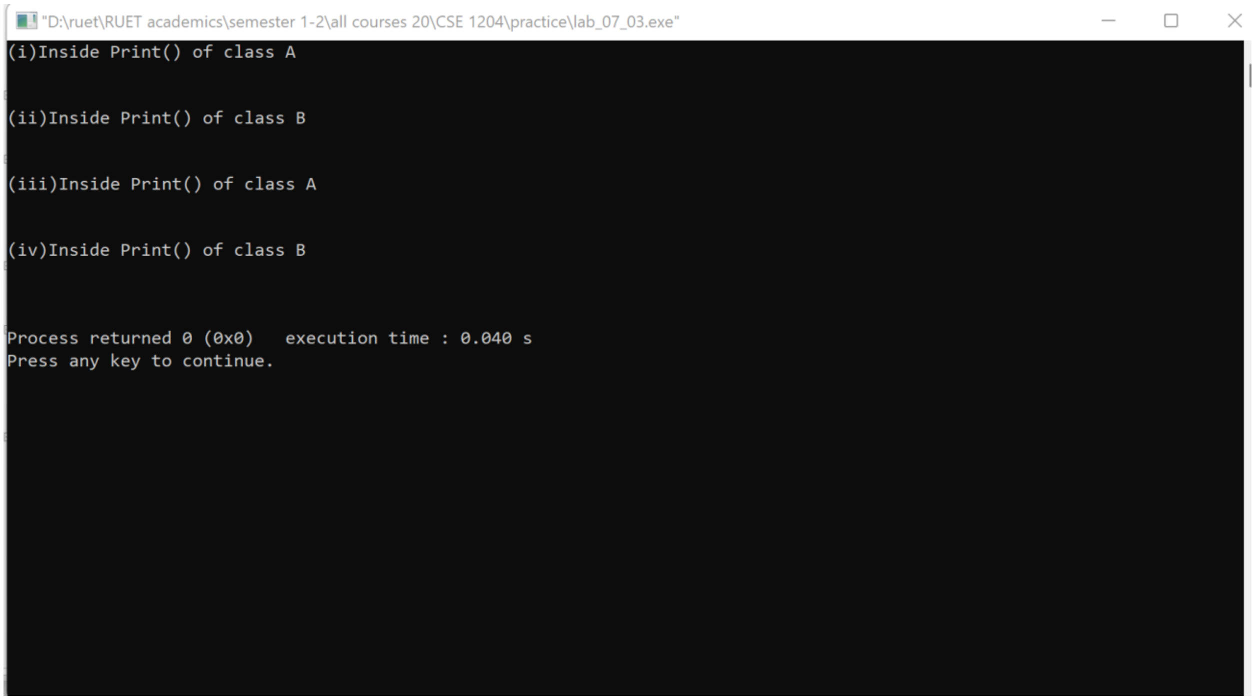

"D:\ruet\RUET academics\semester 1-2\all courses 20\CSE 1204\practice\lab_07_03.exe"

```
(i)Inside Print() of class A


(ii)Inside Print() of class B


(iii)Inside Print() of class A


(iv)Inside Print() of class A



Process returned 0 (0x0)   execution time : 0.046 s
Press any key to continue.
```

Using virtual keyword:

```
#include<iostream>


using namespace std;


class A
{
public:
    virtual void Print()
    {
        cout<<"Inside Print() of class A"<<endl;
    }
};
class B:public A
{
public:
    void Print()
    {
        cout<<"Inside Print() of class B"<<endl;
    }
};

int main(){
    cout<<"(i)";
    A a;
    a.Print();
    cout<<endl<<endl;
     cout<<"(ii)";
    B b;
    b.Print();
     cout<<endl<<endl;
      cout<<"(iii)";
    A *p;
    p=&a;
    p->Print();
     cout<<endl<<endl;
      cout<<"(iv)";
    p=&b;
    p->Print();
     cout<<endl<<endl;
}
```

OUTPUT:

