

The Open University of Sri Lanka
Faculty of Natural Sciences
B.Sc. Degree Programme - Level 04
Department of Computer Science
CSU4616 : Object Oriented Programming Using C++
and Java



Final Examination 2025/2026

Duration: Three Hours Only

Date: 27.11.2025

Time: 1.30 p.m.-4.30 p.m.

General Instructions

1. Read all instructions carefully before answering the questions.
2. This question paper consists of SIX questions in SEVEN pages.
3. Answer ANY FOUR questions, in the provided answer book.
4. Answer to each question should commence from a new page.
5. Involvement in any activity that is considered as an exam offense will lead to punishment.
6. Use blue or black ink to answer the questions.
7. Clearly state your Registration Number for B.Sc/B.Ed Degree Programme in your answer script.

QUESTION 01

- a. Indicate whether each of the following statements are True or False.
- i. C++ does not allow punctuation characters such as @, \$, and % within identifiers.
 - ii. C++ is a case-sensitive programming language.
 - iii. In C++, the header file `#include <iostream>` is used to work with input and output objects.

[Marks 03]

- b. Fill in the blanks with the appropriate term from the list given below.
 (Monolithic, Object Oriented analysis, Pseudo-code, Object Oriented design)
- i. The developed at the problem-solving phase is converted to a program, which can be run using a computer at the implementation phase.
 - ii. In the phase, the user requirements are translated into a set of abstractions, which can be represented as a set of classes with known relationships.
 - iii. In the phase, the initial class structure is refined, and implementation-specific design elements are added.

- iv. In programming, there are no subprograms, and the data is globally available, leaving no chance for data hiding. [Marks 04]
- c. A library wants to maintain details about the books available in its collection. Each book has a title, an author, and a price. The library needs a simple program to store and display this information using Object Oriented Programming concepts.
- i. Define a Java class named Book with the following instance variables.
title (String), author (String), price (double) [Marks 03]
 - ii. Write a parameterized constructor to initialize all instance variables of the Book class. [Marks 03]
 - iii. Write getter and setter methods for all instance variables. [Marks 06]
 - iv. Write a main() method in a separate class named LibraryTest to:
 - Create two Book objects using the parameterized constructor.
 - Display the details of each book using the getter methods.
 - The output should appear in the following format:
 Book 1: Java Programming by James Gosling - Rs.2500.00
 Book 2: Data Structures by Robert Lafore - Rs.1800.00
 [Marks 06]

QUESTION 02

- a.
- i. Describe the difference between the following access levels for member variables and methods.
Private, Protected, Public [Marks 03]
 - ii. Explain the following terms, giving each a suitable example.
Class and Object [Marks 04]
- b.
- i. List two (02) differences between C++ and Java. [Marks 02]
 - ii. Write a complete C++ program that:
 - Prompts the user to enter two integers
 - Uses a function named max(int num1, int num2) to determine and return the larger value
 - Displays the maximum of the two numbers [Marks 04]
- c. In a library system, some book details need different levels of access. Certain information can be public, while other details should be protected or private. Use suitable access modifiers when designing the Book class.
- i. Create a class named Book with the following requirements. Declare each variable using the correct access modifier.
 - The title should be accessible from anywhere.
 - The author should be accessible only within the same class.
 - The ISBN should be accessible within the same class and subclasses.

- The year should be accessible within the same package (default).
[Marks 04]
- ii. Write a constructor for the Book class to initialize all four (4) variables.
[Marks 04]
- iii. Write getter methods ONLY for the variables that require controlled access.
(You must decide which variables need getters based on their access level.)
[Marks 04]

QUESTION 03

- a.
- i. Write the syntax for declaring a class in C++.
Include the keywords, access specifiers, and member declarations in your syntax.
[Marks 02]
 - ii. What is the output of the following C++ program?

```
#include <iostream>
using namespace std;
int operate(int a, int b)
{
    return (a * b); }
float operate(float a, float b)
{
    return (a / b); }
int main()
{
    int x = 8, y = 4;
    float n = 9.0, m = 3.0;
    cout << operate(x, y) << "\t";
    cout << operate(n, m);
    return 0;
}
```

[Marks 02]
- b.
- i. Define inheritance in Object Oriented Programming and explain how it supports specialization in a class hierarchy.
[Marks 02]
 - ii. What is generalization in Object Oriented Programming, and how does it differ from specialization?
[Marks 02]
- c. A pet shop wants to maintain information about animals. All animals have a name. Cats and Dogs are types of animals. Use inheritance to model this system.
- i. Create a superclass named Animal with the following:
 - Private attribute: name (String)
 - Constructor to initialize the name
 - Getter and setter methods for name
 - Method makeSound() that returns "Some generic animal sound"
[Marks 05]

- ii. Create subclasses Cat and Dog that inherit from Animal.

