NORTHERN UNIVERSITY
BANGLADESH
Knowledge for Innovation and Change

Assignment 3

Course code: CSE 1360

Submitted By

Student name: Tarif Uddin Razi

Student ID: 42250202537

Section: 2C

Department: **Computer Science and Engineering**, NUB

Submitted To

Faculty name: Tamanna Afroz Lecturer

Lecturer, **Computer Science and Engineering**, NUB

Submitted on 08/11/2025

TASK-1

You are given the following “University” class:

public class University{

public String name;

public String country;

}

Now write a tester class named “University Tester”. a. Write the main method and create 2 objects of University class and print the location of

the objects and print the instance variables of the objects. Are the location of the objects

the same?

b. Now change the instance variables of the first object. name = “Imperial College London”

country = “England” Now change the instance variables of the second object. name = “Brac University”

country = “Bangladesh” Now check if the instance variables of both objects have changed or not and whether the

instance variables of both objects are of the same value or not.

solution:

```
#include <iostream>

#include <string>

using namespace std;


class University {

public:

    string name;
```

```
    string location;
};

int main() {

    University uniA;
    University uniB;

    cout<<"Location of uniA"<<&uniA<<endl;
    cout<<"Location of uniB"<<&uniB<<endl;

    if (&uniA==&uniB){
        cout<<"Same Location"<<endl;
    }
    else{
        cout<<"Different Location"<<endl;
    };

    uniA.name="Imperial College London";
    uniA.location="London";

    uniB.name="Brac University";
    uniB.location="Bangladesh";

    cout<<"After assign vales:"<<endl;
```

```
    cout<<"University A: "<<uniA.name<<", Country: "<<uniA.location<<endl;
    cout<<"University B: "<<uniB.name<<", Country: "<<uniB.location<<endl;

    if (uniA.name==uniB.name && uniA.location==uniB.location)
    {
        cout<<"Same Value"<<endl;
    }
    else{
        cout<<"Different Value"<<endl;
    }
    return 0;
}
```

Task -2

Write the driver code of “Circle” class to generate the following output:

```
int main() {
//Your code here
}
```

Design Class Output

```
class Circle {
public:
double radius = 5;
}
```

Radius of the circle is 5.0

The area of the circle is 78.53981633974483

The circumference of the circle is

31.41592653589793

solution:

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


class Circle {
public:
   double radius=5;
};

int main(){
   Circle circle;

   double area = M_PI*circle.radius*circle.radius;
   double circumference = 2*M_PI*circle.radius;

   cout <<"Redius of the circle is  "<<circle.radius<< endl;
   cout<<"The area of the circle is "<<area<<endl;
```

```
    cout<<"The circumference of the circle is "<<circumference<<endl;

    return 0;
}
```

Task—3

Design the “Student” class so that the main method prints the following:

Tester Class Output

```
#include <iostream>
#include <string>
int main() {
Student s1;
std::cout << "Name of the Student: " << s1.name << std::endl;
std::cout << "ID of the Student: " << s1.id << std::endl;
s1.id = 123;
std::cout << "ID of the Student: " << s1.id << std::endl;
return 0;
}
```

Name of the Student: Bob

ID of the Student: 1

ID of the Student: 123

Solution:

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

class Student {
public:
    std::string name = "Bob";
    int id = 1;
};

int main() {
    Student s1;

    cout << "Name of the Student: " << s1.name << endl;
    cout << "ID of the Student: " << s1.id << endl;

    s1.id = 123;
    cout << "ID of the Student: " << s1.id << endl;

    return 0;
}
```

Task 4

Write the code for the “Vehicle” class. The tester class and the output is given below:

Tester class	Output

#include <iostream>
#include <string>
int main() {
Vehicle car;
std::cout << "Attributes of car object:" << std::endl;
std::cout << car.type << std::endl;
std::cout << car.wheels << std::endl;
std::cout << car.color << std::endl;
std::cout << "=========" << std::endl;
Vehicle bike;
bike.type = "Motor bike";
bike.wheels = 2;
bike.color = "Red";
std::cout << "Attributes of bike object:" << std::endl;
std::cout << bike.type << std::endl;
std::cout << bike.wheels << std::endl;
std::cout << bike.color << std::endl;
return 0;}
	Attributes of car object:
	Car
	4
	White
	=========
	Attributes of bike object:
	Motor bike
	2
	Red

