Synchronized:

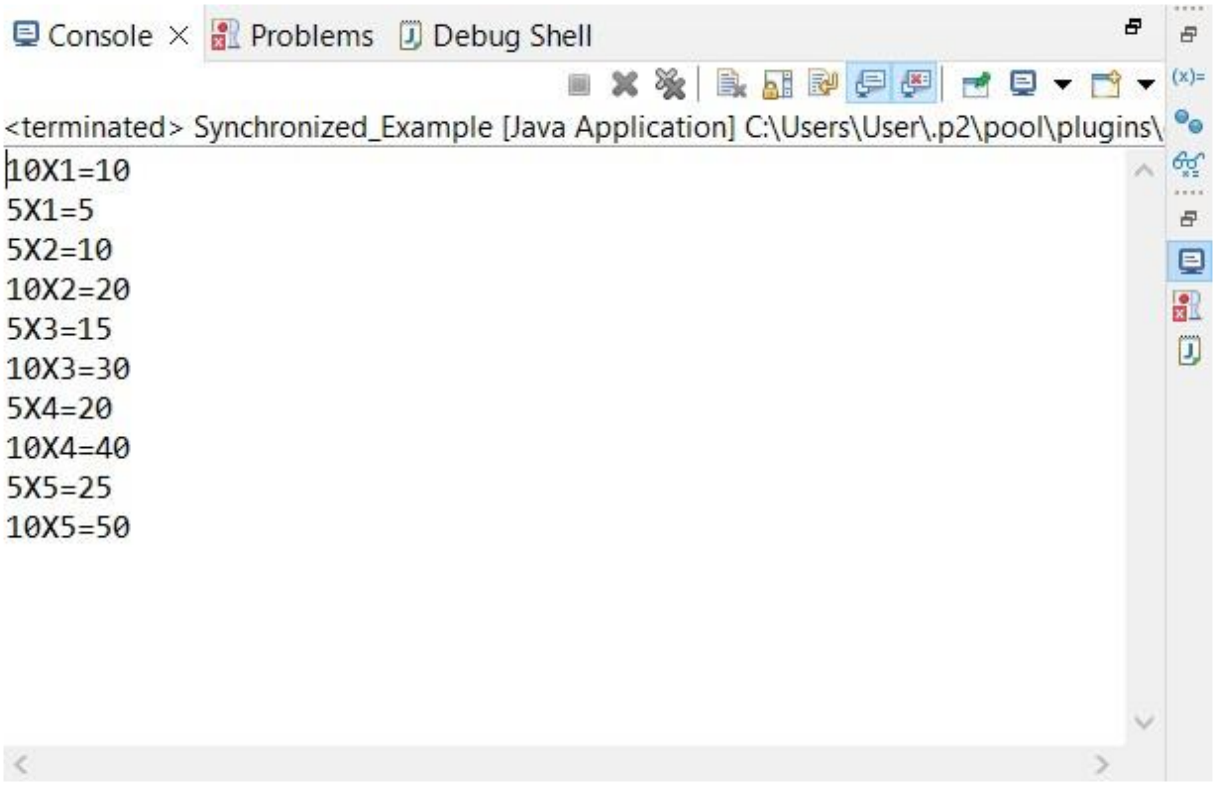
Here i am first multiplying 5 with upto 5 and than i am printing 10 upto 5 times

Example 1:

```
        package Threads;

class d {
        public  void add(int l) {

                for (int i = 1; i <= 5; i++) {
                        System.out.println(l + "X" + i + "=" + (i * l));
                        try {
                                Thread.sleep(1000);
                        } catch (InterruptedException e) {
                        }}}
}

class d1 extends Thread {
        d d1;

        d1(d s) {
                d1 = s;
        }

        public void run() {
                d1.add(5);
        }
}

class d2 extends Thread {
        d d12;

        d2(d s) {
                d12 = s;
        }

        public void run() {
                d12.add(10);
        }
}

public class Synchronized_Example {

        public static void main(String[] args) throws Exception {

                d df = new d();
                d1 dg = new d1(df);
                d2 dh = new d2(df);
                dg.start();
                dh.start();
        }
}
```

