Multi-Threading
Workbook

Answer the Following

1. What is multithreading?

2. What are the two ways to create the thread?

3. What is the signature of the constructor of a thread class?

4. What are the methods used for Inter Thread communication and in which class these methods are defined?

5. What is difference between wait() and sleep() method?

6. What is the purpose of synchronized block?

7. What are the states of thread?

8. What is meant by daemon thread? In java runtime, what is it's role?

9. When should we interrupt a thread?

10. What is the difference between preemptive scheduling and time slicing?

11. What is deadlock?

State whether the following are True/False

1. It is possible to start a thread twice. [ ]
2. We can make the user thread as daemon thread if thread is started. [ ]
3. The word synchronized can be used with only a method. [ ]
4. The suspend() method is used to teriminate a thread. [ ]
5. Garbage collector thread belongs to high priority. [ ]
6. Among wait(), notify(), notifyall() the wait() method only throws IOException. [ ]
7. wait(), notify(), notifyall() are defined as final and can be called only from with in a synchronized method. [ ]

Multiple Choice Questions

1. Which of these method can be used to make the main thread to be executed last among all the threads?

   (a) stop()
   (b) sleep()
   (c) join()
   (d) call()

2. Which of these method is used to find out that a thread is still running or not?

   (a) run()
   (b) alive()
   (c) isAlive()
   (d) checkRun()

3. What is the default value of priority variable MIN_PRIORITY AND MAX_PRIORITY?

   (a) 1 and 10
   (b) 0 and 1
   (c) 1 and 256
   (d) 0 and 256

4. Which of these method waits for the thread to treminate?

   (a) stop()
   (b) join()
   (c) isAlive()
   (d) sleep()

5. Which of these interface is implemented by Thread class?

   (a) Runnable
   (b) Connections
   (c) Set
   (d) MapConnections

6. Which of these method is used to begin the execution of a thread?

   (a) run()
   (b) start()
   (c) runThread()
   (d) startThread()

7. Which of these statement is incorrect?

(a) A thread can be formed by implementing Runnable interface only.

(b) A thread can be formed by a class that extends Thread class.

(c) start() method is used to begin execution of the thread.

(d) run() method is used to begin execution of a thread before start() method in special cases.

8. Which of these method of Thread class is used to find out the priority given to a thread?

   (a) get()

   (b) ThreadPriority()

   (c) getPriority()

   (d) getThreadPriority()

9. Which of these are types of multitasking?

   (a) Process based

   (b) Thread based

   (c) Process and Thread based

   (d) None

Exercises

- Write the expected output, or compiler errors if any, for each of the following programs in the box provided below each program.
- Then execute the programs and check your answers.
- Then answer the questions given below.

Program 1

```
public class Test implements Runnable {
    public static void main(String[] args) {
        Test test = new Test();
        Thread thread = new Thread(test);
        thread.start();
        thread.join();
        System.out.print(‘‘main’’);
    }
    public void run() {
        System.out.print(‘‘run’’);
    }
}
```

Q1: What will be the output of the program?

Program 2

```
public class Test{
    public static void main(String[] args) {
        Painter painter1 = new Painter();
        painter1.start();
        Painter painter2 = new Painter();
        painter2.start();
    }
}
class Painter implements Runnable {
    public void run() {
        System.out.println(‘‘we are painting’’);
    }
}
```

Q1: How many times the statement “we are painting” would be printed in this program?

Program 3

```
public class Test implements Runnable {
    Integer id;
    public static void main(String[] args) {
        new Thread(new Test()).start();
        new Thread(new Test()).start();
    }
    public void run() {
        press(id);
    }
    synchronized void press(Integer id) {
        System.out.print(id.intValue());
        System.out.print((++id).intValue());
    }
}
```

Q1: What is the possible result of compiling and running the code?

Program 4

```
class Swimmer implements Runnable {
     String name ;
     Swimmer(String name) {
          this.name = name;
     }
     public void run() {
          new Test().swim(name);
     }
}
public class Test {
     public void swim(String name) {
          System.out.print(name);
          System.out.print(name);
     }
     public static void main(String[] args) {
          new Thread(new Swimmer(”Tom”)).start();
          new Thread(new Swimmer(”Hanks”)).start();
     }
}
```

Q1: What are the possible outputs of running this program once as it is, and second with marking swim() synchronized?

Program 5

```
public class Tester extends Thread {
     public void run() {
          System.out.print(‘‘run’’);
     }
     public static void main(String[] args) {
          Tester thread = new Tester();
          new Thread(thread).start();
          new Thread(thread).start();
     }
}
```

Q1: What is the result of compiling and running the code?

Program 6

```
class Test {
    public static void main(String [] args) {
        printAll(args);
    }
    public static void printAll(String[] lines) {
        for(int i = 0; i < lines.length; i++) {
            System.out.println(lines[i]);
            Thread.currentThread().sleep(1000);
        }
    }
}
```

Q1: What will be the output of the above code?

Program 7

```
class MyThread extends Thread {
    public static void main(String [] args) {
        MyThread t = new MyThread();
        Thread x = new Thread(t);
        x.start();
    }
    public void run() {
        for(int i = 0; i < 3; ++i) {
            System.out.print(i + ”..”);
        }
    }
}
```

Q1: What will be the output of the above code?