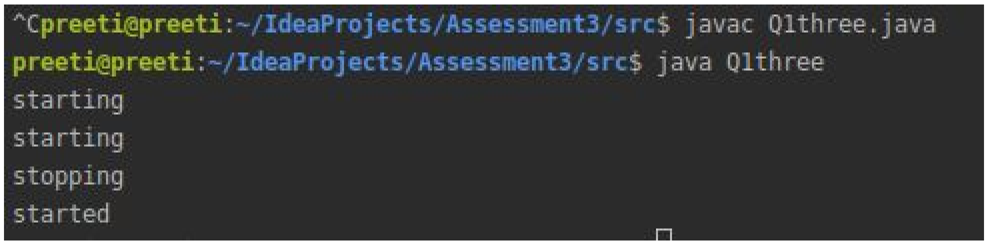
EXERCISE

1. **Write a program to demonstrate the use of volatile keyword.**

   CODE:

```
public class Q1three
{
    private static volatile boolean running=false;

    public static void main(String[] args) throws Exception
    {
        new Thread(new Runnable() {                                   //new thread
            public void run() {
                while (!running) {                                    //wait
                }
                System.out.println("starting");

                while (running) {                                     //wait
                }
                System.out.println("started");
            }
        }).start();

        Thread.sleep(1000);
        System.out.println("starting");

        running=true;

        Thread.sleep(1000);
        System.out.println("stopping");

        running=false;
    }
}
```

   OUTPUT:

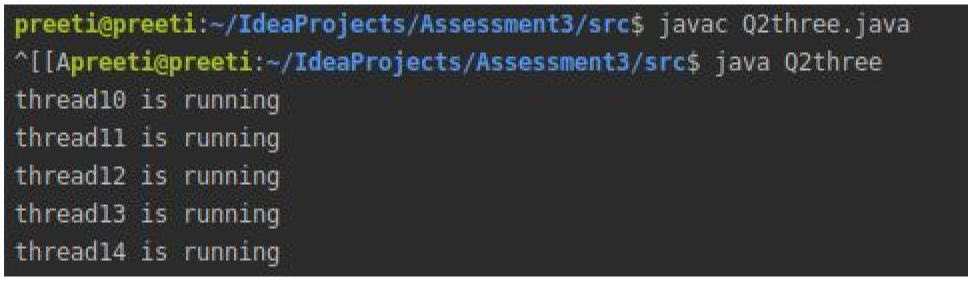


```
^Cpreeti@preeti:~/IdeaProjects/Assessment3/src$ javac Q1three.java
preeti@preeti:~/IdeaProjects/Assessment3/src$ java Q1three
starting
starting
stopping
started
```

2. **Write a program to create a thread using Thread class and Runnable interface each.**

```
import java.lang.*;
class Hello extends Thread
{
    public void run()
    {
        try
        {
            System.out.println("thread"+Thread.currentThread().getId()+" is running");
        }
        catch(Exception e)
        {
            System.out.println("exception is caught");
        }
    }
}
public class Q2three
{
    public static void main(String[] args)
    {
//          int n=5;
        for(int i=0;i<5;i++)
        {
            Hello ob=new Hello();
            ob.start();
        }

    }
}
```

OUTPUT:



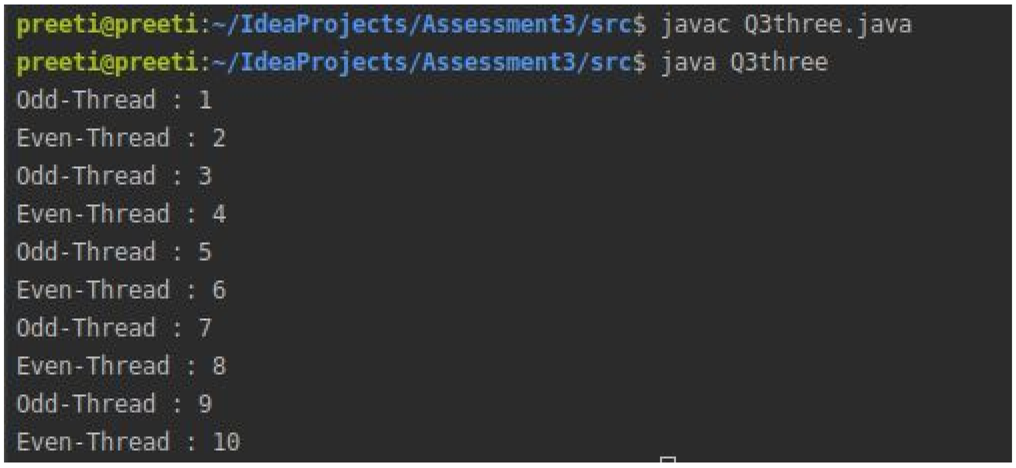
```
preeti@preeti:~/IdeaProjects/Assessment3/src$ javac Q2three.java
^[[Apreeti@preeti:~/IdeaProjects/Assessment3/src$ java Q2three
thread10 is running
thread11 is running
thread12 is running
thread13 is running
thread14 is running
```

3. **Write a program to create a Thread pool of 2 threads where one Thread will print even numbers and other will print odd numbers.**

   CODE:

```
class OddThread extends Thread
{
    int limit;
    sharedPrinter printer;
    public OddThread(int limit, sharedPrinter printer)
    {
        this.limit = limit;
        this.printer = printer;
    }
    @Override
    public void run()
    {
        int oddNumber = 1;
        while (oddNumber <= limit)
        {
            printer.printOdd(oddNumber);
            oddNumber = oddNumber + 2;
        }
    }
}

class EvenThread extends Thread
{
    int limit;
    sharedPrinter printer;
    public EvenThread(int limit, sharedPrinter printer)
    {
        this.limit = limit;
        this.printer = printer;
    }
    @Override
    public void run()
    {
        int evenNumber = 2;
        while (evenNumber <= limit)
        {
            printer.printEven(evenNumber);
            evenNumber = evenNumber + 2;
        }
    }
}
class sharedPrinter
{

    boolean isOddPrinted = false;

    synchronized void printOdd(int number)
    {
        while (isOddPrinted)
        {
            try
            {
                wait();
            }
            catch (InterruptedException e)
            {
                e.printStackTrace();
            }
        }
        System.out.println(Thread.currentThread().getName()+" : "+number);
        isOddPrinted = true;
        try
        {
            Thread.sleep(1000);
        }
        catch (InterruptedException e)
        {
            e.printStackTrace();
        }
        notify();
    }

    synchronized void printEven(int number)
    {
        while (! isOddPrinted)
        {
            try
            {
                wait();
            }
            catch (InterruptedException e)
            {
                e.printStackTrace();
            }
        }
        System.out.println(Thread.currentThread().getName()+" : "+number);
        isOddPrinted = false;
        try
        {
            Thread.sleep(1000);
        }
        catch (InterruptedException e)
        {
            e.printStackTrace();
        }
        notify();
    }
}
//Main Class
public class Q3three
{
    public static void main(String[] args)
    {
        sharedPrinter printer = new sharedPrinter();
        OddThread oddThread = new OddThread(10, printer);
        oddThread.setName("Odd-Thread");
        EvenThread evenThread = new EvenThread(10, printer);
        evenThread.setName("Even-Thread");
        oddThread.start();
        evenThread.start();
    }
}
```

OUTPUT:



```
preeti@preeti:~/IdeaProjects/Assessment3/src$ javac Q3three.java
preeti@preeti:~/IdeaProjects/Assessment3/src$ java Q3three
Odd-Thread : 1
Even-Thread : 2
Odd-Thread : 3
Even-Thread : 4
Odd-Thread : 5
Even-Thread : 6
Odd-Thread : 7
Even-Thread : 8
Odd-Thread : 9
Even-Thread : 10
```