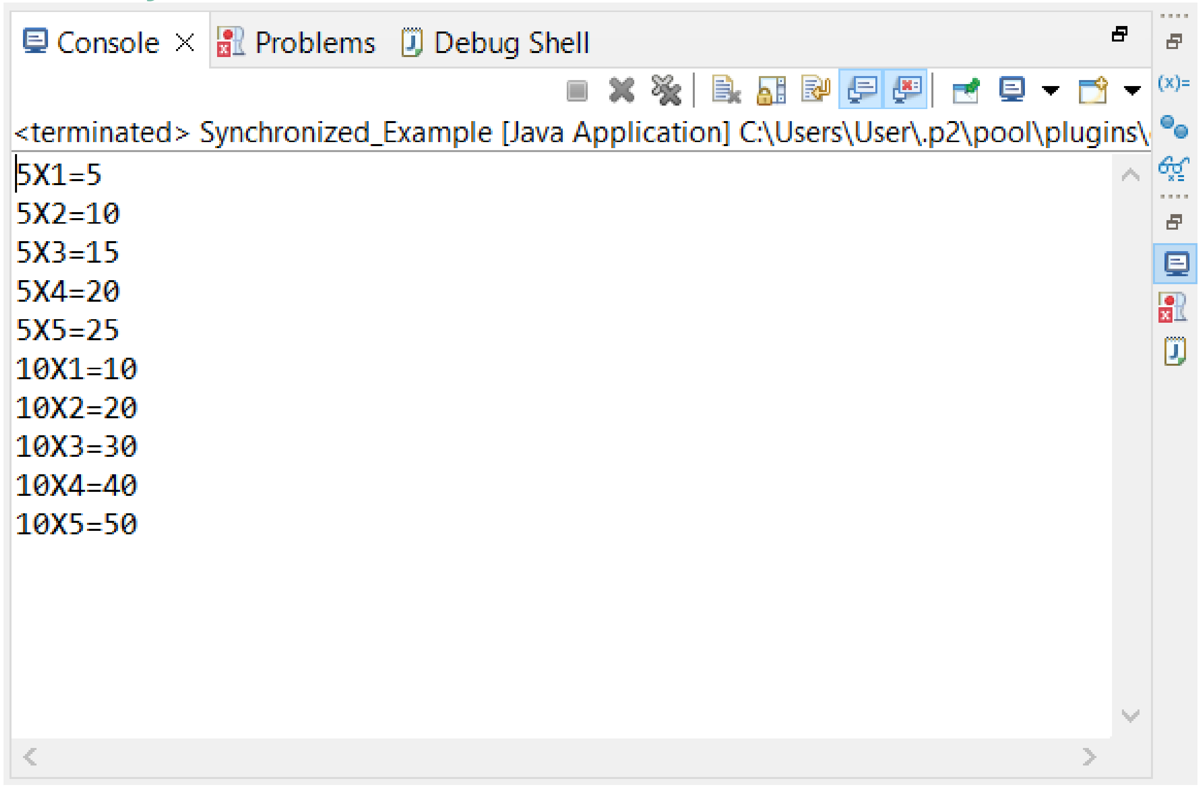
Output:

Before Synchronized:



```
<terminated> Synchronized_Example [Java Application] C:\Users\User\.p2\pool\plugins\
10X1=10
5X1=5
5X2=10
10X2=20
5X3=15
10X3=30
5X4=20
10X4=40
5X5=25
10X5=50
```

After synchronized:



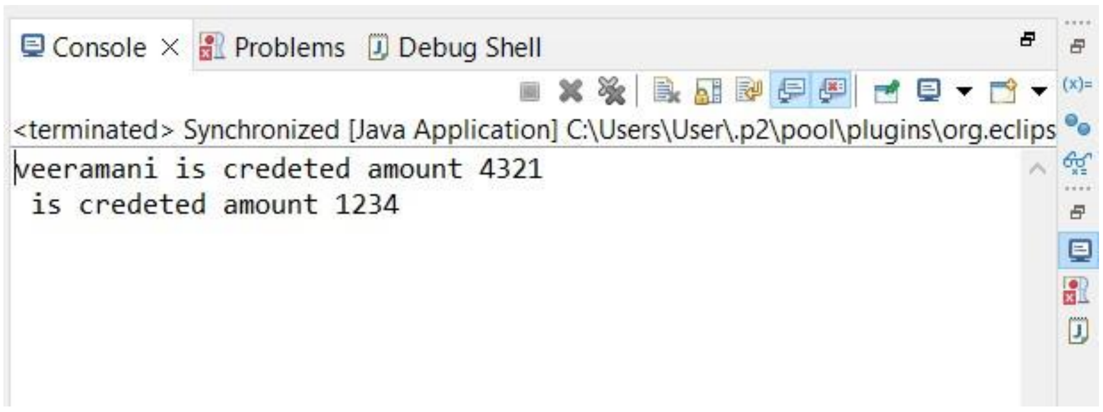
```
<terminated> Synchronized_Example [Java Application] C:\Users\User\.p2\pool\plugins\
5X1=5
5X2=10
5X3=15
5X4=20
5X5=25
10X1=10
10X2=20
10X3=30
10X4=40
10X5=50
```

Example 2:

here i am passing two values from two methods and calling both methods at same time using two threads.

