Dr B R Ambedkar National Institute of Technology

Grand Trunk Road, Barnala - Amritsar Bypass Road,
Jalandhar, Punjab 144011

ITPC-221 OBJECT-ORIENTED PROGRAMMING CONCEPT LAB

B. TECH IT – II YEAR (3rd SEMESTER)

Submitted by
Ajay Kumar Yadav
21124007

Submitted to
Dr Vanitha P S
Department of Information Technology

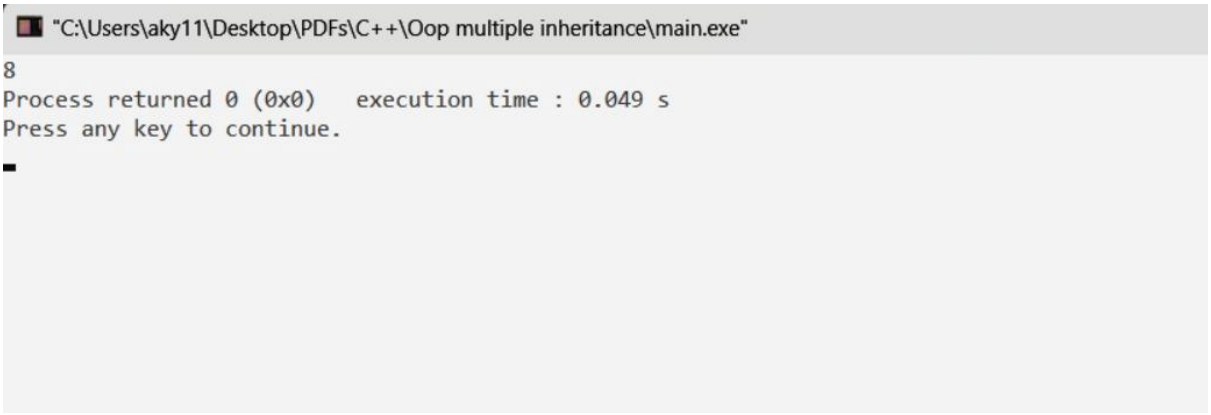
QUESTION 9: Program to show multiple inheritance.

Multiple inheritance

EX.NO.: 9
DATE :10/10/2022
AIM: Implementing multiple inheritance

```
#include <iostream>

using namespace std;

class base{
int a;
public:
   void setdata (int x){
   a=x;
   }
   int getdata(){
   return a;}
};
class base2{
int a2;
public:
   void setdata2(int x){
   a2=x;
   }
   int getdata2(){
   return a2;}
};

class derived : public base, public base2{
int c;
public:
   int getdata3(){
   c = getdata() + getdata2();
   cout << c;
   }
};

int main()
{ derived d ;
d.setdata(3);
d.setdata2(5);
d.getdata3();
  return 0;
}
```

OUTPUT :



RESULT: code executed successfully.

QUESTION 10 : Program to show multilevel inheritance.

