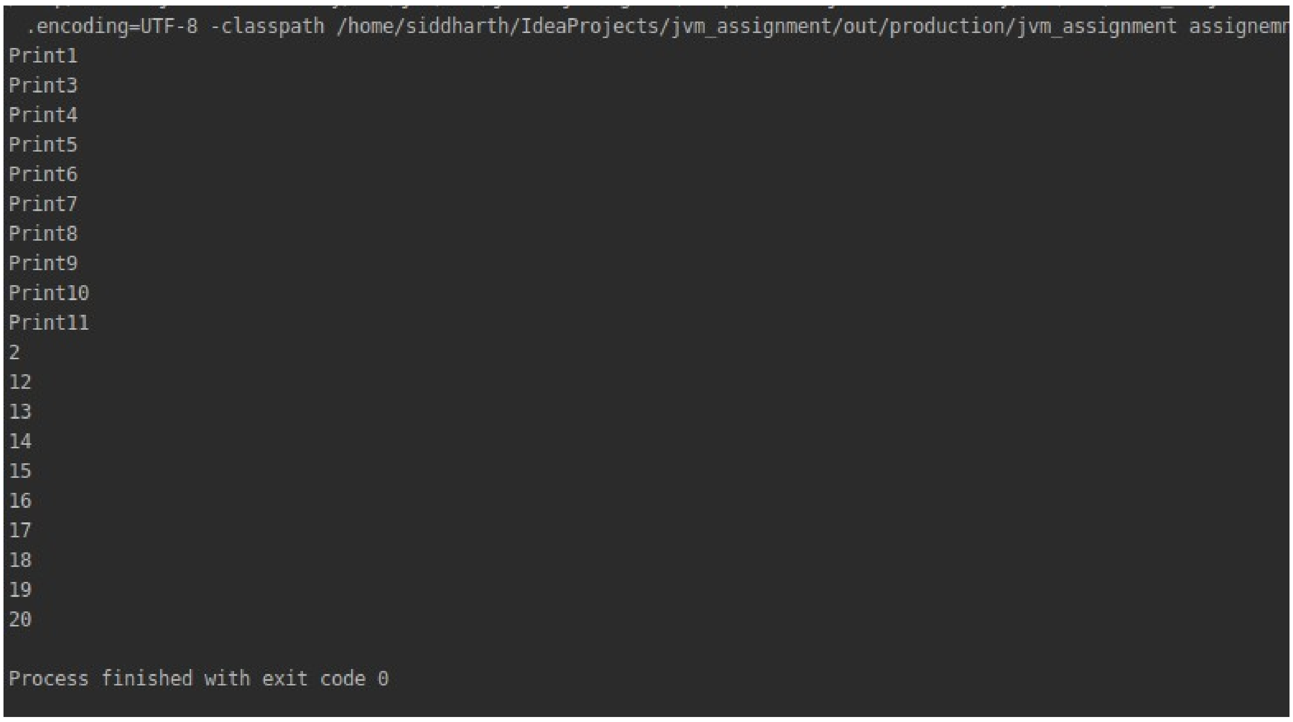
q-1 Write a programe do to demonstrate the use of volatile keyword

```
package assignemnt3;
class xyz extends Thread {
    private volatile int count=0;
    public void run()
    {
       for(int i=0;i<10;i++)
       {
           try {
               count++;
               System.out.println(""+count);
               Thread.sleep(2000);
           }
           catch(InterruptedException e)
           {
           }
       }
    }
    void print()
    {
        for(int i=0;i<10;i++)
        {
            count ++;
            System.out.println("Print"+count);
        }
    }
}
public class as3q1 {
    public static void main(String[] args) {
        xyz x=new xyz();
        x.start();
        x.print();
    }
}
```



