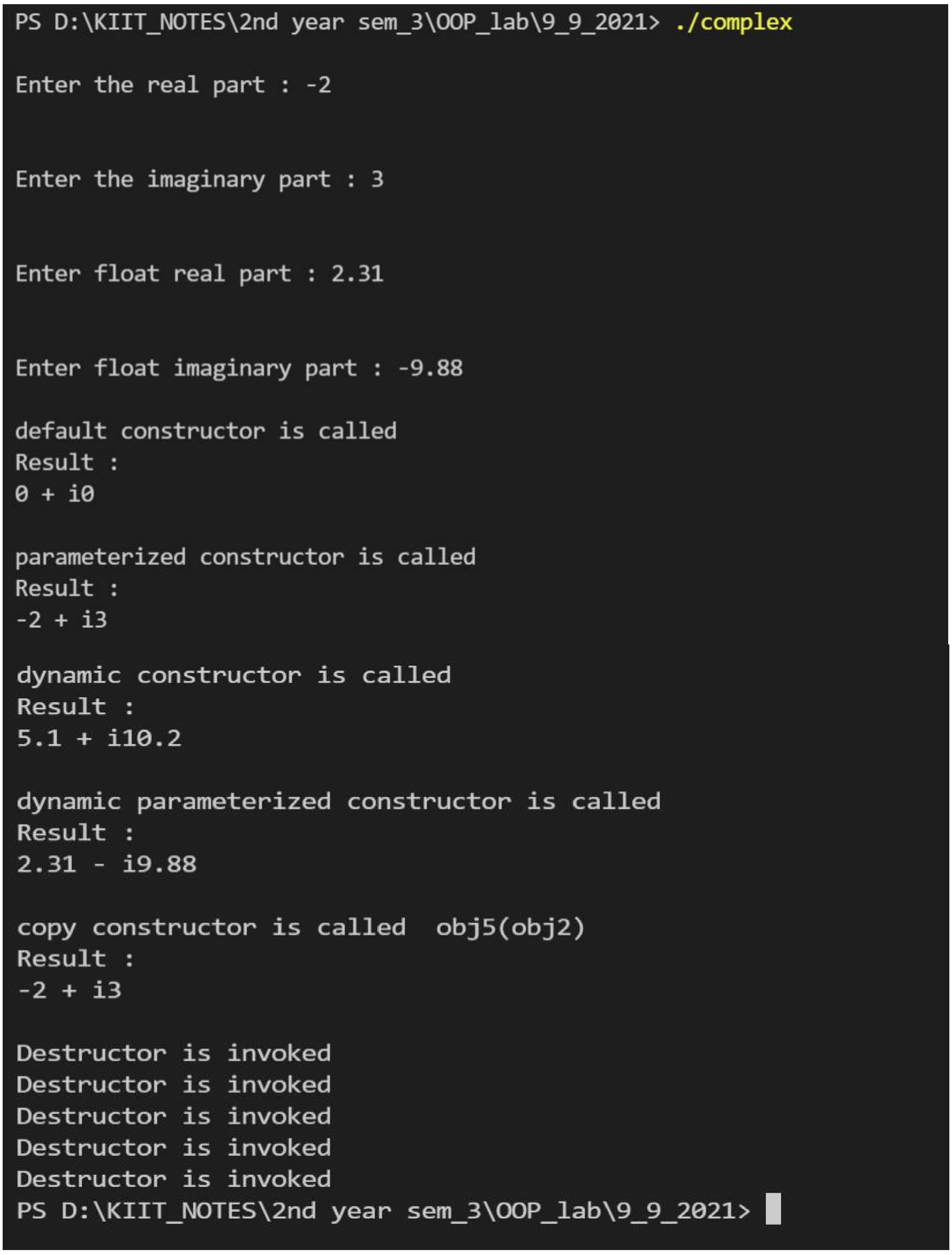
Name - Akriti Choudhary
Roll number - 2005776
Lab7
Subject - OOP lab
Class - B14
Branch - CSE
Date- 9/09/2021

Question 1:Create a class complex which stores real and imaginary part of a complex number. Include all types of constructors and destructor. The destructor should display a message about the destructor being invoked. Create objects using different constructors and display them.