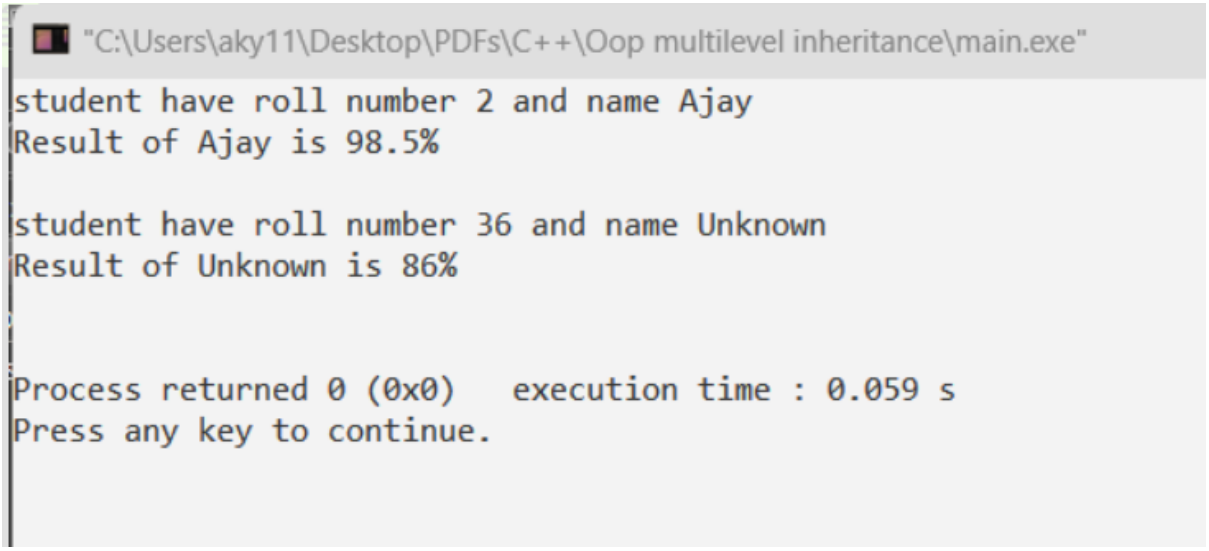
Multilevel inheritance

EX.NO.: 10
DATE :10/10/2022
AIM: implementing multiple inheritance CODE

```
#include <iostream>

using namespace std;

class student{
protected:
   int roll;
public:
   string name;
   void setdata(int a, string n){
   roll=a;
   name=n;
   }
   void display(){
   cout << "student have roll number "<< roll<< " and name " <<name <<endl;
   }
};

class exam: public student{
protected:
int maths, physics;
public:
   void setmarks( int m, int p){
   maths=m;
   physics=p;
   }
   int getm(){
   return maths;
   }
   int getp(){
   return physics;
   }
};

class result: public exam{
public:
   float getresult(){
   cout << "Result of " <<name<< " is "<< ((float)getm()+(float)getp())/2<< "%"
<<endl<<endl;
   }
};

int main(){
result Ajay, unknown;
Ajay.setdata(2,"Ajay");
```

```
Ajay.display();
Ajay.setmarks(98,99);
Ajay.getresult();

unknown.setdata(36,"Unknown");
unknown.display();
unknown.setmarks(77,95);
unknown.getresult();
   return 0;
}
```

Output:



RESULT: code executed successfully.

QUESTION 11: Program to show hybrid inheritance.

