WEEK-10

06/2/24

+ Demonstrate Inter process Communication & deadlock

```
public class Deadlockexample {
        final sharedresource sharedresource = new
    SharedResource ();

    Thread process 1 = new Thread(() -> {
      try {
         sharedresource.method1();
      } catch (InterruptedException e) {
           e.printStackTrace ();
      }
});

Thread process 2 = new thread(() -> {
     try {
          sharedResources.method2();
     } catch (InterruptedException e) {
          e.printStackTrace ();
     }
 });
     process 1.start();
     process 2.start();
} }
```

```
class sharedresources {
    private final object lock1 = new object();
    private final object lock2 = new object();

    public void method1() throws InterruptedException {
        synchronized (lock1) {
            System.out.println(x: "Method 1 acquired lock1");
        }
    }
}


public void method2() throws InterruptedException {
        synchronize (lock2) {
            System.out.println("Method 2 acquired lock2");
            Thread.sleep(1000);

            Synchronized (lock1) {
                System.out.println("Method 2 acquired lock 2");
            }
        }
    }
}
```

Output:-

```
Method 1 acquired lock 1
Method 2 acquired lock 2
```

classmate
Date
Page

Output :- Main thread entered A.foo
Racing Thread entered B.bar
Main Thread trying to call B.last()
Inside A.last
Back in Main Thread
Racing Thread trying to call A.last()
Inside A.last
Back in other thread.

[illegible]
06.02.24

2-255-405

classmate
Date
Page

/2/2024

Program -10

+ Inter process Communication

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get()
     {
         while (!ValueSet)
         try {
                System.out.println("\n Consumer waiting\n");
         wait();
         } catch (InterruptedException e)
           {
             System.out.println("InterruptionException
               caught");
            }
           System.out.println("Got :"+n);
           value set = false;
             System.out.println("\n Intimate Producer\n");
           notify ();
           return n;
          }
```

```
Synchronized void put(int n) {
 while (value set)
  try {
      System.out.println("\n Producer waiting \n");
   wait();
   }
   catch (InterruptedException e)
   {
     System.out.println("Interrupted Exception
                         caught");
    }
    this.n = n;
    value set = true;
    System.out.println("\n Intimate Consumer \n");
    notify();
    }
    }

class Producer implements Runnable
{
   Q q;
   producer (Q q)
   {
    This.q = q;
new Thread (this, "Producer").start();
}
public void run()
 {
  int i=0;
  while (i<15) {
   q.put(i++);
  }
 }
}

class consumer implements Runnable {
    {
     Q q;
    Consumer (Q q) {
      this.q = q;
    new Thread (this, "Consumer").start();
    }
    public void run()
    int i=0;
    while (i<15) {
    int r=q.get();
    System.out.println("Consumed: "+r);
      i++;
    }
}
```

```
class PCfixed {
    public static void main(String args[])
    {
      Q q = new Q();

      new producer(q);
      new consumer(q);
      System.out.println("Press Control-C to stop.");
    }
}
```

Output :-

```
Put : 1
Got : 1
Put : 2
Got : 2
Put : 3
Got : 3
Put : 4
Got : 4
:
:
```

[illegible]
13.02.24