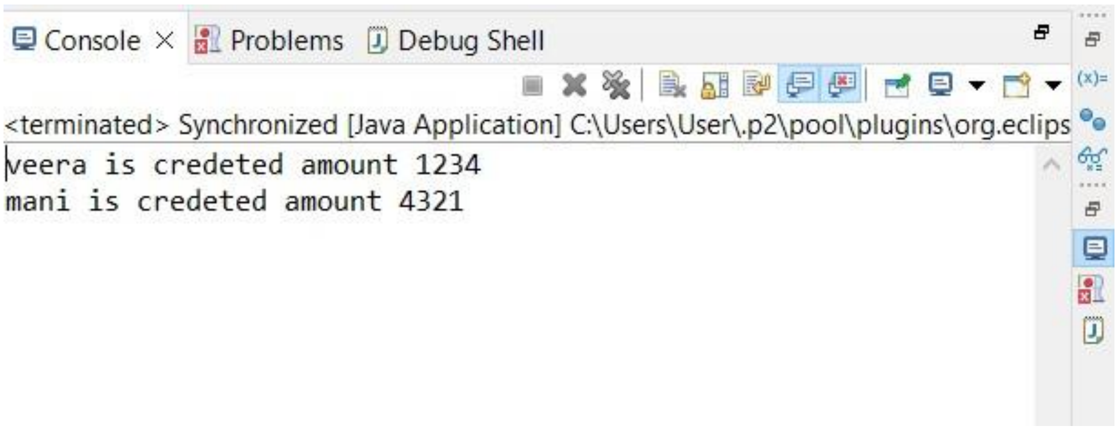
```
class  user{

        void exc(String name,String num) {
                System.out.print(name);
                System.out.println(" is credeted amount "+num);
         }
}
class user1 extends Thread{
        user u;
        user1(user u){
                this.u=u;
        }
        public void run() {
                u.exc("veera","1234");
        }
}
class user2 extends Thread{
        user u;
        user2(user u){
                this.u=u;
        }
        public void run() {
                u.exc("mani","4321");
        }
}
public class Synchronized {

        public static void main(String[] args)throws Exception {

                user u=new user();
                user1 u1=new user1(u);
                user2 u2=new user2(u);
                u1.start();
                u2.start();

        }

}
```

Before Synchronized:



```
<terminated> Synchronized [Java Application] C:\Users\User\.p2\pool\plugins\org.eclips
veeramani is credeted amount 4321
 is credeted amount 1234
```

After synchronized:



```
<terminated> Synchronized [Java Application] C:\Users\User\.p2\pool\plugins\org.eclips
veera is credeted amount 1234
mani is credeted amount 4321
```