For Cat:

- Private attribute breed (String)
- Constructor to initialize name and breed using the superclass constructor
- Getter and setter for breed
- Override makeSound() to return "Meow"

For Dog:

- Private attribute color (String)
- Constructor to initialize name and color using the superclass constructor
- Getter and setter for color
- Override makeSound() to return "Woof"

[Marks 06]

- iii. Write a main() method in a separate class named Zoo to:

- Create one Cat object and one Dog object
- Display all their details and sound in a single sentence each.
For Cat: "Cat Milky (Breed: Persian) says Meow"
For Dog: "Dog Buddy (Color: Brown) says Woof"

[Marks 06]

QUESTION 04

- a. Indicate whether each of the following statements are True or False.

- i. Private, static, and final methods cannot be overridden.
- ii. If a class is extending an abstract class or implementing an interface then it has to override all the abstract methods unless the class itself is an abstract class.
- iii. Method overloading is a runtime (dynamic) polymorphism.
- iv. Operator overloading is a concept which enables the programmer to assign multiple meanings to operators.

[Marks 04]

- b. Write a C++ program to implement a class called Rectangle with the following specifications.

- Data members (private): height, width
- Member functions:

To assign values to height and width

To calculate and return the area of the rectangle

Formula: Area = height × width

[Marks 04]

c. A school wants to maintain information about shapes in its drawing application. Every shape has a name and a color. Some shapes, like Rectangle, also have specific attributes such as width and height. You are required to use inheritance and method overriding to display shape details.

i. Create a superclass named Shape with the following.

- Private attributes: name (String), color (String)
- Constructor to initialize these attributes
- Method display() to print "Shape: <name>, Color: <color>"

[Marks 06]

ii. Create a subclass named Rectangle that inherits from Shape with the following.

- Private attributes: width and height (double)
- Constructor to initialize name, color, width, and height (call superclass constructor for name and color)
- Override the display() method to:
 - Call the superclass display() method
 - Print "Width: <width>, Height: <height>"

[Marks 06]

iii. Write a main() method in a class ShapeTest to:

- Create one Shape object and one Rectangle object
- Call the display() method for both objects

[Marks 05]

QUESTION 05

a.

i. Differentiate between a constructor and a destructor in C++ by stating three (03) differences. [Marks 03]

ii. What is operator overloading in C++? [Marks 03]

b.

i. What is a nested class in Java? [Marks 02]

ii. What will be the output of the below program?

```
class Library {
    static String name = "City Library";
    class Book {
        String name = "Java Programming";
        void printName() {
            System.out.println(name);
        }
    }
}

public class MainClass {
    public static void main(String[] args) {
        Library lib = new Library();
        System.out.println(lib.name);
        Library.Book book = lib.new Book();
    }
}
```

```

        System.out.println(book.name);
        book.printName();
    } }

```

[Marks 03]

- c. A music app wants to play different types of audio. Every audio type must implement the ability to play and stop the audio.

- i. Create an interface named Playable with two methods: play() and stop().

[Marks 04]

- ii. Create a class Song that implements Playable. Implement the methods to print.
 "Playing Song" for play()
 "Stopping Song" for stop()

[Marks 03]

- iii. Create another class Podcast that implements Playable. Implement the methods to print:

"Playing Podcast" for play()
 "Stopping Podcast" for stop()

[Marks 03]

- iv. Write a main() method in a class AudioTest to:
 Create one Song object and one Podcast object
 Call play() and stop() for both objects

[Marks 04]

QUESTION 06

- a. Define the following two states in the Java thread life cycle.

Newborn State, Runnable State

[Marks 04]

- b. Consider the following Java program that demonstrates basic thread creation and execution.

```

class WorkerThread extends Thread {
    public void run() {
        System.out.println("Worker thread is running");
        System.out.println("Worker thread has completed.");
    }
}

public class MainClass {
    public static void main(String[] args) {
        WorkerThread t1 = new WorkerThread();
        t1.start();
        System.out.println("Main thread is running");
    }
}

```

Answer the following:

- i. Describe the states that the thread t1 passes through during the execution of this program.

[Marks 04]

- ii. Explain the purpose of the start() method and how it differs from directly calling the run() method. [Marks 04]
- iii. Write the possible output of the program. [Marks 03]
- c. A transport company wants to develop a simple C++ program to manage vehicle information. You are required to use inheritance to model the system.
- i. Create a base class named Vehicle with the following.
- One private data member: num_wheels
 - A default constructor and a parameterized constructor (that allows setting the number of wheels when creating a Car object)
 - A public method showVehicle() that displays the number of wheels
- [Marks 03]
- ii. Create a derived class named Car, which the following.
- Inherits publicly from Vehicle
 - Has one private data member: passengers
 - Has a parameterized constructor that initializes both passengers and the inherited num_wheels
 - Includes a show() method that:
Calls showVehicle() and Displays the number of passengers.
- [Marks 04]
- iii. Write a main() function to:
- Create one Car object with 4 wheels and 5 passengers.
 - Display all details using the show() method.
- [Marks 03]

*****All Rights Reserved*****