Solution with scheme-Model Answer

Prof. Priti Badar									
Internal Assessment Test 2 – Jan 2026									
Sub	**Object Oriented Programming with JAVA**					Sub code	**BCS306A**	Branch	**CSE**
Date	06/01/2026	Duration	**90 mins**	Max Marks	**50**	Sem /Sec	**III Sem A/B/C**		**OBE**

	Answer any FIVE FULL Questions	**MARKS**	**CO**	**RBT**
1	a. Define package. Explain the steps involved in creating a user-defined package with an example. b. Demonstrate the working of a nested try block with an example.	**[6]** **[4]**	**CO4** **CO4**	**L2** **L3**
2.	a. Write a program that contains one method that will throw an IllegalAccessException and use proper exception handles so that the exception should be printed. b. Define an exception. What are the key terms used in exception handling? Explain.	**[6]** **[4]**	**CO4** **CO4**	**L3** **L2**
3.	a. What is the need of synchronization? Explain with an example how synchronization is implemented in JAVA. b. What do you mean by a thread? Explain the different ways of creating threads.	**[6]** **[4]**	**CO5** **CO5**	**L2** **L2**
4.	a. Discuss values() and value Of() methods in Enumerations with suitable examples. b. Summarize the type wrappers supported in Java.	**[6]** **[4]**	**CO5** **CO5**	**L2** **L2**
5.	a. Explain isAlive() and join() methods with program and output. b. Explain interthread communication using wait(), notify(), and notifyAll() with example.	**[6]** **[4]**	**CO3** **CO5**	**L2** **L2**
6.	a. Explain how an interface is used to achieve multiple Inheritances in Java. b. Develop a JAVA program to create an outer class with a function display. Create another class inside the outer class named inner with a function called display and call the two functions in the main class.	**[6]** **[4]**	**CO3** **CO3**	**L2** **L3**

1.

Q1 (a) [6 Marks] [2]

Define Package & Steps to Create User-Defined Package

Definition [2]

A package in Java is a namespace that groups related classes and interfaces together.
It avoids naming conflicts and provides access protection. [2]

Need of Package

- **Organizes large programs**
- **Avoids class name conflicts**
- **Provides security (access specifiers)**
- **Improves code reusability**

Steps to Create User-Defined Package

1. **Declare package using:**
2. **package packagename;**
3. **Save file with class name.**
4. **Compile using:**
5. **javac -d . FileName.java**
6. **Import package in another program:**
7. **import packagename.ClassName;**

Example
Step 1: Create Package

```
package mypack;

public class Demo {
   public void show() {
      System.out.println("User Defined Package");
   }
}
```

Compile:

```
javac -d . Demo.java
```

Step 2: Use Package

```
import mypack.Demo;

class Test {
   public static void main(String args[]) {
      Demo obj = new Demo();
      obj.show();
   }
}
```

Q1 (b) [4 Marks]

Nested Try Block

Definition 1M

A try block inside another try block is called Nested try block.

Program 3M

```
class NestedTryDemo {
   public static void main(String args[]) {
      try {
         int a = 10/2;

         try {
            int b = 10/0;
         }
         catch(ArithmeticException e) {
            System.out.println("Inner Exception: " + e);
         }

      }
```

```
        catch(Exception e) {
            System.out.println("Outer Exception");
        }
    }
}
```

Output

Inner Exception: java.lang.ArithmeticException: / by zero

Q2 (a) [6 Marks]

Program to Throw IllegalAccessException

```
class ExceptionDemo {

    static void check() throws IllegalAccessException {
        throw new IllegalAccessException("Access Denied");
    }

    public static void main(String args[]) {
        try {
            check();
        }
        catch(IllegalAccessException e) {
            System.out.println("Exception Caught: " + e);
        }
    }
}
```

2M

4M

Output

Exception Caught: java.lang.IllegalAccessException: Access Denied

---

Q2 (b) [4 Marks]

Define Exception & Key Terms

Exception

An exception is an abnormal condition that disrupts normal program execution.