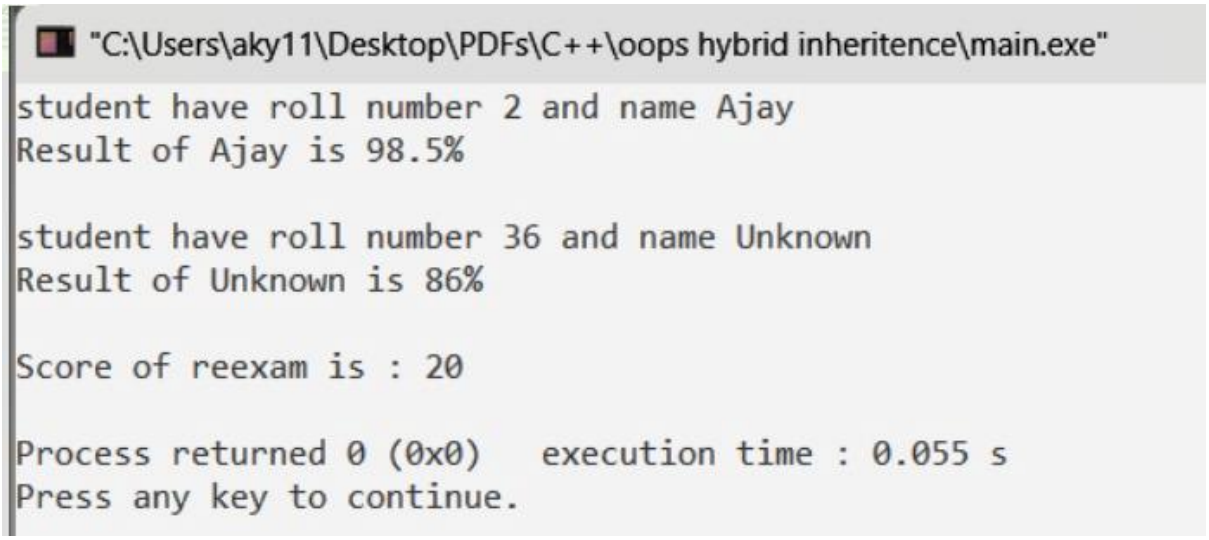
Hybrid inheritance

EX.NO.: 11
DATE :10/10/2022
AIM: implementing Hybrid inheritance CODE

```
#include <iostream>

using namespace std;

class student{
protected:
   int roll;
public:
   string name;
   void setdata(int a, string n){
   roll=a;
   name=n;
   }
   void display(){
   cout << "student have roll number "<< roll<< " and name " <<name <<endl;
   }
};

class exam: public student{
protected:
int maths, physics;
public:
   void setmarks( int m, int p){
   maths=m;
   physics=p;
   }
   int getm(){
   return maths;
   }
   int getp(){
   return physics;
   }
};

class reexam{
protected:
   int extraSubject;
public:
   void setExtraMarks(int m){
   extraSubject = m;
   }
   void getExtraMarks(){
   cout << "Score of reexam is : " << extraSubject <<endl;
   }
};
```

```
class result: public exam,public reexam{
public:
   float getresult(){
   cout << "Result of " <<name<< " is "<< ((float)getm()+(float)getp())/2<< "%"
<<endl<<endl;
   }
};


int main(){
result Ajay, unknown;
Ajay.setdata(2,"Ajay");
Ajay.display();
Ajay.setmarks(98,99);
Ajay.getresult();

unknown.setdata(36,"Unknown");
unknown.display();
unknown.setmarks(77,95);
unknown.getresult();
unknown.setExtraMarks(20);
unknown.getExtraMarks();

   return 0;
}
```

Output:



RESULT: code executed successfully.

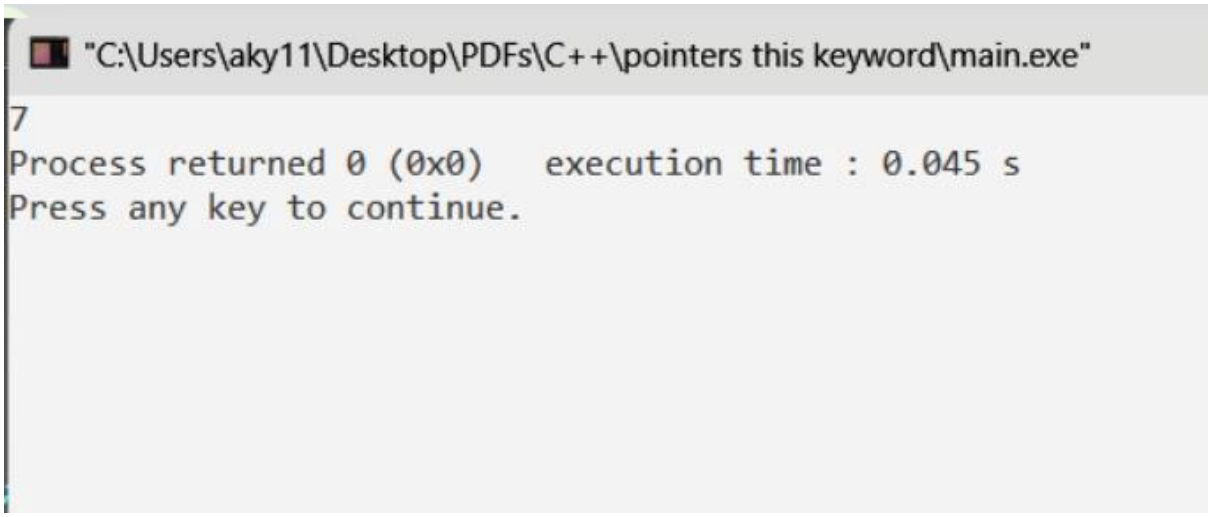
QUESTION 12: Program to understand the Use of “this” Pointer.