Solution:

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

class Vehicle{
public:
    string type = "Car";
    int wheels = 4;
    string color = "White";
};

int main() {
    Vehicle car;
    cout << "Attributes of car object:" << endl;
    cout << car.type << endl;
    cout << car.wheels << endl;
    cout << car.color << endl;
    cout << "=========" << endl;

    Vehicle bike;
    bike.type = "Motor bike";
    bike.wheels = 2;
    bike.color = "Red";
    cout << "Attributes of bike object:" << endl;
```

```
    cout << bike.type << endl;
    cout << bike.wheels << endl;
    cout << bike.color << endl;
return 0;
}
```

Task 5

Write the code for the "**Tournament**" class. The tester class and the **output** is given below:

Tester class	Output

#include <iostream>
#include <string>
#include <vector>
 	
class Tester5 {
public:
static void main() {
Tournament asiaCup;
std::cout << asiaCup.name << " " << asiaCup.sportsType << " "
<< asiaCup.numberOfTeams << " ";
for (const auto& team : asiaCup.teams)
{ std::cout << team << " ";
}
std::cout << std::endl;
std::cout << "**********" << std::endl;
asiaCup.name = "Asia Cup";
asiaCup.sportsType = "Cricket";
asiaCup.numberOfTeams = 4;
asiaCup.teams = {"BD", "IND", "PAK", "SL"};
std::printf("%s %s Tournament is played between %d teams\n", asiaCup.name.c_str(), asiaCup.sportsType.c_str(), asiaCup.numberOfTeams);
 	
std::cout << "The teams are:" << std::endl;
for (size_t i = 0; i < asiaCup.teams.size(); i++) { std::cout
<< asiaCup.teams[i] << std::endl;
}
}
};
	null null 0 null
	**********
	Asia Cup Cricket Tournament is played between 4 teams The teams are:
	BD
	IND
	PAK
	SL

Solution:

```
#include <iostream>
#include <string>
#include <vector>

using namespace std;

class Tournament {
public:
   std::string name;
   std::string sportsType;
   int numberOfTeams;
   std::vector<std::string> teams;
};

class Tester5 {
public:
   static void main() {
      Tournament asiaCup;
      std::cout << asiaCup.name << " " << asiaCup.sportsType << " "<< asiaCup.numberOfTeams
<< " ";
      for (const auto& team : asiaCup.teams) {
      std::cout << team << " ";
   }

   std::cout << std::endl;
   std::cout << "***************" << std::endl;

   asiaCup.name = "Asia Cup";
   asiaCup.sportsType = "Cricket";
   asiaCup.numberOfTeams = 4;
   asiaCup.teams = {"BD", "IND", "PAK", "SL"};
```

```
    std::printf("%s %s Tournament is played between %d teams\n",
    asiaCup.name.c_str(), asiaCup.sportsType.c_str(),
    asiaCup.numberOfTeams);

    std::cout << "The teams are:" << std::endl;

    for (size_t i = 0; i < asiaCup.teams.size(); i++) {
        std::cout << asiaCup.teams[i] << std::endl;
    }

    }
};

int main() {
    Tester5::main();  // call the static function
    return 0;
}
```

Task 6

Design the "**ImaginaryNumber**" to generate the **output** given below:

Tester Class	Output

#include <iostream>
int main() {
ImaginaryNumber num1;
num1.printNumber();
std::cout << "1********" << std::endl;
num1.realPart = 3;
num1.imaginaryPart = 7;
num1.printNumber();
std::cout << "2********" << std::endl;
ImaginaryNumber num2;
	0 + 0i
	1********
	3 + 7i
	2********
	1 + 9i

num2.realPart = 1;
num2.imaginaryPart = 9;
num2.printNumber();
return 0;
}	

Solution:

```
#include <iostream>

class ImaginaryNumber{
   public:
   int realPart = 0;
   int imaginaryPart = 0;
   void printNumber(){
      std::cout << realPart << " + " << imaginaryPart << "i" << std::endl;
   }
};

int main() {
   ImaginaryNumber num1;
   num1.printNumber();
   std::cout << "1********" << std::endl;
   num1.realPart = 3;
   num1.imaginaryPart = 7;
   num1.printNumber();
   std::cout << "2********" << std::endl;
   ImaginaryNumber num2;
   num2.realPart = 1;
   num2.imaginaryPart = 9;
   num2.printNumber();

   return 0;
}
```

Task 7

Complete the **“Cat”** class so the main method produces the following output:

Test Class	Output

int main() {
Cat c1;
cout << "==========\n";
c1.printCat();
c1.color = "Black";
cout << "==========\n";
c1.printCat();
c1.color = "Brown";
c1.action = "jumping";
cout << "==========\n";
c1.printCat();
return 0;
}
	==========
	White cat is sitting
	==========
	Black cat is sitting
	==========
	Brown cat is jumping

Solution:

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

class Cat{
   public:
   string color = "White";
   string action = "sitting";
```

```
    void printCat(){
        cout<< color<<"cat is "<< action<< endl;
    }

};

int main() {
    Cat c1;
    cout << "====================\n";
    c1.printCat();
    c1.color = "Black";
    cout << "====================\n";
    c1.printCat();
    c1.color = "Brown";
    c1.action = "jumping";
    cout << "====================\n";
    c1.printCat();

    return 0;
}
```

Task 8

Complete the **Bird** class so that main method produces the following **output**:

Test class	Output
int main() {	Parrot has flown up 3 feet.

Bird b1;
b1.name = "Parrot";
b1.flyUp(3);
b1.makeNoise();
b1.flyDown(5);
b1.flyDown(2);
b1.flyDown(1);
 	
Bird b2;
b2.name = "Eagle";
b2.flyUp(5);
b2.flyDown(5);
b2.makeNoise();
return 0;
}
	Squawk
	Parrot cannot fly down 5 feet.
	Parrot has flown down 2 feet.
	Parrot has flown down 1 feet and landed.
	Eagle has flown up 5 feet.
	Eagle has flown down 5 feet and landed.
	Squee

Solution:

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

class Bird {
public:
    string name;
    int position = 0;

    void flyUp(int feet) {
        position += feet;
        cout << name << " has flown up " << feet << " feet." << endl;
```

```
    }
    void flyDown(int feet) {
        if (feet > position) {
            cout << name << " cannot fly down " << feet << " feet." << endl;
        } else {
            position -= feet;
            cout << name << " has flown down " << feet << " feet";
            if (position == 0)
                cout << " and landed.";
            cout << endl;
        }
    }
    void makeNoise() {
        if (name == "Parrot")
            cout << "Squawk" << endl;
        else if (name == "Eagle")
            cout << "Squee" << endl;
    }
};

int main() {
    Bird b1;
    b1.name = "Parrot";
    b1.flyUp(3);
    b1.makeNoise();
    b1.flyDown(5);
    b1.flyDown(2);
    b1.flyDown(1);
```

```
    Bird b2;
    b2.name = "Eagle";
    b2.flyUp(5);
    b2.flyDown(5);
    b2.makeNoise();