```
#include <iostream>
using namespace std;
class complex
{
private:
   int real;
   int img;
   float *realPtr;
   float *imgPtr;

public:
   complex() //default constructor
   {
      real = 0;
      img = 0;
      cout << "default constructor is called" << endl;
   }
   complex(int r, int i) //parameterized constructor
      : real(r), img(i)
   {
      cout << "parameterized constructor is called" << endl;
   }
   complex(int a) //dynamic constructor
   {
      realPtr = new float;
      *realPtr = 5.1;
      imgPtr = new float;
      *imgPtr = 10.2;
      cout << "dynamic constructor is called" << endl;
   }
   complex(float re, float im) //dynamic parameterized constructor
   {
      realPtr = new float;
      *realPtr = re;
      imgPtr = new float;
      *imgPtr = im;
      cout << "dynamic parameterized constructor is called" << endl;
   }
   complex(complex &ob) //copy constructor
   {
      real = ob.real;
      img = ob.img;
      cout << "copy constructor is called  obj5(obj2) " << endl;
   }
   void display()
   {
      cout << "Result : " << endl;
      if (img >= 0)
         cout << real << " + i" << img << endl;
      else
```

```
        cout << real << " - i" << -(img) << endl;
      cout<<endl;
    }
    void display(int a)
    {
      cout << "Result : " << endl;
      if (*imgPtr >= 0)
        cout << *realPtr << " + i" << *imgPtr << endl;
      else
        cout << *realPtr << " - i" << -(*imgPtr) << endl;
      cout<<endl;

    }
    ~complex()
    {
      delete (realPtr);
      delete (imgPtr);
      cout << "Destructor is invoked" << endl;
    }
};
int main()
{
    int r, i;