This pointer

EX.NO.: 12
DATE :16/10/2022
AIM: implementing this pointer CODE

```
#include <iostream>

using namespace std;
class A
{
   int a;
public:
   A& setdata(int a )
   {
      this -> a =a;
      return *this;
   } void getdata()
   {
      cout <<a;
   }
};
int main()
{
   A ajay;
   ajay.setdata(7).getdata();
   return 0;
}
```

Output:



Result: Code executed successfully

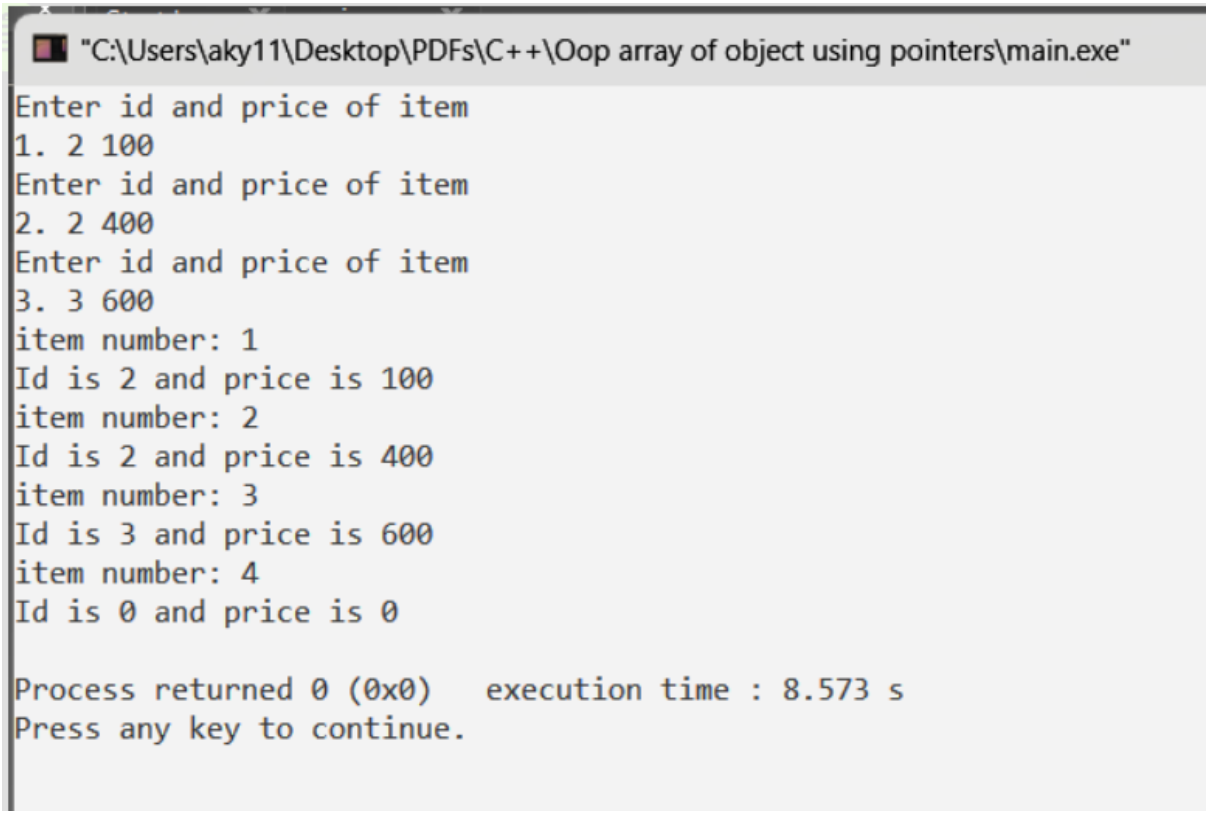
QUESTION 13 : Programs to Understand Storage Specifiers.

Storage specifiers

EX.NO.: 13
DATE :16/10/2022
AIM: Implementing program of storage specifiers.

```
#include <iostream>

using namespace std;

class shop{
int id;
float price;
public:
    void setdata(int i, int p){
    id=i;
    price=p;
    }void getdata(){
    cout << "Id is "<<id<< " and price is "<<price<<endl;
    }
};

int main()
{
   int size = 3;
   int p,q,i;
   shop *ptr = new shop[size];
   shop *ptrtem = ptr;
   shop *ptritem = ptr;
   for (i=0; i<size;i++){
    cout<< "Enter id and price of item "<<endl;
    cout<<i+1<< ". ";
    cin>>p>>q;
    ptr ->setdata(p,q);
    ptr++;
   } for (i=0;i<size;i++){
   cout<<"item number: "<<i+1<<endl;
   ptrtem ->getdata();
   ptrtem++;
   }
   //to show addition in pointer
   cout<<"item number: "<<i+1<<endl;
   (ptrtem+2) ->getdata();
    return 0;
}
```

Output:



RESULT: code executed successfully.