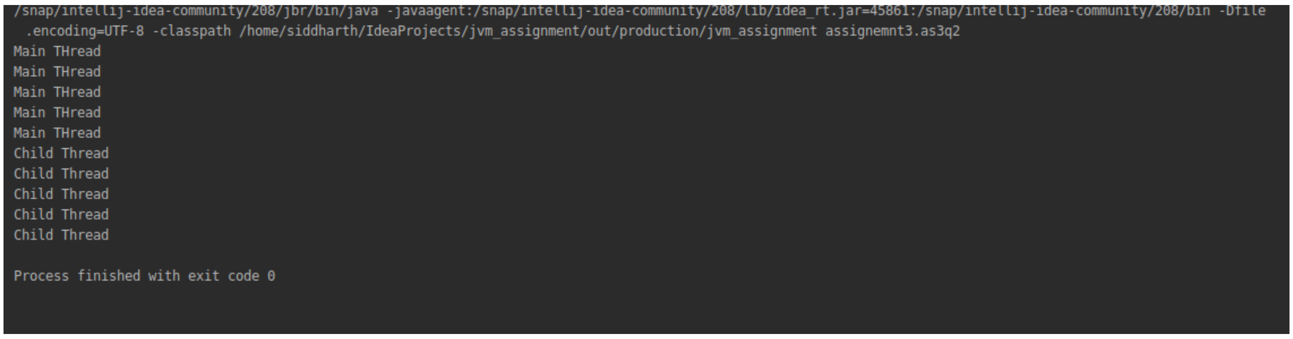
```
 .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemn
Print1
Print3
Print4
Print5
Print6
Print7
Print8
Print9
Print10
Print11
2
12
13
14
15
16
17
18
19
20

Process finished with exit code 0
```

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

<u>Extending thread class</u>

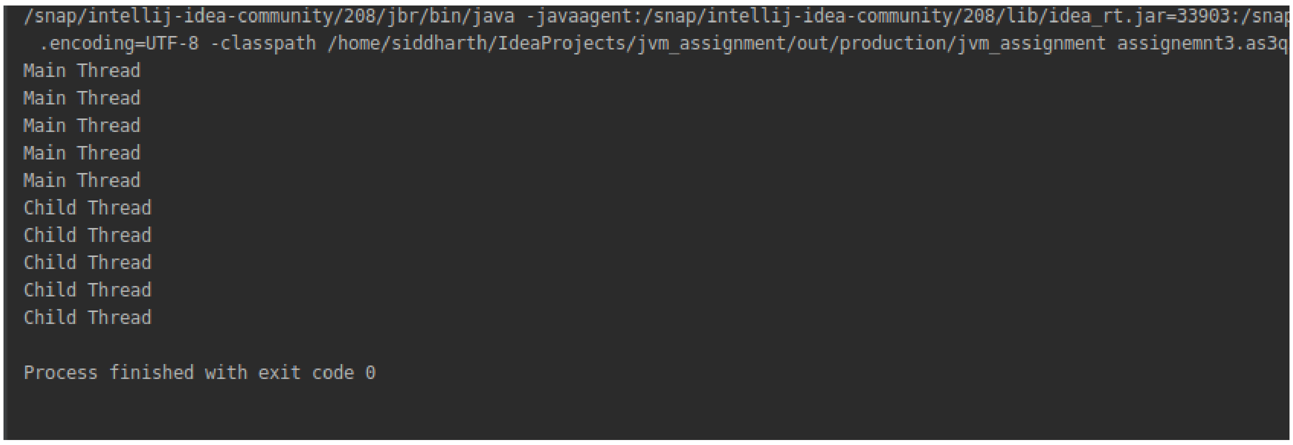
```
class  MyThread extends Thread{
   public void run() {
```

```java
        for (int i = 0; i < 5; i++) {
            System.out.println("Child Thread");
        }
    }
}
public class as3q2 {
    public static void main(String[] args) {
     MyThread k=new MyThread();
     k.start();
     for(int i=0;i<5;i++)
     {
         System.out.println("Main THread");
     }
    }
}
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-community/208/lib/idea_rt.jar=45861:/snap/intellij-idea-community/208/bin -Dfile
  .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemnt3.as3q2
Main THread
Main THread
Main THread
Main THread
Main THread
Child Thread
Child Thread
Child Thread
Child Thread
Child Thread

Process finished with exit code 0
```

```java
package assignemnt3;
class  MyRunnable  implements Runnable{
    public void run() {
        for (int i = 0; i < 5; i++) {
            System.out.println("Child Thread");
        }
    }
}
public class as3q2ii {
    public static void main(String[] args) {
        MyRunnable r=new MyRunnable();
        Thread t=new Thread(r);
        t.start();
        for(int i=0;i<5;i++)
        {
            System.out.println("Main Thread");
        }
    }
}
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-community/208/lib/idea_rt.jar=33903:/snap
  .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemnt3.as3q
Main Thread
Main Thread
Main Thread
Main Thread
Main Thread
Child Thread
Child Thread
Child Thread
Child Thread
Child Thread

Process finished with exit code 0
```

