

NAME :- Aaditya Khot

PRN :- 21610051

SUBJECT :- JAVA

ASSIGNMENT :- 7

BATCH :- S-5

Question 1:- Can we call the run() method instead of start()?

Yes, it is possible to call the run() method instead of start() in Java multithreading, but it would not create a new thread and would not execute the code concurrently. Instead, it would simply execute the code in the same thread that called the run() method, like a regular method call.

Question 2:- Explain the use of word Synchronized

When a method or a block of code is marked as synchronized, only one thread can execute that code at any given time. Other threads must wait until the executing thread releases the lock on the object that the synchronized block is synchronized on.

Question 3:-

```
public class three {  
    public static void main(String[] args) {  
        Thread currentThread = Thread.currentThread();  
        System.out.println("Current thread: " +  
currentThread.getName());  
        System.out.println("ID: " + currentThread.getId());  
        System.out.println("Priority: " +  
currentThread.getPriority());  
    }  
}
```

```

        System.out.println("State: " +
currentThread.getState());

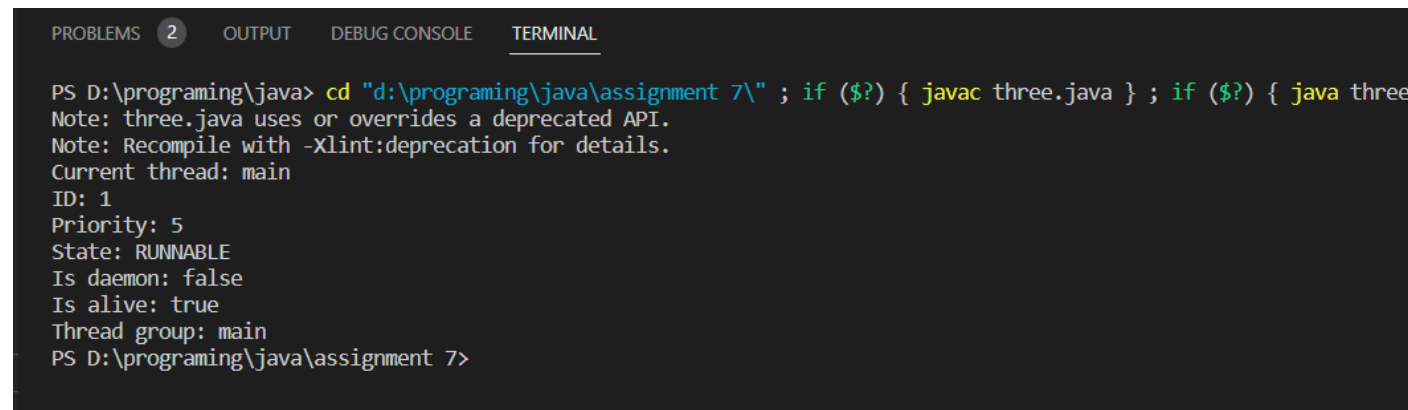
        System.out.println("Is daemon: " +
currentThread.isDaemon());

        System.out.println("Is alive: " +
currentThread.isAlive());

        System.out.println("Thread group: " +
currentThread.getThreadGroup().getName());

    }
}

```



```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL
PS D:\programing\java> cd "d:\programing\java\assignment 7\" ; if ($?) { javac three.java } ; if ($?) { java three
Note: three.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Current thread: main
ID: 1
Priority: 5
State: RUNNABLE
Is daemon: false
Is alive: true
Thread group: main
PS D:\programing\java\assignment 7>

```

Question 4:- Create a thread using Thread class and Runnable class.

```

public class four extends Thread {

    @Override

    public void run() {

```

```

        for(int i=0;i<10;i++){
            System.out.println("Thread is running");
        }
    }

    public static void main(String[] args) {
        four thread = new four();
        thread.start();
        for(int i=0;i<10;i++){
            System.out.println("Main Thread is running");
        }
    }
}

```

```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL
PS D:\programing\java> cd "d:\programing\java\assignment 7\" ; if ($?) { javac four.java } ;
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
PS D:\programing\java\assignment 7>

```

Using Runnable interface :-

```
public class four implements Runnable {
```

```
    @Override
```

```
    public void run() {
```

```
        for(int i=0;i<10;i++){
```

```
            System.out.println("Thread is running");
```

```
        }
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        four runnable = new four();
```

```
        Thread thread = new Thread(runnable);
```

```
        thread.start();
```

```
        for(int i=0;i<10;i++){
```

```
            System.out.println("Main Thread is running");
```

```
        }
```

```
    }
```

}

```
PS D:\programing\java> cd "d:\programing\java\assignment 7\" ; if ($?) { javac four.
Main Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
Main Thread is running
PS D:\programing\java\assignment 7>
```

Question 5:- Write a program for thread communication and synchronization.

```
class Customer{
    int amount=10000;

    synchronized void withdraw(int amount){
        System.out.println("going to withdraw...");

        if(this.amount<amount){
```

```
System.out.println("Less balance; waiting for  
deposit...");
```

```
try {wait();} catch(Exception e) {}  
}
```

```
this.amount-=amount;
```

```
System.out.println("withdraw completed...");  
}
```

```
synchronized void deposit(int amount){
```

```
System.out.println("going to deposit...");
```

```
this.amount+=amount;
```

```
System.out.println("deposit completed... ");
```

```
notify();
```

```
}
```

```
}
```

```
class five{
```

```
public static void main(String args[]){
```

```
final Customer c=new Customer();
```

```
new Thread(){
```

```
public void run(){c.withdraw(15000);}
```

```
}.start();  
new Thread(){  
public void run(){c.deposit(10000);}  
}.start();  
}}
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\programing\java> cd "d:\programing\java\assignment 7\" ; if ($?) {  
going to withdraw...  
Less balance; waiting for deposit...  
going to deposit...  
deposit completed...  
withdraw completed...  
PS D:\programing\java\assignment 7>
```