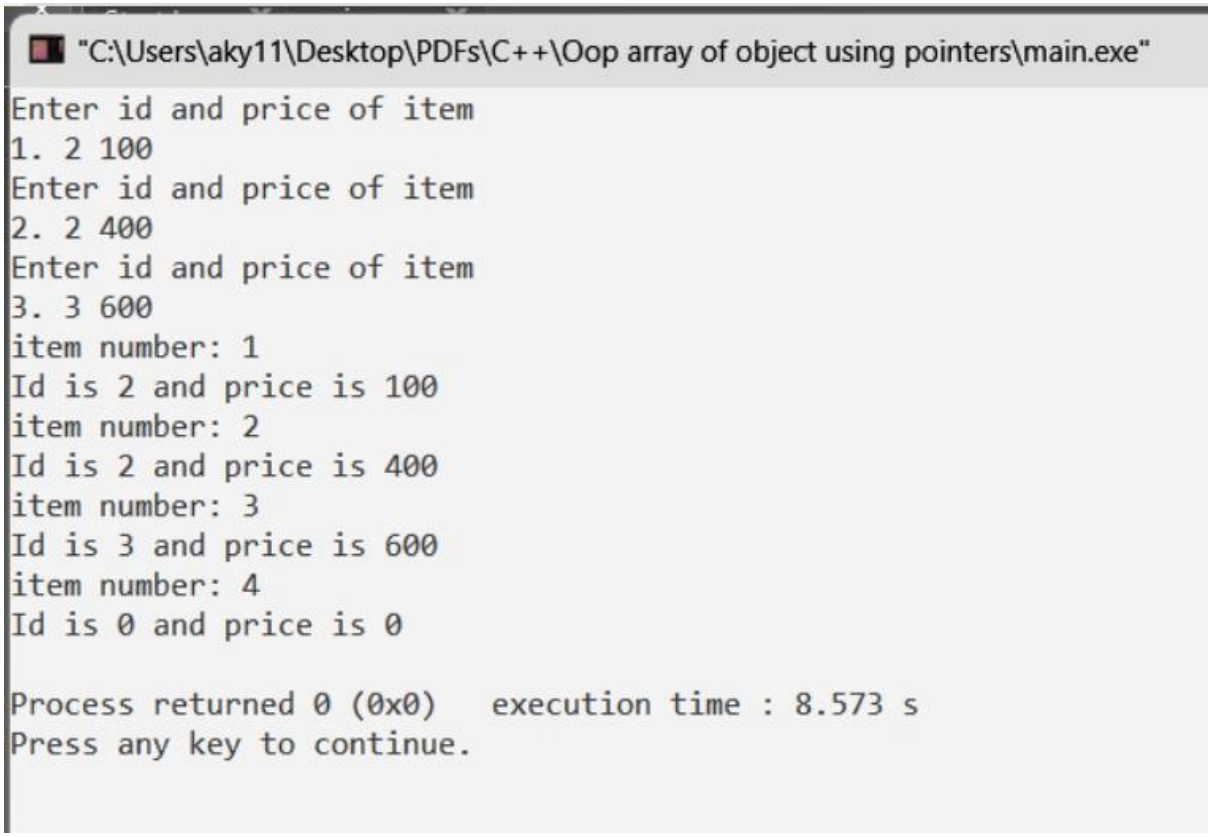
QUESTION 14 : Programs to implement the concept of Pointer Arithmetic. a) Increment/Decrement of a Pointer b) Addition of integer to a pointer.

Pointer arithmetic

EX.NO.: 14
DATE :16/10/2022
AIM: implementing program of pointer arithmetic.