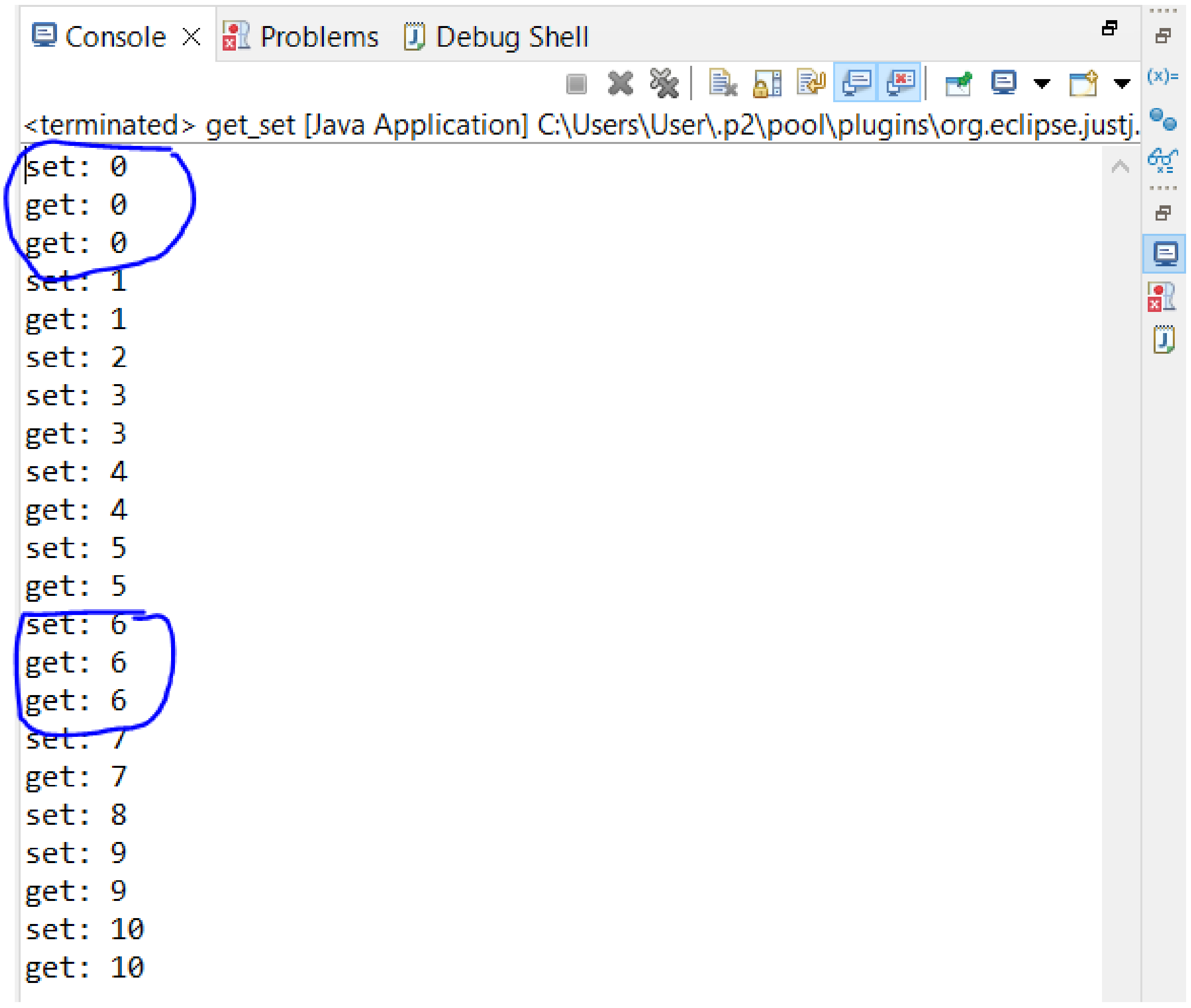
Inter-Thread Communication:

here i am using two methods one method with set value and another method will get that value at same time, here i am using two threads one thread set value and another thread get that value....

```
package Threads;

class c{

	boolean b=false;
	int num;
	public synchronized void set(int k) {
		//while(b) {try{wait();}catch(Exception e) {}}
		num=k;
		System.out.println("set: "+num);
		b=true;
		//notify();

	}
	public synchronized void get() {
	//	while(!b) { try{wait();}catch(Exception e) {}}
		System.out.println("get: "+num);
		b=false;
		//notify();
	}
}
class p extends Thread{
	c c1;
	p(c c1){this.c1=c1;}
	 public void run() {
		int i=0;
		while(i<=10) {
			c1.set(i++);
			//i++;
			try {Thread.sleep(1000);}catch(Exception e) {}
		}
	}
}
class p1 extends Thread{
	c c1;
	p1(c c1){this.c1=c1;}
	public void run() {
		int i=0;
		while(i<=10) {
			c1.get();
			i++;
			try {Thread.sleep(1000);}catch(Exception e) {}
		}
	}
}
public class get_set {

	public static void main(String[] args)throws Exception {
		 c c11=new c();
		 p p1=new p(c11);
```