    cout << "\nEnter the real part : ";
    cin >> r;
    cout << endl;
    cout << "\nEnter the imaginary part : ";
    cin >> i;
    cout << endl;

    float ree, imm;
    cout << "\nEnter float real part : ";
    cin >> ree;
    cout << endl;
    cout << "\nEnter float imaginary part : ";
    cin >> imm;
    cout << endl;

    complex obj1; //default constructor
    obj1.display();

    complex obj2(r, i); //parameterized constructor
    obj2.display();

    complex obj3(0); //dynamic constructor
    obj3.display(0);

    complex obj4(ree, imm); //dynamic parameterized constructor
    obj4.display(0);

    complex obj5(obj2); //copy constructor
    obj5.display();

    return 0;
}
```

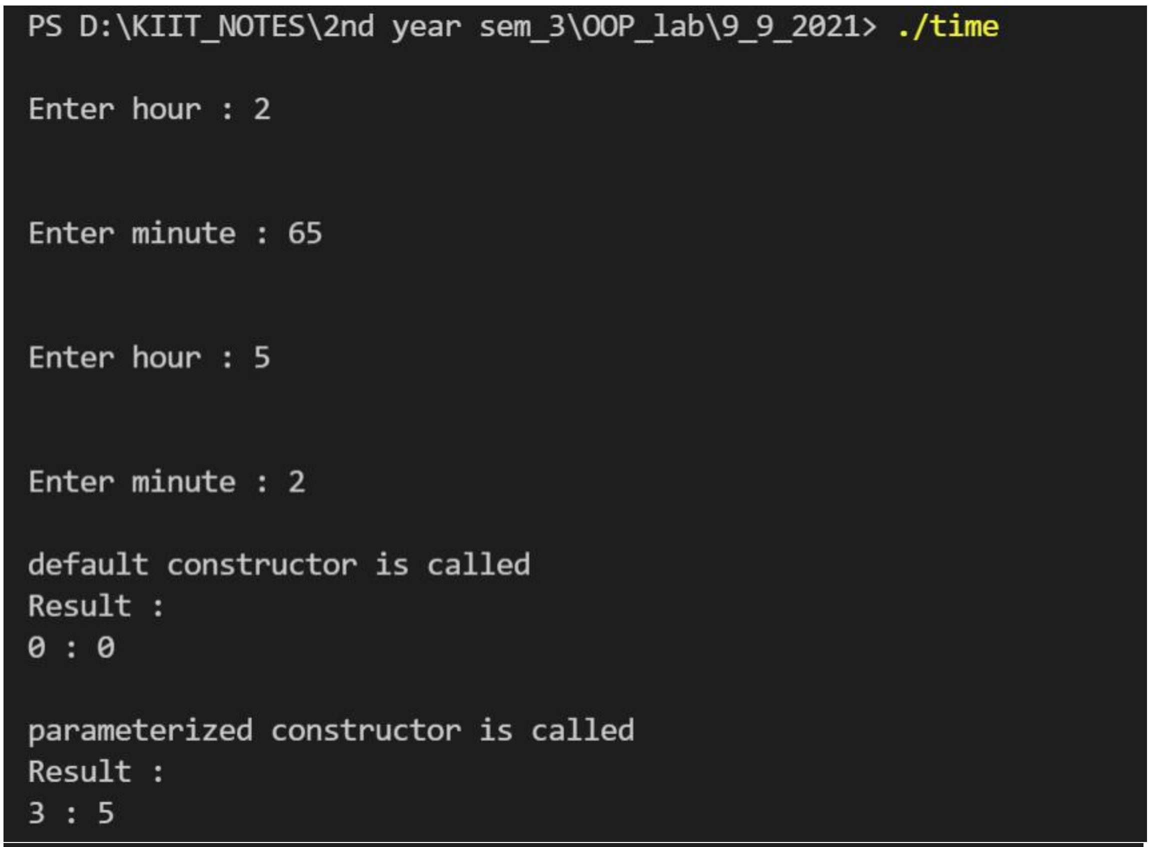


```
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021> ./complex

Enter the real part : -2


Enter the imaginary part : 3


Enter float real part : 2.31


Enter float imaginary part : -9.88

default constructor is called
Result :
0 + i0

parameterized constructor is called
Result :
-2 + i3

dynamic constructor is called
Result :
5.1 + i10.2

dynamic parameterized constructor is called
Result :
2.31 - i9.88

copy constructor is called  obj5(obj2)
Result :
-2 + i3

Destructor is invoked
Destructor is invoked
Destructor is invoked
Destructor is invoked
Destructor is invoked
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021>
```

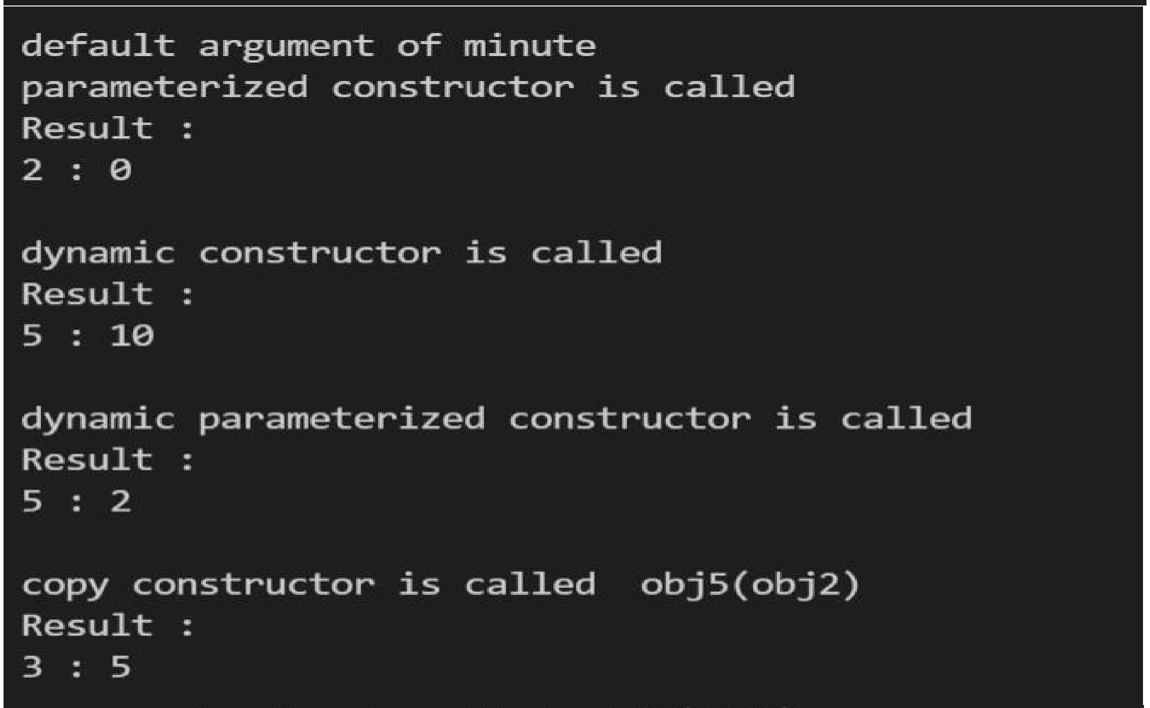
Question2 : Create a class which stores time in hh:mm format. Include all the constructors. The parameterized constructor should initialize the minute value to zero, if it is not provided.