q-3 Write a program using synchronization block and synchronization method

```
package assignemnt3;
class  display
{
     public synchronized void wish(String name)
     {
          for(int i=0;i<10;i++)
          {
               System.out.print("Good Morning ");
               try{
                    Thread.sleep(2000);
               }
               catch(InterruptedException e)
               {
               }
               System.out.println(name);
          }
     }
}
class thmy extends Thread{
     display obj;
     String name;
     thmy(display obj,String username)
     {
          this.obj=obj;
         name=username;
     }
     public void run()
     {
          obj.wish(name);
     }
}
public class as3q5 {
     public static void main(String[] args) {
          display m=new display();
          thmy t=new thmy(m,"Siddharth");
          thmy t2=new thmy(m,"Mohan");
          t.start();
          t2.start();
     }
}
```

Synchronized block

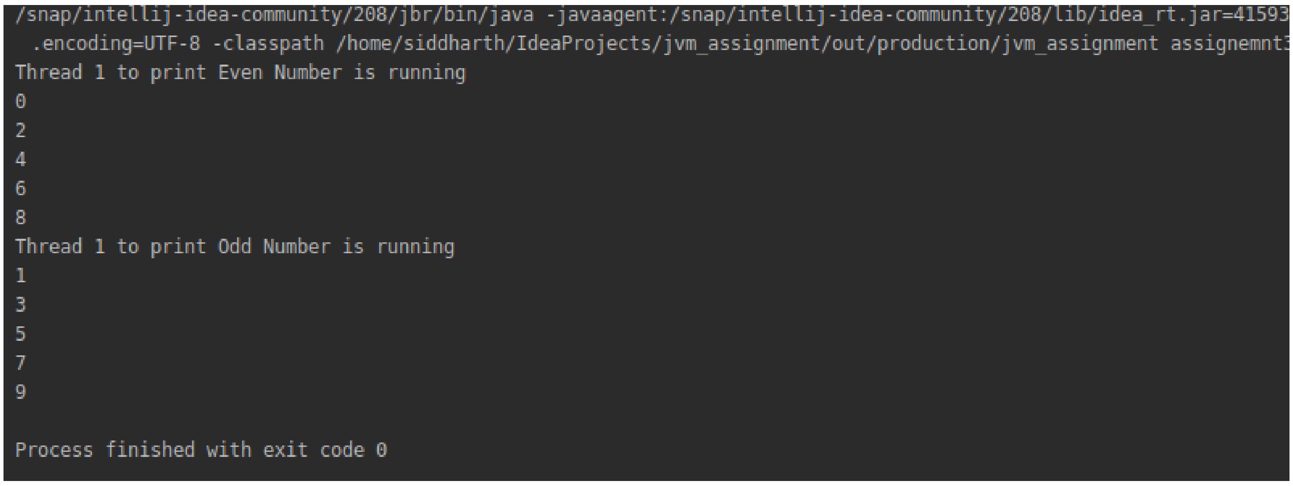
```
package assignemnt3;
class Display{
     public void wish(String name)
     {
          synchronized (this)
          {
               for(int i=0;i<5;i++)
               {
                    System.out.print("Good Morning: ");
                    try{
                         Thread.sleep(2000);
                    }
                    catch (InterruptedException e)
                    {
                    }
```

```
                  System.out.println(name);
              }
          }
     }
}
class mmthread extends Thread {
   Display d;
   String uname;
   mmthread(Display d,String un)
   {
        this.d=d;
        uname=un;
   }
   public void run()
   {
        d.wish(uname);
   }
}
class SyncDemo{
     public static void main(String[] args) {
          Display m=new Display();
          mmthread mt=new mmthread(m,"Siddharth");
          mmthread mt2=new mmthread(m,"Micky");
          mt.start();
          mt2.start();
     }
}
```

