**MULTITHREADING**

**CONCEPTS LEARNED**

1. Creating threads.
2. Joining threads.
3. Thread synchronisation.
4. Inter thread communication.
5. Program 1.
6. Program 2.

### **CREATING THREADS**

There are two ways in which thread can be created.

1. Extending Thread class
2. Implementing runnable interface

### **JOINING THREADS**

Use to execute one thread after another thread has completed its execution.

t1.join();

### **THREAD SYNCHRONIZATION**

When multiple threads try to access same resource, synchronization is used.

### **INTER-THREAD COMMUNICATION**

**wait()** tells the calling thread to give up the monitor and go to sleep until some other thread enters the same monitor and calls notify().

**notify()** wakes up the first thread that called wait() on the same object.

**notifyAll()** wakes up all the threads that called wait() on the same object. The highest priority thread will run first.

## **PROGRAM 1**

Given three arrays

Array 1 = [2,4,6,8,10,12];

Array 2 = [12,14,16,18,20,22];

Array 3 = [22,24,26,28,30,32];

Create three threads thread1,thread2 and thread3 that should access individual arrays.

Output : 2 12 22 4 14 24 6 16 26 8 18 28 10 20 30 12 22 32

#### **SOLUTION**

#### <https://github.com/M-Abishaik/Zterns/tree/master/Multithreading/Thread_one>

#### **PROGRAM 2**

[Given two arrays of same length](https://github.com/sruthiviswanathan/Multithreading/blob/master/src/com/zilker/multithreading/Multithreadsum.java)

Array 1 = [1,2,3,4,5];

Array 2 = [2,3,4,5,6];

Square the elements in the first array.Cube all the elements in the second array.

Add the elements of first array with elements of second array.

Print the resultant array.Use appropriate multithreading techniques.

**SOLUTION**

<https://github.com/M-Abishaik/Zterns/tree/master/Multithreading/Thread_two>