```
#include <iostream>
using namespace std;
class timeFormat
{
private:
   int hour;
   int minute;
   int *hPtr;
   int *mPtr;

public:
   timeFormat() //default constructor
   {
      hour = 0;
      minute = 0;
      cout << "default constructor is called" << endl;
   }
   timeFormat(int h, int m = 0) //parameterized constructor
      : hour(h), minute(m)
   {
      cout << "parameterized constructor is called" << endl;
   }
   timeFormat(int a, int b, int c) //dynamic constructor
   {
      hPtr = new int;
      *hPtr = 5;
      mPtr = new int;
      *mPtr = 10;
      cout << "dynamic constructor is called" << endl;
   }
   timeFormat(unsigned int h, unsigned int m) //dynamic parameterized constructor
   {
      hPtr = new int;
      *hPtr = h;
      mPtr = new int;
      *mPtr = m;
      cout << "dynamic parameterized constructor is called" << endl;
   }
   timeFormat(timeFormat &ob) //copy constructor
   {
      hour = ob.hour;
      minute = ob.minute;
      cout << "copy constructor is called  obj5(obj2) " << endl;
   }
   void display()
   {
      cout << "Result : " << endl;
      if (minute <= 60)
         cout << hour << " : " << minute << endl;
      else
         cout << hour + 1 << " : " << minute - 60 << endl;
      cout << endl;
   }
```

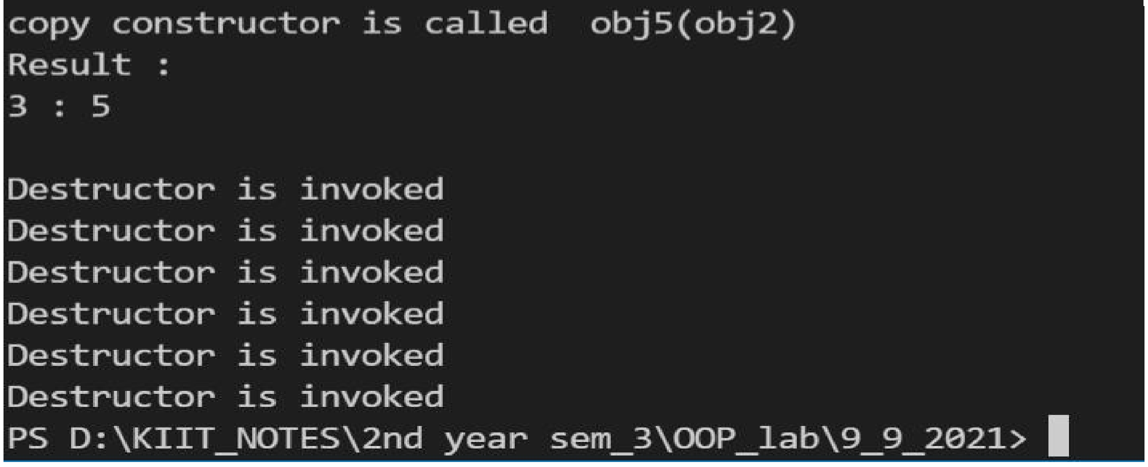
```
    void display(int a)
    {
        cout << "Result : " << endl;
        if (*mPtr <= 60)
            cout << *hPtr << " : " << *mPtr << endl;
        else
            cout << *hPtr + 1 << " : " << *mPtr - 60 << endl;
        cout << endl;
    }
    ~timeFormat()
    {
        delete (hPtr);
        delete (mPtr);
        cout << "Destructor is invoked" << endl;
    }
};
int main()
{
    int h, m;
