CL-1004
Object Oriented Programming
Lab # 10

Objectives:

- Polymorphism
- Virtual, pure virtual function
- Abstract and concrete classes
- Three file structure

Note: Carefully read the following instructions (*Each instruction contains a weightage*)

1. There must be a block of comments at start of every question's code by students; the block should contain brief description about functionality of code.
2. Comment on every function about its functionality.
3. Use understandable name of variables.
4. Proper indentation of code is essential
5. Write a C++ statement(s) for each of the following task one after the other, in the same order.
6. Make a Microsoft Word file and paste all of your C++ code with all possible screenshots of every **task outputs in MS word and do not submit .cpp file with word file**.
7. First think about statement problems and then write/draw your logic on copy.
8. After copy pencil work, code the problem statement on MS Studio C++ compiler.
9. At the end when you done your tasks, attached C++ created files in MS word file and make your submission on Microsoft teams. (Make sure your submission is completed).
10. Please submit your file in this format 19F1234_L9.
11. Do not submit your assignment after deadline.
12. Do not copy code from any source otherwise you will be penalized with negative marks.

Problem 1: | (Multiple Inheritance)

Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes. The constructors of inherited classes are called in the same order in which they are inherited. For example, in the following program, B's constructor is called before A's constructor
Write the main for following classes

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

class A
{
public:
A() { cout << "A's constructor called" << endl; }
};

class B
{
public:
B() { cout << "B's constructor called" << endl; }
};

class C: public B, public A // Note the order
{
public:
C() { cout << "C's constructor called" << endl; }
};
```

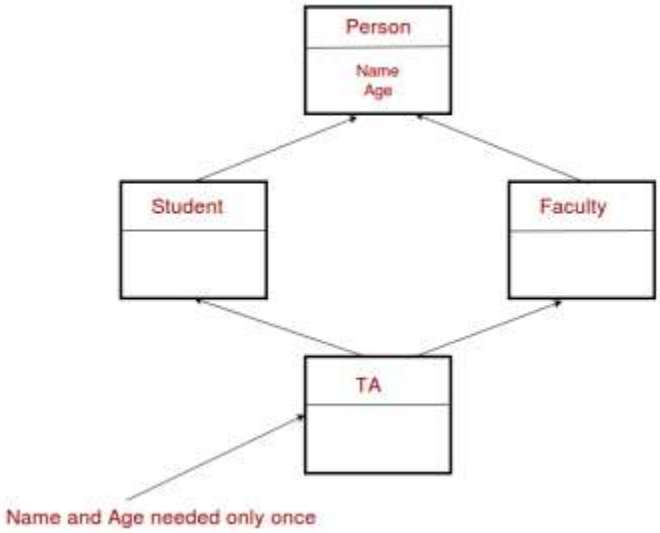
The output for the main should be like this

```
B's constructor called
A's constructor called
C's constructor called
```

Use three file structure

Problem 2: | (Multiple and Multi Level Inheritance, Diamond Problem)

The diamond problem occurs when two super classes of a class have a common base class. For example, in the following diagram, the TA class gets two copies of all attributes of Person class, this causes ambiguities.



a) What is the output for below code?

```
#include<iostream>
using namespace std;
class Person {
// Data members of person
public:
    Person(int x) { cout << "Person::Person(int ) called" << endl; }
};

class Faculty : public Person {
// data members of Faculty
public:
    Faculty(int x):Person(x) {
    cout<<"Faculty::Faculty(int ) called"<< endl;
    }
};

class Student : public Person {
// data members of Student
public:
    Student(int x):Person(x) {
        cout<<"Student::Student(int ) called"<< endl;
    }
};

class TA : public Faculty, public Student {
public:
    TA(int x):Student(x), Faculty(x) {
        cout<<"TA::TA(int ) called"<< endl;
    }
};

int main() {
    TA ta1(30);
}
```

b) Mention problem if any in **part a** and implement it correctly.
c) Write the code in 3-file structure.

Problem 3: | (Multi Level Inheritance, Function Overriding)

Write a program which can detect the sounds of Animals.
Write a class Animals having the following attributes:

1. String Name
2. Sound()

Inherit the following classes from Animals having the same attributes and behavior of base class
Override the sound function for each class

1. Cat
2. Dog
3. Tiger_Family
4. Deer

From Tiger_family class inherit the following sub classes publically: (override the sound function for each class).

1. Tiger
2. Lion

Note:

- Make instance of each class and call sound function.
- Each class animal should have its own sound.
- Program should be in 3-file structure.

Problem 4: | (Multi Level Inheritance, Function Overriding)

Write a program to computerize the billing system of a hospital.

1. Design the class doctorType, inherited from the class personType, which you already defined in previous lab, with an additional data member to store a doctor's speciality. Add appropriate constructors and member functions to initialize, access, and manipulate the data members.
2. Design the class billType with data members to store a patient's ID and a patient's hospital charges, such as pharmacy charges for medicine, doctor's fee, and room charges. Add appropriate constructors and member functions to initialize and access and manipulate the data
   members.
3. Design the class patientType, inherited from the class personType, defined in Chapter 12, with additional data members to store a patient's ID, age, date of birth, attending physician's name, the date when the patient was admitted in the hospital, and the date when the patient was discharged from the hospital. (Use the class dateType to store the date of birth, admit date, discharge date, and the class doctorType to store the attending physician's name.)
   Add appropriate constructors and member functions to initialize, access, and manipulate the data members.

Write a program to test your classes.