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

class contained {
private:
int num;

public:
void shownum()
{
        cout << "Hello from contained class\n";
        cout << "num = " << num << endl;
}

int& getnum()
{
        return num;
}
};

class container {
contained c;

public:
container()
{
        cout << "Hello from container class\n";
        c.getnum() = 70;
        c.shownum();
}
};

int main()
{
container cont;
}
```

Output:



RESULT: code executed successfully.

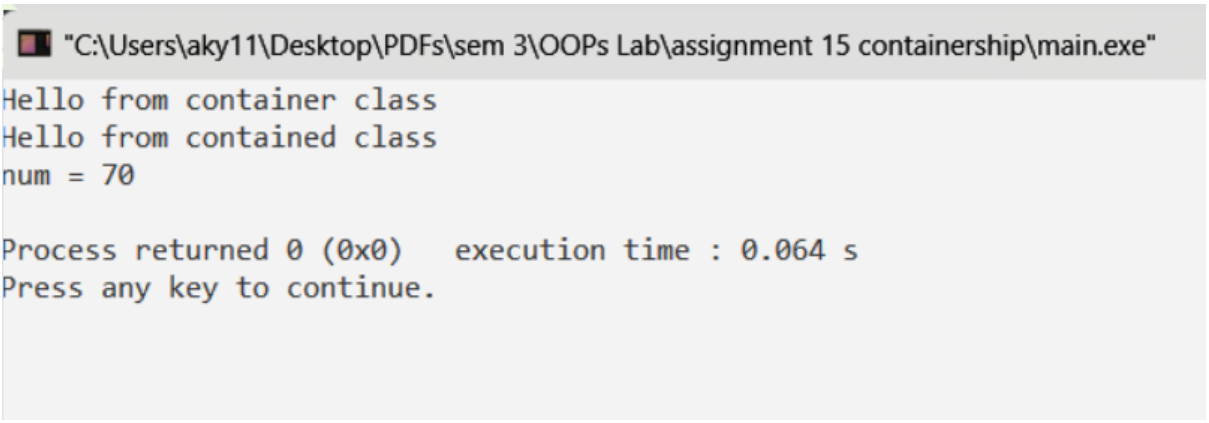
QUESTION 15 : Program to show the concept of containership.

Containership

EX.NO.: 15
DATE :20/10/2022
AIM: implementing program of containership.

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

class contained {
private:
int num;

public:
void shownum()
{
        cout << "Hello from contained class\n";
        cout << "num = " << num << endl;
}

int& getnum()
{
        return num;
}
};

class container {
contained c;

public:
container()
{
        cout << "Hello from container class\n";
        c.getnum() = 70;
        c.shownum();
}
};

int main()
{
container cont;
}
```

Output:



RESULT: code executed successfully.

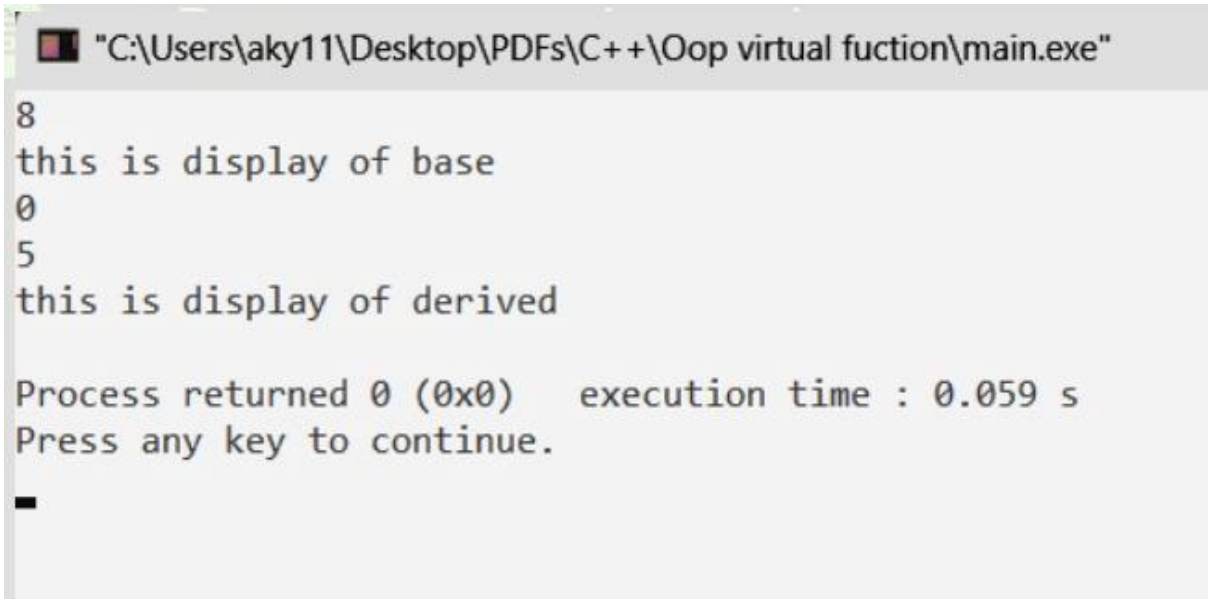
QUESTION 16 : Program to show the concept of run time polymorphism using virtual function.