    cout << "\nEnter hour : ";
    cin >> h;
    cout << endl;
    cout << "\nEnter minute : ";
    cin >> m;
    cout << endl;

    unsigned int hh, mm;
    cout << "\nEnter hour : ";
    cin >> hh;
    cout << endl;
    cout << "\nEnter minute : ";
    cin >> mm;
    cout << endl;

    timeFormat obj1; //default constructor
    obj1.display();

    timeFormat obj2(h, m); //parameterized constructor
    obj2.display();

    cout<<"default argument of minute"<<endl;
    timeFormat obj6(h); //parameterized constructor with one default argument
    obj6.display();

    timeFormat obj3(0, 0, 0); //dynamic constructor
    obj3.display(0);

    timeFormat obj4(hh, mm); //dynamic parameterized constructor
    obj4.display(0);

    timeFormat obj5(obj2); //copy constructor
    obj5.display();

    return 0;
}
```



```
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021> ./time

 Enter hour : 2


 Enter minute : 65


 Enter hour : 5


 Enter minute : 2

 default constructor is called
 Result :
 0 : 0

 parameterized constructor is called
 Result :
 3 : 5
```



```
 default argument of minute
 parameterized constructor is called
 Result :
 2 : 0

 dynamic constructor is called
 Result :
 5 : 10

 dynamic parameterized constructor is called
 Result :
 5 : 2

 copy constructor is called  obj5(obj2)
 Result :
 3 : 5
```

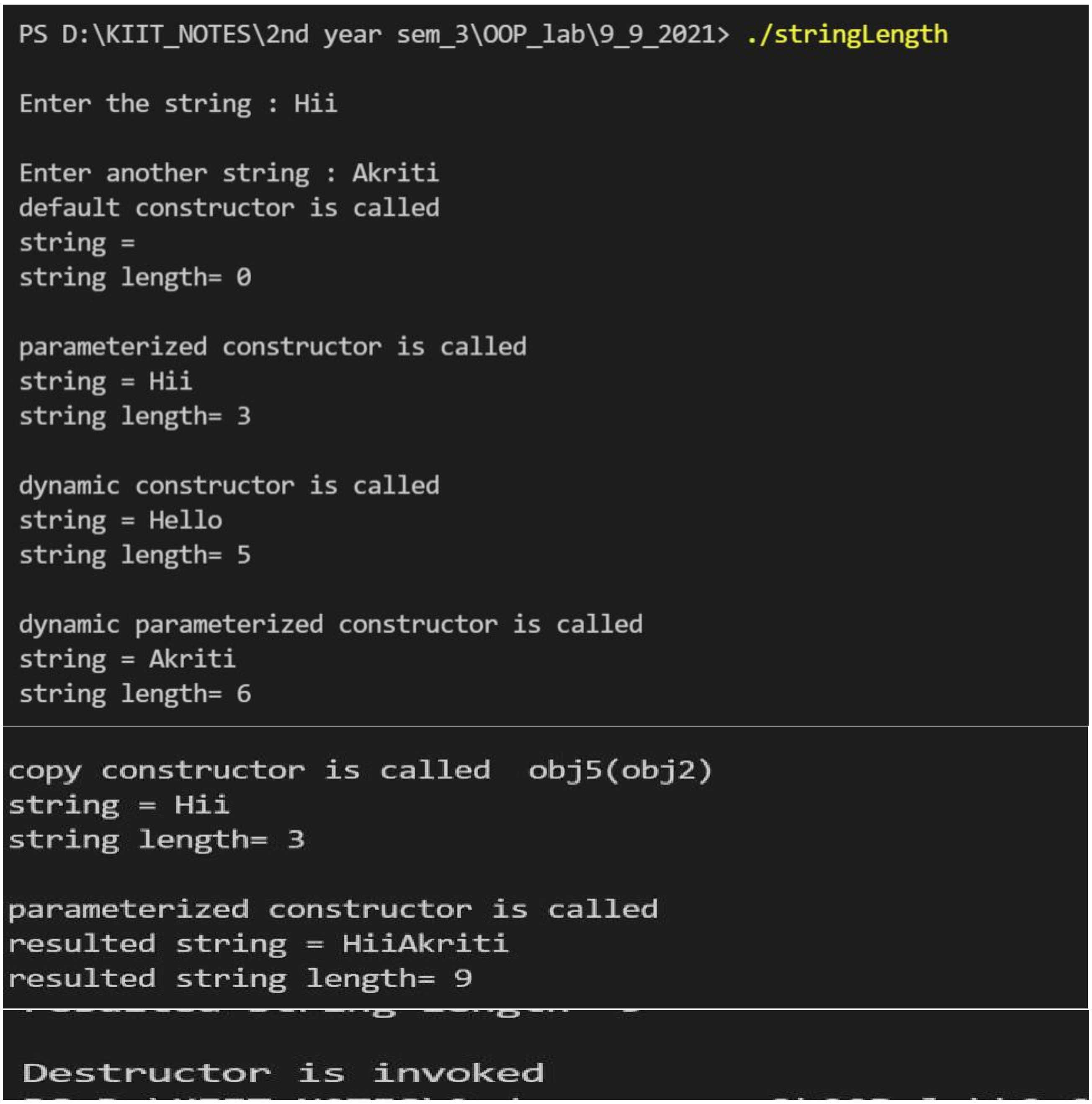


```
copy constructor is called  obj5(obj2)
Result :
3 : 5
```

```
Destructor is invoked
Destructor is invoked
Destructor is invoked
Destructor is invoked
Destructor is invoked
Destructor is invoked
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021>
```

