National University of Computer & Emerging Sciences, Karachi
Spring-2025 School of Computing
Quiz No. 2
27th March 2025

Course Code: CS-1004	**Course Name: Object Oriented Programming**
Instructor Name / Names: Ms. Atiya Jokhio	
Student-ID:	

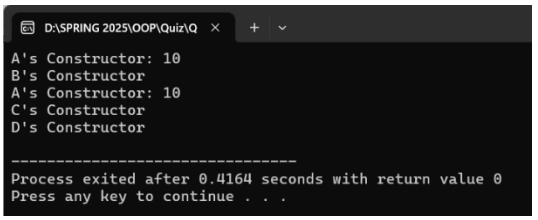
Time Allowed: 30 minutes. **Total Points**: 10

TYPE A

Question:1

a) Fix the errors [If any] in the given code below and predict the output **[2 points]**

```
#include <iostream>
using namespace std;
class A {
public:
   A(int x) { cout << "A's Constructor: " << x << endl; }
};
class B : public A {
public:
   B(int x) : A(x) { cout << "B's Constructor\n"; }
};
class C : public A {
public:
   C(int x) : A(x) { cout << "C's Constructor\n"; }
};
class D : public B, public C {
public:
   D(int x) : B(x), C(x) { cout << "D's Constructor\n"; }
};
int main() {
   D obj(10);
   return 0;
}
```



b) Encircle the correct statement **[2 points]**

If a base class constructor requires arguments, what must the derived class do?

a) Pass arguments from its own constructor
b) Ignore the base class constructor
c) Define a separate constructor in the derived class
d) Use virtual keyword

Question:2 **[6 points]**

Your task is to design a Library Management System where different types of Library Members have unique borrowing behaviors.

- **Base Class: LibraryMember**
- Protected Members: string name → Member's name, int booksBorrowed → Total books borrowed, int totalBooksBorrowed → Total books borrowed in the member's lifetime
- Methods:
  - virtual void borrowBook(int numBooks) → A general borrowing function.

- void displayTotalBorrowed() → Displays total books borrowed.

- **Derived Classes:**

- StudentMember (inherits from LibraryMember):
  - Redefines borrowBook(int numBooks) → Cannot borrow more than 5 books at a time.
- FacultyMember (inherits from LibraryMember):
  - Redefines borrowBook(int numBooks) → Can borrow up to 10 books, but gets priority access.

Note: Ensure that each member follows their borrowing limit based on their type.

.

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

// Base Class
class LibraryMember {
protected:
    string name;
    int booksBorrowed;        // Books borrowed in current session
    int totalBooksBorrowed;   // Lifetime total

public:
    LibraryMember(string memberName) {
        name = memberName;
        booksBorrowed = 0;
        totalBooksBorrowed = 0;
    }

    // Virtual method for borrowing books
    virtual void borrowBook(int numBooks) {
        booksBorrowed += numBooks;
        totalBooksBorrowed += numBooks;
        cout << name << " borrowed " << numBooks << " books." << endl;
    }

    // Display total books borrowed
    void displayTotalBorrowed() {
        cout << name << " has borrowed a total of " << totalBooksBorrowed <<
" books." << endl;
    }
};

// Derived Class - Student Member
class StudentMember : public LibraryMember {
public:
    StudentMember(string memberName) : LibraryMember(memberName) {}
```

```
    // Max 5 books
    void borrowBook(int numBooks) override {
        if (numBooks <= 5) {
            booksBorrowed += numBooks;
            totalBooksBorrowed += numBooks;
             cout << name << " (Student) borrowed " << numBooks << " books."
<< endl;
        } else {
             cout << "Error: Student members can borrow a maximum of 5 books
at a time." << endl;
        }
    }
};

// Derived Class - Faculty Member
class FacultyMember : public LibraryMember {
public:
    FacultyMember(string memberName) : LibraryMember(memberName) {}

    // Max 10 books with priority access
    void borrowBook(int numBooks) override {
        if (numBooks <= 10) {
            booksBorrowed += numBooks;
            totalBooksBorrowed += numBooks;
               cout << name << " (Faculty - Priority Access) borrowed " <<
numBooks << " books." << endl;
        } else {
            cout << "Error: Faculty members can borrow a maximum of 10 books
at a time." << endl;
        }
    }
};

// Main function to test
int main() {
    StudentMember student("Alice");
    FacultyMember faculty("Dr. Bob");

    student.borrowBook(3);      // Valid
    student.borrowBook(6);      // Should give an error

    faculty.borrowBook(7);      // Valid
    faculty.borrowBook(11);     // Should give an error

    student.displayTotalBorrowed();
    faculty.displayTotalBorrowed();
```

```
    return 0;
}
```