Run time polymorphism using virtual function

EX.NO.: 16
DATE :20/10/2022
AIM: implementing run time polymorphism using virtual function.

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

class base{
protected:
   int a;
public:
   void setdata(int x)
   {
      a=x;
   } virtual void getdata()
   {
      cout<<a<<endl;
      cout<<"this is display of base"<<endl;
   }
};
class derive: public base{
   int b;
public:
   void setdata(int y)    {
      b=y;
   }
void getdata()    {
      cout <<b<<endl;
      cout<<a<<endl;
      cout<<"this is display of derived"<<endl;
   }
};

int main(){
   base * bp;
   base*q;
   base bo;
   derive deo;
   q=&bo;
   q ->setdata(8);
   q ->getdata();
   bp = &deo;
   bp ->setdata(5);
   bp ->getdata();


   return 0;
}
```

Output:



RESULT: code executed successfully.

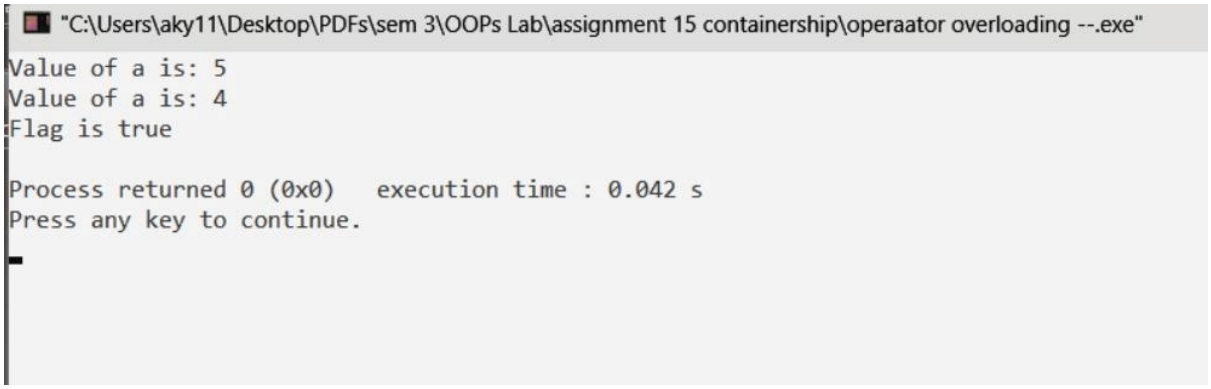
QUESTION 17: Program to overload unary operator. a) decrement (--) operator b) logical not (!) operator.

Operator overloading(unary)

EX.NO.: 17
DATE :31/10/2022
AIM: implementing operator overloading.

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

class A
{
   int num;
   bool flag;
public:
   A(int a)
   {
      num = a;
      flag = false;
   }
   void getnum()
   {
      cout << "Value of a is: "<< num <<endl;
   }
   bool getflag(){
   return flag;
   }
   void operator -()
   {
      num--;
   }
   void operator !()
   {
      flag = !flag;
   }
};

int main()
{
   A a(5);
   a.getnum();
   -a;
   a.getnum();
   if(a.getflag()) cout << "Flag is true"<<endl;
   !a;
   if(a.getflag()) cout << "Flag is true"<<endl;

   return 0;
}
```

Output:



Result: Code executed successfully

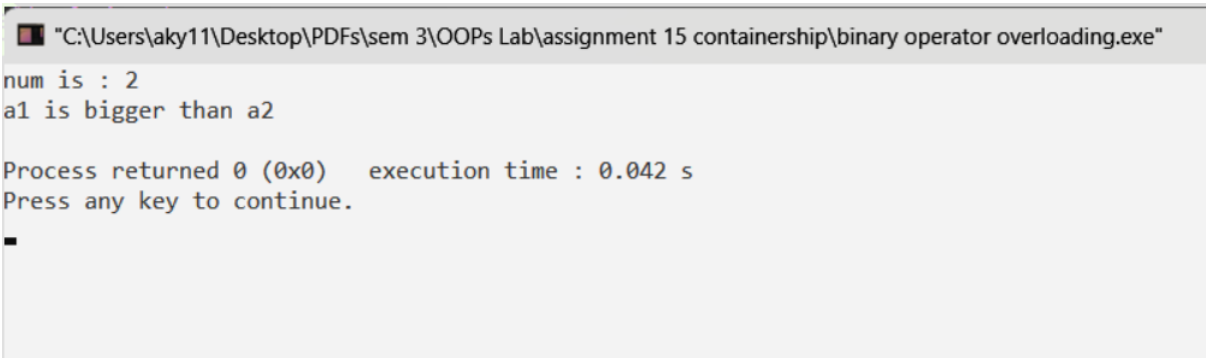
QUESTION 18: Program to overload binary operator. a) % operator b) <= operator

Operator overloading(Binary)

EX.NO.: 18
DATE :31/10/2022
AIM: Implementing operator overloading.

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

class A{
int num;
public:
   A(int n){
   num =n;
   }
   void getnum(){
   cout<< "num is : "<<num<<endl;
   }
   A operator % (A &obj){
   A res(0);
   res.num = this->num % obj.num;
   return res;
   }
   bool operator <= (A &obj){
   bool res;
   res = this->num <= obj.num;
   return res;
   }
} ;

int main(){
A a1(8),a2(3);
A a3 = a1%a2;
a3.getnum();
a1<=a2? cout<< "a2 is bigger than a1" <<endl: cout << "a1 is bigger than a2"<<endl;
return 0;
}
```

Output:



Result: Code executed successfully

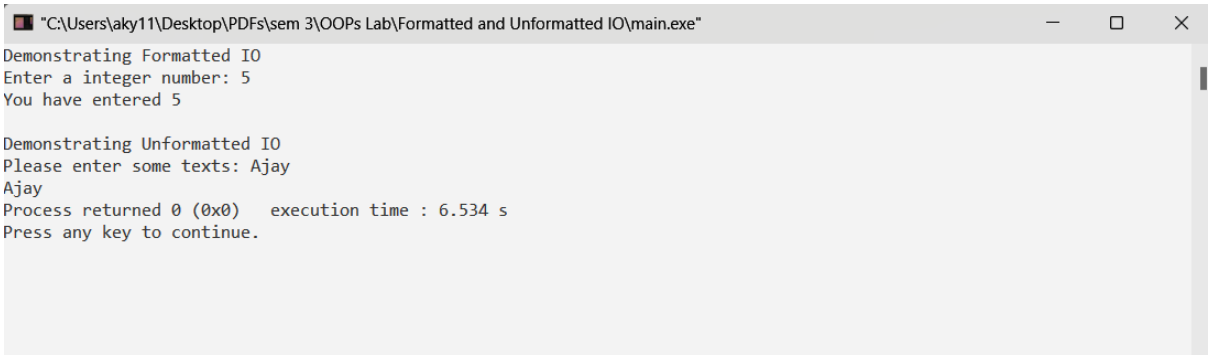
QUESTION 19: Program to work with formatted and unformatted IO operations

Formatted/Unformatted IO

EX.NO.: 19
DATE :01/11/2022
AIM: Formatted and Unformatted IO

```
#include <iostream>
#include <windows.h>
using namespace std;

int main()
{
   cout << "Demonstrating Formatted IO" <<endl;
   int num1;
   printf("Enter a integer number: ");
   scanf("%d", &num1);
   printf("You have entered %d\n\n", num1);

   cout << "Demonstrating Unformatted IO" <<endl;
   char name[50];
   printf("Please enter some texts: ");
   scanf("\n");
   gets(name);
   printf("%s",name);
   return 0;
}
```

Output:



Result: Code executed successfully