National University of Computer and Emerging Sciences

Lab Manual 13
Object Oriented Programming

Course Instructor	Mr. Usama Hassan
Lab Instructor (s)	Ms. Fariha Maqbool Mr. Sohaib Ahmad
Section	BSE-2C
Semester	Spring 2023

Department of Computer Science
FAST-NU, Lahore, Pakistan

Objectives

After performing this lab, students will practice:

- ✓ Virtual Destructor
- ✓ Implement multiple and virtual inheritance

Task-1: (Virtual Destructor)

Write a program to practice memory management alongside polymorphism.
You are not allowed to change function prototypes
Implement following class structure. In addition to this you are to implement destructors in all classes below to ensure dynamically allocated memory is properly deleted.

Person (Base Class)	Employee (Derived)	Student (Derived)

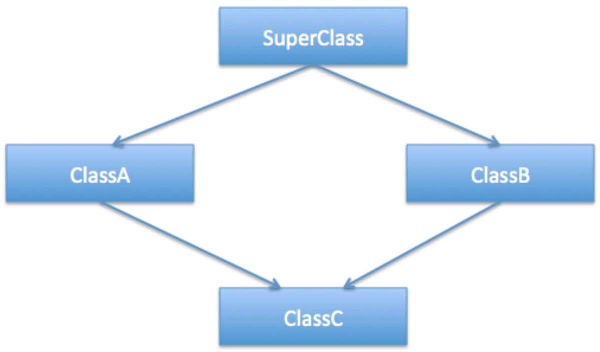
//member variables
{
Private:
String *fullName;
Int *height;
 	
Public:
 	
Person(string name,Int height) //constructor
 	
Virtual void printInfo();(1)
//this function is to print all private varaibles
 	
//destructor to be implemented alongside type of class eg cout<<"person destructor" (5)
 	
}
	//member variables
	{
	Private:
	String *departement;
	Int *ID;
	
	Public:
	
	Employee(string name,Int height,string departement,Int id) : Person(name, height)
	//constructor
	
	void printInfo();
	//this function is to print all private variables alongside type of class
	
	//destructor to be implemented alongside type of class eg cout<<"employee destructor"
	}
		//member variables
		{
		Private:
		String *schoolName;
		
		Public:
		
		Student **(string name,Int height, string SchoolName) : Person(name, height)**
		//constructor
		
		void printInfo();
		//this function is to print all private variables alongside type of class
		
		//destructor to be implemented alongside type of class eg cout<<"student destructor"
		}

Main Program:

1. Create a array of base class pointers of size 2. (2d Array)
2. Initialize each of the base class pointer with employee and student object respectively.
3. Run a loop to call printInfo on the array created.
4. Call delete operator on the array of base class to test the memory management.

Multiple Inheritance & Polymorphism

For this exercise, we are going to work on a classical multiple inheritance issue known as 'diamond problem'. Diamond problem is an ambiguity that arises when two classes A and B inherit from super class or base class, and class C also inherits from both A and B. The class hierarchy structure resembles a diamond as shown below.



1. For simplicity, create a single **.cpp file** with the following classes: **Faculty, Administrator, Teacher, & AdministratorTeacher**.
2. **Faculty** inherits **Administrator** and **Teacher**. While **AdministratorTeacher** has two parents **Administrator** and **Teacher** which represents that an **Administrator** can be a **Teacher** and vice versa

Task 1:

3. Add a **print()** method to **Faculty**, **Administrator**, and **Teacher** which displays the class name.
4. In the driver, create a pointer array of 3 **Faculty** objects.
5. Create one object for each of the remaining three classes as well and assign these three object to the **Faculty** object array.
6. Now, in a loop call the print method on the **Faculty** object array and observe the code behavior.
7. You may observe that “Faculty” is displayed on the console 3 times which is wrong.
8. To make corrections, use polymorphism. Make the **print()** method virtual and execute again.
9. This time you will encounter an error. It occurs because the **AdministratorTeacher** object shows ambiguous behavior when calling the **print()** method (It does not know which print method it should call)
10. To resolve this issue, we are going to use **virtual inheritance**. First, make the **print()** method pure virtual in **Faculty.** And add a **print()** method in the **AdministratorTeacher** class as well. Qualify or override this **print()** method by calling the **print()** of either **Teacher** or **Administrator** specifically.
11. Execute the program and report the issue in comment
12. Now, use virtual inheritance i.e. declare **Faculty** inheritance using public virtual keyword for **Teacher** and **Administrator** classes.

13. Execute the program again. This time you will observe that correct class names are displayed on console.

<u>Task 2:</u>

14. Each faculty member is assigned an id by the university. To represent this, add an **get_id()** method to both **Administrator** and **Teacher** which returns a unique integer number**.**
15. Call the **get_id()** method using the **AdministratorTeacher** object. Observe the error. Can you explain why?
16. One approach would be to qualify the **get_id()** method as we did with **print()**. But if we are making use of inheritance, there should only be one **get_id()** method in the **Faculty** class. Remove the **get_id()** method from the child classes. Execute the program and observe that it works. Can you explain why is there no ambiguity when calling the **get_id()** method from **AdministratorTeacher** object?

Note:

- Follow all the code indentation, naming conventions and code commenting guidelines.
- Make sure your program is executable.