Question3 : Create a class which stores a sting and its length as data members. Include all the constructors. Include a member function to join two strings and display the concatenated string.

```
#include <iostream>
#include <string>
using namespace std;
class stringLength
{
private:
   string s;
   string s1;
   int len;
   string *sPtr;
   int *lenPtr;

public:
   stringLength() //default constructor
   {
      s = "";
      len = 0;
      cout << "default constructor is called" << endl;
   }

   stringLength(string S, int l) //parameterized constructor
      : s(S), len(l)
   {
      cout << "parameterized constructor is called" << endl;
   }

   stringLength(string S, string ss) //parameterized constructor
   {
      s = S;
      s1 = ss;
      cout << "parameterized constructor is called" << endl;
   }

   stringLength(int a) //dynamic constructor
   {
      sPtr = new string;
      *sPtr = "Hello";
      lenPtr = new int;
      *lenPtr = (*sPtr).length();
      cout << "dynamic constructor is called" << endl;
   }

   stringLength(string S, float L) //dynamic parameterized constructor
   {
      sPtr = new string;
      *sPtr = S;
      lenPtr = new int;
      *lenPtr = L;
      cout << "dynamic parameterized constructor is called" << endl;
   }

   stringLength(stringLength &ob) //copy constructor
   {
        s = ob.s;
        len = ob.len;
        cout << "copy constructor is called  obj5(obj2) " << endl;
    }

    ~stringLength();

    void display()
    {
        cout << "string = " << s << endl;
        cout << "string length= " << len << endl;
        cout << endl;
    }

    void display(int a)
    {
        cout << "string = " << *sPtr << endl;
        cout << "string length= " << *lenPtr << endl;
        cout << endl;
    }

    void add()
    {
        string s3 = s.append(s1);