q-4 Write a program to create a Thread pool of 2 threads where one Thread will print even numbers and other will print odd numbers

```
package assignemnt3;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class as3q4{
     public static void main(String[] args) {
          ExecutorService executorService = Executors.newSingleThreadScheduledExecutor();
          executorService.submit(new Runnable() {
               @Override
               public void run() {
                    System.out.println("Thread 1 to print Even Number is running");
                    for(int i=0;i<10;i+=2){
                         System.out.println(i);
                         try {
                              Thread.sleep(100);
                         } catch (InterruptedException e) {
                              e.printStackTrace();
                         }
                    }
               }
          });
          executorService.submit(new Runnable() {
               @Override
               public void run() {
                    System.out.println("Thread 1 to print Odd Number is running");
                    for(int i=1;i<10;i+=2){
                         System.out.println(i);
                         try {
                              Thread.sleep(100);
                         } catch (InterruptedException e) {
                              e.printStackTrace();
                         }
                    }
```

```
                }
            });
            executorService.shutdown();
        }
}
/*
import javax.annotation.processing.Processor;
import java.util.concurrent.Executor;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class Processor12 implements Runnable{
    private int id;
    public Processor12(int id){
        this.id=id;
    }
    public void run()
    {
        System.out.println("Starting"+id);
        try{
            Thread.sleep(5000);
        }catch(InterruptedException e)
        {
            e.printStackTrace();
        }
    }
}
public class as3q4 {
    public static void main(String[] args) {
        ExecutorService executor= Executors.newFixedThreadPool(2);
         for(int i=0;i<5;i++)
         {
             executor.submit(new Processor12(i));
         }
         executor.shutdown();
        System.out.println("All task submited");
        executor.awaitTermination(i,time);
    }
}
*/
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-community/208/lib/idea_rt.jar=41593
  .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemnt3
Thread 1 to print Even Number is running
0
2
4
6
8
Thread 1 to print Odd Number is running
1
3
5
7
9

Process finished with exit code 0
```

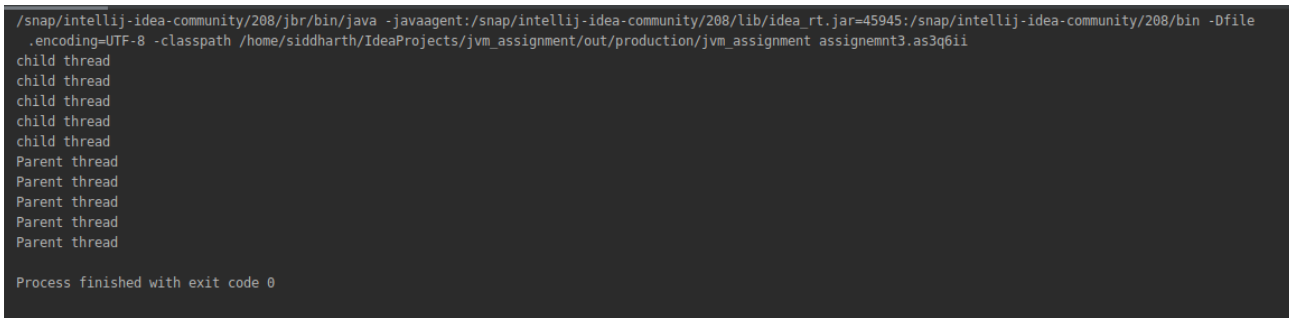
q-5 Write a program to demonstrate wait and notify methods

```
package assignemnt3;
class ThreadB extends Thread{
    int total;
    public void run(){
        synchronized(this){
```

```java
            for(int i=0; i<100 ; i++){
                total += i;
            }
            notify();
        }
    }
}
public class as3q11 {
    public static void main(String[] args){
        ThreadB b = new ThreadB();
        b.start();
        synchronized(b) {
            try {
                System.out.println("Waiting for b to complete...");
                b.wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Total is: " + b.total);
        }
    }
}
```

