PROGRAMMING IN JAVA

Assignment 6

TYPE OF QUESTION: MCQ

Number of questions: 10 Total marks**: 10 × 1 = 10**

QUESTION 1:

What is the output of the following program?

```
public class Nptel{

  public static void main(String[] args) {
    try {
      int a = 5 / 0;
    } catch (Exception e) {
        catch (ArithmeticException a) {

       }
    }
    System.out.println("Programming In Java");
  }
}
```

a. **"Programming In Java"**

b. **Run time error**

c. **Compile time error**

d. **ArithmeticException**

Correct Answer:

c. **Compile time error**

Detailed Solution:

This first handler catches exceptions of type `Exception`; therefore, it catches any exception, including `ArithmeticException`. The second handler could never be reached. This code will not compile.

QUESTION 2:

What is the output of the following program?

```
public class Nptel extends Thread {

  public void run() {
    for (int i = 1; i < 5; i++) {
      System.out.print(i++ + " ");
    }
  }

  public static void main(String args[]) {
    Nptel t1 = new Nptel();
    t1.run();
  }
}
```

a. **1 3**

b. **1 2 3 4**

c. **Runtime error**

d. **1 2**

Correct Answer:

a. **1 3**

Detailed Solution:

Inside the `for` loop, the increment operation `i++` is used, which is a post-increment operation. This means that the increment happens after the value of `i` is used. But since this increment operation is inside the print statement, it will increment `i` by 1 right after printing it, effectively skipping every other number. So, the output of this code will be: “1 3”

This is because the `i++` operation inside the `System.out.print` statement causes `i` to increment by an additional 1 each time through the loop, resulting in only the odd numbers between 1 and 3 being printed.

QUESTION 3:

For the program given below, what will be the output after its execution?

```
public class Nptel extends Thread {

  public static void main(String[] args) {
    Thread thread = Thread.currentThread();
    System.out.println(thread.activeCount());
  }
}
```

a. 0

b. true

c. 1

d. false

Correct Answer:

c. 1

Detailed Solution:

`java.lang.Thread.activeCount()` returns an estimate of the number of active threads in the current thread's thread group and its subgroups which is 1 in this case since it's the only thread running the program.

QUESTION 4:

Which of the following is a correct constructor for a thread object?

a. `Thread(Runnable a, String str);`

b. `Thread(Runnable a, int priority);`

c. `Thread(Runnable a, ThreadGroup t);`

d. `Thread(int priority);`

Correct Answer:

a. `Thread(Runnable a, String str);`

Detailed Solution:

`Thread(Runnable a, String str)` creates a new `Thread` object. The others are not valid constructors to create a `Thread` object.

QUESTION 5:

What is the output of the following program?

```
class Nptel extends Thread {

  public void run() {
    System.out.println("Running");
  }
}

public class ThreadTest {

  public static void main(String args[]) throws InterruptedException {
    Runnable r = new Nptel();
    Thread myThread = new Thread(r);
    myThread.start();
  }
}
```

a. **Compiler Error**

b. **“Running”**

c. **Runtime Exception**

d. **No output, but no error**

Correct Answer:

b. **“Running”**

Detailed Solution:

The class `Thread` implements the `Runnable` interface, so the assignment is valid. Also, you can create a new thread object by passing a `Runnable` reference to a `Thread` constructor, is also valid. Hence, the program will compile without errors and print “Running” in the console.

QUESTION 6:

How many threads does the following program run on?

```
public class ThreadExtended extends Thread {

  public void run() {
    System.out.println("\nThread is running now\n");
  }

  public static void main(String[] args) {
    ThreadExtended threadE = new ThreadExtended();

    threadE.start();
  }
}
```

a. 0

b. 1

c. 2

d. 3

Correct Answer:

c. 2

Detailed Solution:

There are 2 threads. Main program is also run as a thread. And, program has created one child thread. Hence, total 2 threads are there in the program.

QUESTION 7:

In the following java program, what is the NAME of the thread?

```
class Nptel extends Thread{

    public static void main(String args[]) {
      Thread t = Thread.currentThread();
      System.out.println(t);
    }
}
```

a. **thread**

b. **main**

c. **system**

d. **None of the above**

Correct Answer:

b. **main**

Detailed Solution:

The name of the `Tread` `t` is `main` as it's the only tread that is running. It is set to `currentThread()` which is `main`.

QUESTION 8:

Which of the following line(s) of code is suitable to START a thread at #1?

```
class Nptel extends Thread {

  public static void main(String args[]) {
    /* Missing code */
    _____________________ // #1
  }

  public void run() {}
}
```

a. **Thread t = new Thread(Nptel);**

b. **Thread t = new Thread(Nptel);**
   t.start();

c. **Nptel run = new Nptel();**
   Thread t = new Thread(run);
   t.start();

d. **Thread t = new Thread();**
   Nptel.run();

Correct Answer:

c. **Nptel run = new Nptel();**
   Thread t = new Thread(run);
   t.start();

Detailed Solution:

An instance of the `Nptel` class is created. This class should implement the `Runnable` interface, which means it must have a `run` method. This `run` method contains the code that will be executed in the new thread. (`Nptel run = new Nptel();`)
A new `Thread` object is created, passing the `Nptel` instance (`run`) to the `Thread` constructor. This tells the `Thread` that it should execute the `run` method of the `Nptel` instance in the new thread. (`Thread t = new Thread(run);`)
The `start` method of the `Thread` instance is called. This method starts the new thread and calls the `run` method of the `Nptel` instance in that new thread. (`t.start();`) So, the `run` method of the `Nptel` instance will be executed in a new thread, separate from the main thread of the application.

QUESTION 9:

What is the name of the priority of this Thread in this program?

```
class Nptel extends Thread{

  public static void main(String args[]) {
    Thread t = Thread.currentThread();
    System.out.println(t.getPriority());
  }
}
```

a. 1

b. 4

c. 0

d. 5

Correct Answer:

d. 5

Detailed Solution:

The default priority given to a thread is 5.

QUESTION 10:

What does I/O stand for in Java?

a. **Input/Output**

b. **Inheritance/Overriding**

c. **Integer/Object**

d. **Iteration/Observation**

Correct Answer:

a. **Input/Output**

Detailed Solution:

I/O stands for Input/Output in Java. It refers to the process of reading data from input sources and writing data to output destinations.