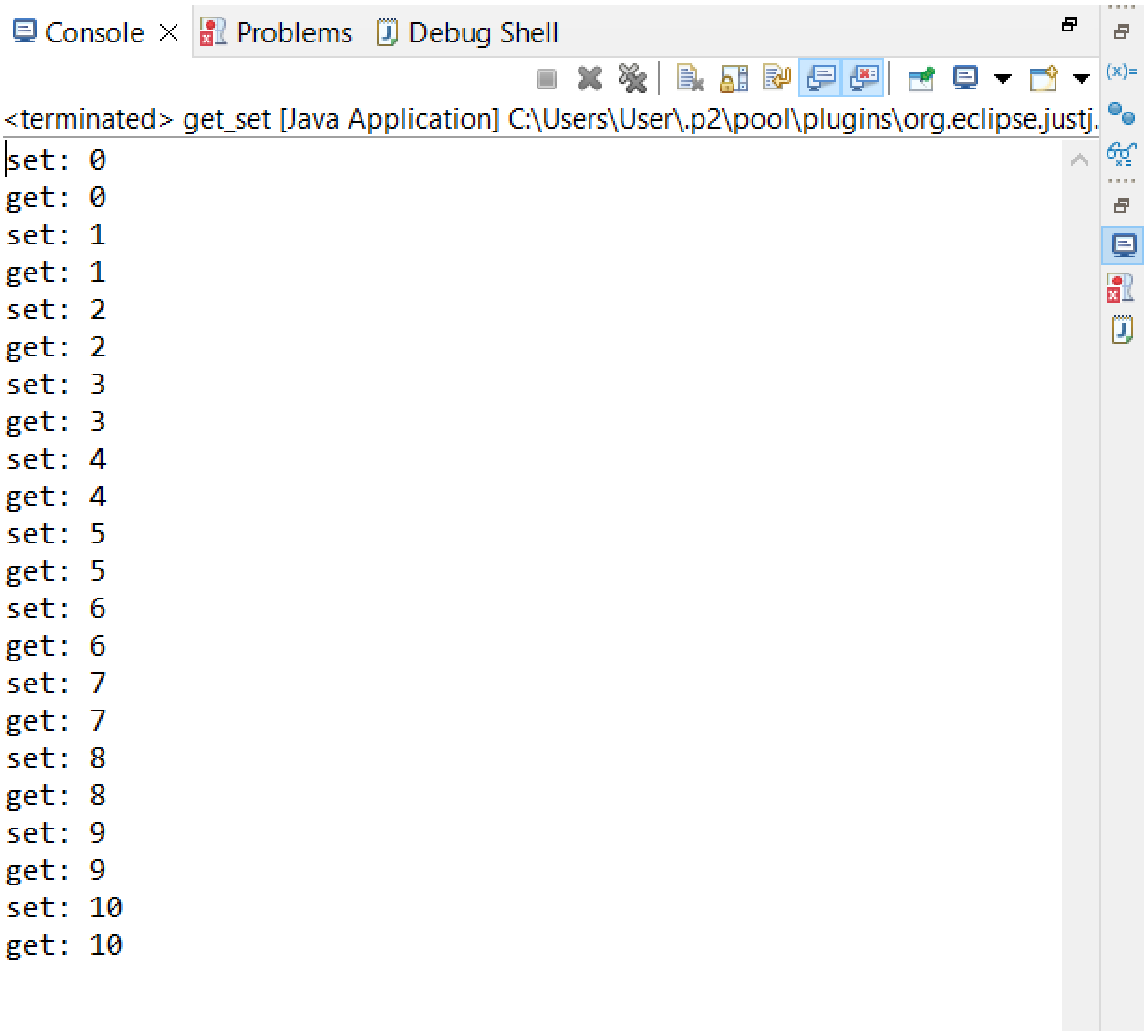
```
                p1 p2=new p1(c11);
                p1.start();
                p2.start();
        }
}
```

Output:



```
<terminated> get_set [Java Application] C:\Users\User\.p2\pool\plugins\org.eclipse.justj.
set: 0
get: 0
get: 0
set: 1
get: 1
set: 2
set: 3
get: 3
set: 4
get: 4
set: 5
get: 5
set: 6
get: 6
get: 6
set: 7
get: 7
set: 8
set: 9
get: 9
set: 10
get: 10
```

After Wait and notify methods:



```
<terminated> get_set [Java Application] C:\Users\User\.p2\pool\plugins\org.eclipse.justj.
set: 0
get: 0
set: 1
get: 1
set: 2
get: 2
set: 3
get: 3
set: 4
get: 4
set: 5
get: 5
set: 6
get: 6
set: 7
get: 7
set: 8
get: 8
set: 9
get: 9
set: 10
get: 10
```