q-6 Write a program to demonstrate sleep and join methods.

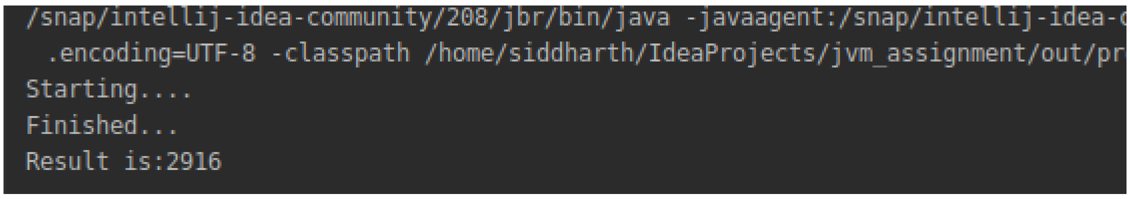
```java
package assignemnt3;
class Mthread extends Thread{
        public void run ()
        {
            for(int i=0;i<5;i++)
            {
                System.out.println("child thread");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException ex) {
                }
            }
        }
}
public class as3q6ii {
    public static void main(String[] args) throws InterruptedException {
        Mthread t=new Mthread();
        t.start();
        t.join();
        for(int i=0;i<5;i++)
        {
          System.out.println("Parent thread");
        }
    }
}
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-community/208/lib/idea_rt.jar=45945:/snap/intellij-idea-community/208/bin -Dfile
 .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemnt3.as3q6ii
child thread
child thread
child thread
child thread
child thread
Parent thread
Parent thread
Parent thread
Parent thread
Parent thread

Process finished with exit code 0
```

q-7 Run a task with the help of callable and store it's result in the Future.

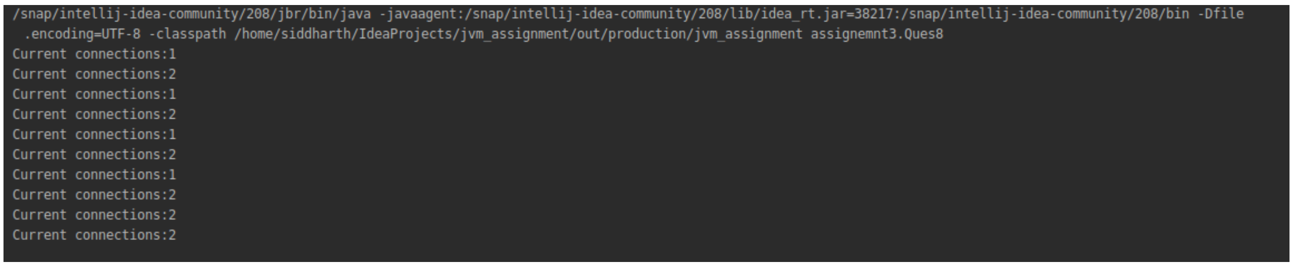
```java
package assignemnt3;
import java.util.Random;
import java.util.concurrent.*;
class q7 {
    public static void main(String[] args) {
        ExecutorService executor = Executors.newCachedThreadPool();
        Future<Integer> future = executor.submit(new Callable<Integer>() {
            @Override
            public Integer call() throws Exception {
                Random random = new Random();
                int duration = random.nextInt(4000);
                System.out.println("Starting....");
                Thread.sleep(duration);
                System.out.println("Finished...");
                return duration;
            }
        });
        executor.shutdown();
        try{
            System.out.println("Result is:"+ future.get());
        }
        catch (InterruptedException | ExecutionException e)
        {
            e.printStackTrace();
        }
    }
}
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-c
  .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/pr
Starting....
Finished...
Result is:2916
```