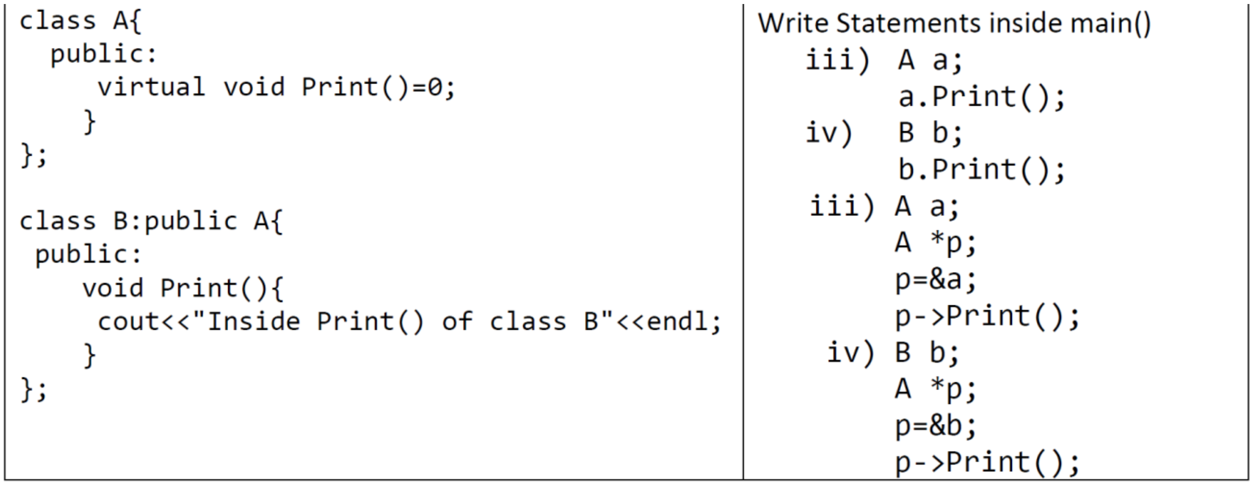


"D:\ruet\RUET academics\semester 1-2\all courses 20\CSE 1204\practice\lab_07_03.exe"

```
(i)Inside Print() of class A


(ii)Inside Print() of class B


(iii)Inside Print() of class A


(iv)Inside Print() of class B



Process returned 0 (0x0)   execution time : 0.040 s
Press any key to continue.
```

Topic 4[Pure Virtual Function]

Problem statement: Modify the class defined in Topic 3 executes the following statements i)-iv) and observe the output:



```
class A{
  public:
     virtual void Print()=0;
    }
};

class B:public A{
 public:
    void Print(){
     cout<<"Inside Print() of class B"<<endl;
    }
};
```

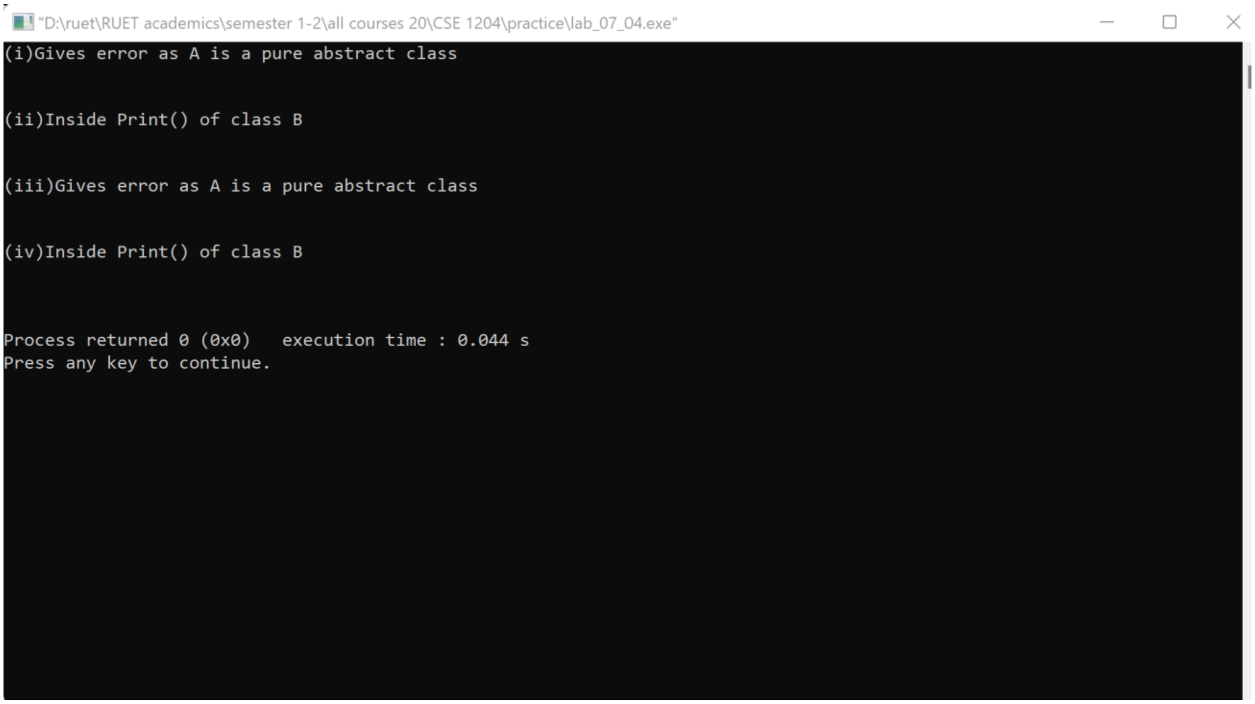
Write Statements inside main()

```
   iii)  A a;
         a.Print();
   iv)   B b;
         b.Print();
   iii) A a;
        A *p;
        p=&a;
        p->Print();
    iv) B b;
        A *p;
        p=&b;
        p->Print();
```

Solution:

```
#include<iostream>


using namespace std;


class A

{

public:

    virtual void Print()=0;


};

class B:public A

{

public:
    void Print()
    {
        cout<<"Inside Print() of class B"<<endl;
    }
};
int main()
{
    cout<<"(i)"<<"Gives error as A is a pure abstract class"<<endl;
    //A a;
    //a.Print();
    cout<<endl<<endl;
    cout<<"(ii)";
    B b;
    b.Print();
    cout<<endl<<endl;
    cout<<"(iii)"<<"Gives error as A is a pure abstract class"<<endl;
    A *p;
    //p=&a;
    //p->Print();
    cout<<endl<<endl;
    cout<<"(iv)";
    p=&b;
    p->Print();
    cout<<endl<<endl;
}
```