2M

Key Terms

Term	Meaning
try	Block containing risky code
catch	Handles exception
finally	Always executed
throw	Explicitly throws exception
throws	Declares exception

2M

Q3 (a) [6 Marks]

Need of Synchronization

Need

When multiple threads access shared data, synchronization prevents:

- **Data inconsistency**
- **Race condition**

2M

Example

```
class Table {
    synchronized void printTable(int n) {
        for(int i=1;i<=5;i++) {
            System.out.println(n*i);
        }
    }
}

class Thread1 extends Thread {
    Table t;
    Thread1(Table t){ this.t=t; }
    public void run(){ t.printTable(5); }
}
```

2M

```
class SyncDemo {
   public static void main(String args[]) {
      Table obj = new Table();
      new Thread1(obj).start();
      new Thread1(obj).start();
   }
}
```

2M

Q3 (b) [4 Marks]
Thread & Ways to Create Thread
Thread
A thread is a lightweight sub-process that runs concurrently.

2M

Ways to Create Thread

1. **By Extending Thread Class**

```
class MyThread extends Thread {
   public void run() {
      System.out.println("Thread running");
   }
}
```

2. **By Implementing Runnable**

```
class MyThread implements Runnable {
   public void run() {
      System.out.println("Thread running");
   }
}
```

2 M

Q4 (a) [6 Marks]
values() and valueOf() in Enum

2M

```
enum Day { MON, TUE, WED }

class EnumDemo {
   public static void main(String args[]) {

      for(Day d : Day.values()) {
         System.out.println(d);
      }

      Day d1 = Day.valueOf("MON");
      System.out.println("Selected: " + d1);
   }
}
```

2M

2M

Output

```
MON
TUE
WED
Selected: MON
```

Q4 (b) [4 Marks]
Wrapper Classes in Java
Wrapper classes convert primitive data types into objects.

2M

Primitive	Wrapper
int	Integer
char	Character
double	Double
float	Float
boolean	Boolean
byte	Byte
short	Short

2M

2.	**long Long**	
	Q5 (a) [6 Marks] **isAlive() and join()** **class Demo extends Thread {** **public void run() {** **System.out.println("Thread is running");** **}** **public static void main(String args[]) throws Exception {** **Demo t = new Demo();** **System.out.println("Before start: " + t.isAlive());** **t.start();** **t.join();** **System.out.println("After completion: " + t.isAlive());** **}** **}** **Output** **Before start: false** **Thread is running** **After completion: false**	2M 2M 2M
	Q5 (b) [4 Marks] **Interthread Communication** **Uses:** • **wait()** • **notify()** • **notifyAll()**	2M
	Example **class Customer {** **int amount = 1000;** **synchronized void withdraw(int amt) {** **if(amount < amt) {** **try { wait(); } catch(Exception e) {}** **}** **amount -= amt;** **System.out.println("Withdraw Completed");** **}** **synchronized void deposit(int amt) {** **amount += amt;** **System.out.println("Deposit Completed");** **notify();** **}** **}**	2M
	Q6 (a) [6 Marks] **Interface for Multiple Inheritance** **interface A {** **void show();** **}** **interface B {** **void display();** **}** **class C implements A, B {** **public void show() {** **System.out.println("Show Method");** **}** **public void display() {**	2M 4M

```
        System.out.println("Display Method");
    }
}
```

Q6 (b) [4 Marks]

Outer & Inner Class Program

```
class Outer {
    void display() {
        System.out.println("Outer Class Display");
    }

    class Inner {
        void display() {
            System.out.println("Inner Class Display");
        }
    }
}

class Main {
    public static void main(String args[]) {
        Outer obj = new Outer();
        obj.display();

        Outer.Inner in = obj.new Inner();
        in.display();
    }
}
```

2M

2M