        int ll = s3.length();
        cout << "resulted string = " << s3 << endl;
        cout << "resulted string length= " << ll << endl;
        cout << endl;
    }
};
stringLength::~stringLength()
{

    cout << "Destructor is invoked" << endl;
    delete (sPtr);
    delete (lenPtr);
}
int main()
{
    string r;
    int i;
    cout << "\nEnter the string : ";
    cin >> r;
    i = r.length();

    string ree;
    float imm;
    cout << "\nEnter another string : ";
    cin >> ree;
    imm = ree.length();

    stringLength obj1; //default constructor
    obj1.display();

    stringLength obj2(r, i); //parameterized constructor
    obj2.display();

    stringLength obj3(0); //dynamic constructor
    obj3.display(0);

    stringLength obj4(ree, imm); //dynamic parameterized constructor
    obj4.display(0);

    stringLength obj5(obj2); //copy constructor
    obj5.display();

    stringLength obj6(r, ree); //parameterized constructor
    obj6.add();

    return 0;
}
```



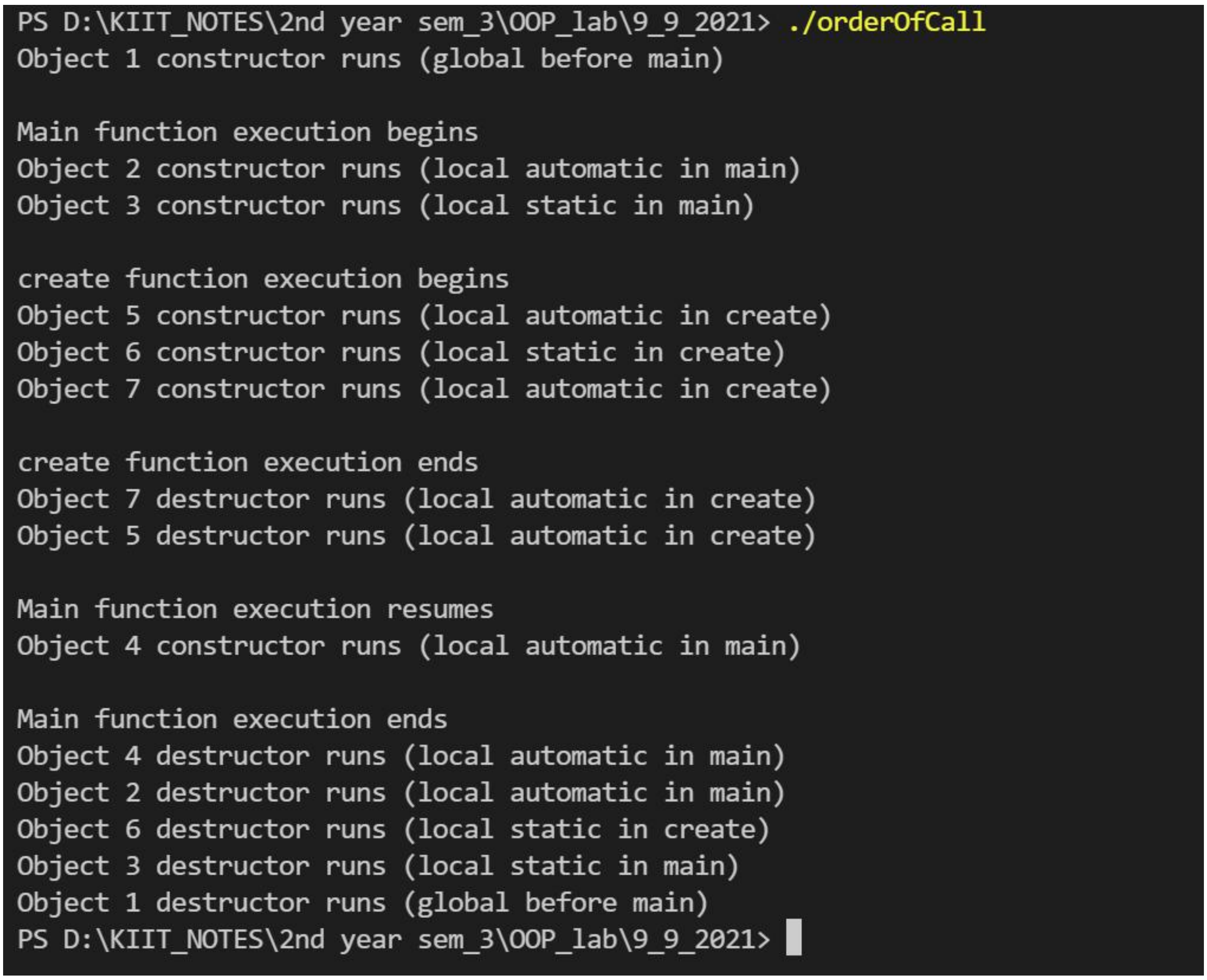
```
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021> ./stringLength

Enter the string : Hii

Enter another string : Akriti
default constructor is called
string =
string length= 0

parameterized constructor is called
string = Hii
string length= 3

dynamic constructor is called
string = Hello
string length= 5

dynamic parameterized constructor is called
string = Akriti
string length= 6
```

```
copy constructor is called  obj5(obj2)
string = Hii
string length= 3

parameterized constructor is called
resulted string = HiiAkriti
resulted string length= 9
```

```
Destructor is invoked
```

Question4 :WAP to demonstrate the order of call of constructors and destructors for a class.