OUTPUT:



"D:\ruet\RUET academics\semester 1-2\all courses 20\CSE 1204\practice\lab_07_04.exe"

```
(i)Gives error as A is a pure abstract class


(ii)Inside Print() of class B


(iii)Gives error as A is a pure abstract class


(iv)Inside Print() of class B



Process returned 0 (0x0)   execution time : 0.044 s
Press any key to continue.
```

Topic 5 [Friend Function]

Problem Statement: Using the following class, write three friend functions

i) Add(): Assign value to the data member x

ii) IncX() : Increase the value of x by m

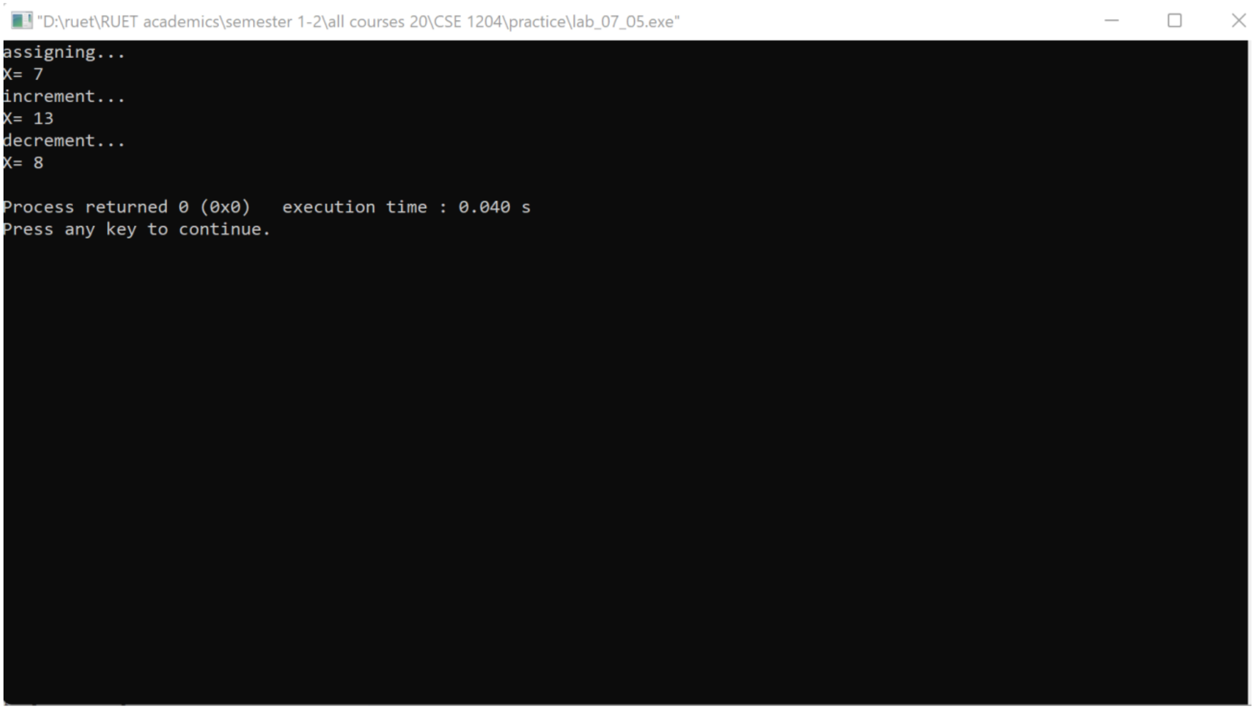
iii) DecX() : Decrease the value of x by n

```
class A{
 private:
    int x;
  public:
     //Prototype of friend functions
};

//write body of friend functions

int main(){
 //call these methods
}
```

Solution:

```
#include<iostream>

using namespace std;


class A

{

private:

    int x;

    public:
        friend void Add(int m,A &a);
        friend void IncX(int m,A &a);
        friend void DecX(int m,A &a);

         void Print(){
            cout<<"X= "<<x<<endl;
        }
};
void Add(int m,A &a)
{
    a.x=m;

}
void IncX(int m,A &a)
{
    a.x+=m;

}
void DecX(int m,A &a)
{
     a.x-=m;

}
int main()
{
    A a;
    cout<<"assigning..."<<endl;
    Add(7,a);
    a.Print();
     cout<<"increment..."<<endl;
    IncX(6,a);
    a.Print();
     cout<<"decrement..."<<endl;
    DecX(5,a);
    a.Print();

}
```

OUTPUT:


"D:\ruet\RUET academics\semester 1-2\all courses 20\CSE 1204\practice\lab_07_05.exe"

```
assigning...
X= 7
increment...
X= 13
decrement...
X= 8

Process returned 0 (0x0)   execution time : 0.040 s
Press any key to continue.
```