    return 0;
}
```

Task 9

Design the **CellPhone** class so that the **main** method of tester class can produce the following output:

Tester Class	Output

int main() {
CellPhone phone1;
phone1.printDetails();
cout << "1##################" << endl;
phone1.storeContact("Joy - 01834");
cout << "==================" <<
endl;
phone1.printDetails();
cout << "2##################" << endl;
phone1.storeContact("Toya - 01334");
phone1.storeContact("Aayan - 01135");
cout << "==================" <<
endl;
phone1.printDetails();
cout << "3##################" << endl;
phone1.storeContact("Sani - 01441");
cout << "==================" <<
endl;
	Phone Model unknown Contacts
	Stored 0
	1##################
	Contact Stored
	==================
	
	Phone Model Nokia 1100 Contacts
	Stored 1 Stored Contacts:
	
	Joy - 01834
	2##################
	Contact Stored Contact
	Stored
	
	==================
	
	Phone Model Nokia 1100 Contacts
	Stored 3 Stored Contacts:
	
	Joy - 01834
	Toya - 01334
	
	Aayan - 01135
	
	3##################

phone1.printDetails();
 	
return 0;
}
	Memory full. New contact can't be stored.
	==========
	Phone Model Nokia 1100 Contacts
	Stored 3 Stored Contacts:
	
	Joy - 01834
	Toya - 01334
	Aayan - 01135

Solution:

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

class CellPhone {
private:
    string model;
    string contacts[3];
    int contactCount;

public:
    CellPhone() {
        model = "unknown";
        contactCount = 0;
    }
```

```
void storeContact(string contact) {
    if (contactCount < 3) {
        contacts[contactCount] = contact;
        contactCount++;

        if (model == "unknown") {
            model = "Nokia 1100";
        }

        cout << "Contact Stored" << endl;
    } else {
        cout << "Memory full. New contact can't be stored." << endl;
    }
}

void printDetails() {
    cout << "Phone Model " << model << endl;
    cout << "Contacts Stored " << contactCount << endl;

    if (contactCount > 0) {
        cout << "Stored Contacts:" << endl;
```

```
            for (int i = 0; i < contactCount; i++) {
                cout << contacts[i] << endl;
            }
        }
    }
};

int main() {
    CellPhone phone1;
    phone1.printDetails();
    cout << "1###################" << endl;

    phone1.storeContact("Joy - 01834");
    cout << "====================" << endl;
    phone1.printDetails();

    cout << "2###################" << endl;
    phone1.storeContact("Toya - 01334");
    phone1.storeContact("Aayan - 01135");
    cout << "====================" << endl;
    phone1.printDetails();
```

```
    cout << "3##################" << endl;
    phone1.storeContact("Sani - 01441");
    cout << "====================" << endl;
    phone1.printDetails();
    return 0;
}
```

Task10

Consider the following class:

```
#include using namespace std;
class Human
{ public:
int age; double
height;
};
```

Show the output of the following sequence of statements:

Code:	Output:

int main() {
Human* h1 = new Human();
Human* h2 = new Human();
h1->age = 21;
h1->height = 5.5;
cout << h1->age << endl;
cout << h1->height << endl;
h2->height = h1->height - 3;
	21
	5.5
	2.5
	22
	22
	5.5
	23

cout << h2->height << endl;
h2->age = h1->age++;
cout << h1->age << endl;
h2 = h1;
cout << h2->age << endl;
cout << h2->height << endl;
h2->age++;
h2->height++;
cout << h1->age << endl;
cout << h1->height << endl;
h1->age = ++h2->age;
cout << h2->age << endl;
cout << h2->height << endl;
delete h1;
delete h2;
return 0;
};
	6.5
	24
	6.5
	free(): double free detected in tcache 2
	Aborted

Task 11

Consider the following class:

```
#include using namespace std;
class Student {
public:
string name;
double cgpa;
};
```

Show the output of the following sequence of statements:

Code	Output

int main() {
Human* h1 = new Human();
Human* h2 = new Human();
h1->age = 21;
h1->height = 5.5;
cout << h1->age << endl;
cout << h1->height << endl;
h2->height = h1->height - 3;
cout << h2->height << endl;
h2->age = h1->age++;
cout << h1->age << endl;
h2 = h1;
cout << h2->age << endl;
cout << h2->height << endl;
h2->age++;
h2->height++;
cout << h1->age << endl;
cout << h1->height << endl;
h1->age = ++h2->age;
cout << h2->age << endl;
cout << h2->height << endl;
delete h1;
delete h2;
return 0;
}
	New Student
	Student Two
	New Student
	2.3
	3.3
	2.3
	old student
	oldest student
	oldest student
	2.3
	3.5
	3.5