q-8 Write a program to demonstrate the use of semaphore

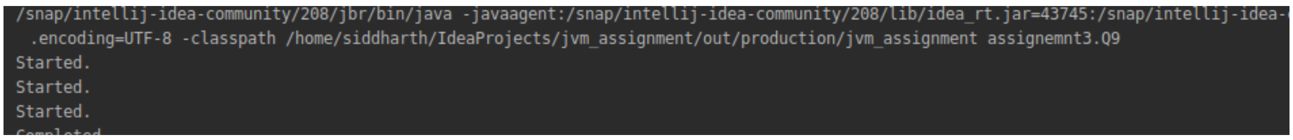
```java
package assignemnt3;
import java.util.concurrent.*;
class connection{
    private Semaphore sem = new Semaphore(2);
    private int connections=0;
    public void connect() throws InterruptedException {
        sem.acquire();
        synchronized (this)
        {
            connections++;
            System.out.println("Current connections:"+ connections);
        }
        Thread.sleep(2000);
        synchronized (this)
        {
            connections--;
        }
        sem.release();
    }
}
class Ques8 {
    public static void main(String[] args) throws Exception{
        ExecutorService executor = Executors.newCachedThreadPool();
        connection obj = new connection();
        for (int i=0;i<10;i++)
        {
            executor.submit(new Runnable() {
                @Override
                public void run() {
                    try {
                        obj.connect();
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }
            });
        }
        executor.shutdown();
        executor.awaitTermination(1, TimeUnit.DAYS);
    }
}
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-community/208/lib/idea_rt.jar=38217:/snap/intellij-idea-community/208/bin -Dfile
  .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemnt3.Ques8
Current connections:1
Current connections:2
Current connections:1
Current connections:2
Current connections:1
Current connections:2
Current connections:1
Current connections:2
Current connections:2
Current connections:2
```

q-9 Write a program to demonstrate the use of CountDownLatch

```java
package assignemnt3;
import java.util.concurrent.CountDownLatch;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class Processor implements Runnable
{
    private CountDownLatch latch;
    public Processor(CountDownLatch latch)
    {
        this.latch=latch;
    }
    @Override
    public void run() {
        System.out.println("Started.");
        try {
            Thread.sleep(3000);
        } catch (InterruptedException ex) {
            ex.printStackTrace();
        }
        latch.countDown();
    }
}
 class Q9 {
    public static void main(String[] args) {
        CountDownLatch latch = new CountDownLatch(3);
        ExecutorService executor = Executors.newFixedThreadPool(3);
        for(int i=0;i<3;i++)
        {
            executor.submit(new Processor(latch));
        }
        try{
            latch.await();
        }
        catch (InterruptedException ex)
        {
            ex.printStackTrace();
        }
        System.out.println("Completed.........");
    }
}
```



```
/snap/intellij-idea-community/208/jbr/bin/java -javaagent:/snap/intellij-idea-community/208/lib/idea_rt.jar=43745:/snap/intellij-idea-
  .encoding=UTF-8 -classpath /home/siddharth/IdeaProjects/jvm_assignment/out/production/jvm_assignment assignemnt3.Q9
Started.
Started.
Started.
```

q-10 Write a program which creates deadlock between 2 threads

```
public class as3q10 {
    public static void main(String[] args) throws InterruptedException{
        Thread.currentThread().join();
        System.out.println("Situation of deadlock");
    }
}
```