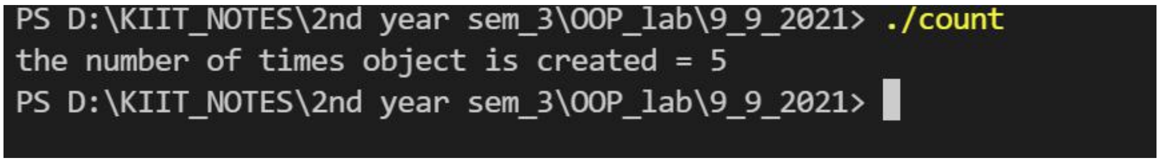
```
#include <iostream>
#include <string>
using namespace std;
class createAndDestroy
{
public:
   createAndDestroy(int, string); //constructor
   ~createAndDestroy();          //destructor

private:
   int objectID;
   string message;
};

createAndDestroy::createAndDestroy(int ID, string messageString)
   : objectID(ID), message(messageString)
{
   cout << "Object " << objectID << " constructor runs " << message << endl;
}

createAndDestroy::~createAndDestroy()
{
   cout << "Object " << objectID << " destructor runs " << message << endl;
}

void create();

createAndDestroy first(1, "(global before main)");
int main()
{
   cout << "\nMain function execution begins" << endl;
   createAndDestroy second(2, "(local automatic in main)");
   static createAndDestroy third(3, "(local static in main)");
   create();
   cout << "\nMain function execution resumes" << endl;
   createAndDestroy fourth(4, "(local automatic in main)");
   cout << "\nMain function execution ends" << endl;

   return 0;
}
void create()
{
   cout << "\ncreate function execution begins" << endl;
   createAndDestroy fifth(5, "(local automatic in create)");
   static createAndDestroy sixth(6, "(local static in create)");
   createAndDestroy seventh(7, "(local automatic in create)");
   cout << "\ncreate function execution ends" << endl;
}
```



```
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021> ./orderOfCall
Object 1 constructor runs (global before main)

Main function execution begins
Object 2 constructor runs (local automatic in main)
Object 3 constructor runs (local static in main)

create function execution begins
Object 5 constructor runs (local automatic in create)
Object 6 constructor runs (local static in create)
Object 7 constructor runs (local automatic in create)

create function execution ends
Object 7 destructor runs (local automatic in create)
Object 5 destructor runs (local automatic in create)

Main function execution resumes
Object 4 constructor runs (local automatic in main)

Main function execution ends
Object 4 destructor runs (local automatic in main)
Object 2 destructor runs (local automatic in main)
Object 6 destructor runs (local static in create)
Object 3 destructor runs (local static in main)
Object 1 destructor runs (global before main)
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021>
```

Question5 :WAP to count number of objects created from a class using concept of static data members and static member function.

```
#include <iostream>
using namespace std;
class book
{
  static int a;

public:
  book()
  {
    a++;
  }
  static void display()
  {
    cout << "the number of times object is created = " << book::a << endl;
  }
};
int book ::a;
int main()
{
  book obj1, obj2, obj3, obj4, obj5;
  obj1.display();

  return 0;
}
```



```
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021> ./count
the number of times object is created = 5
PS D:\KIIT_NOTES\2nd year sem_3\OOP_lab\9_9_2021>
```