21BCE7371
RADHA KRISHNA GARG

# OPERATING SYSTEMS LAB
# IMPLEMENTATION OF MULTITHREADING

## Multithreading in JAVA

### Threads can be created by using two mechanisms :

### 1. Extending the Thread class

### INPUT

#### CODE

```
// Java code for thread creation by extending
// the Thread class
class MultithreadingDemo extends Thread {
    public void run()
    {
        try {
            // Displaying the thread that is running
            System.out.println(
                "Thread " + Thread.currentThread().getId()
                + " is running");
        }
        catch (Exception e) {
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}

// Main Class
public class Multithread {
    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
```

```
            MultithreadingDemo object
                = new MultithreadingDemo();
            object.start();
        }
    }
}
```

## OUTPUT

```
Output

java -cp /tmp/dIDaKeYvsz Multithread
Thread 17 is running
Thread 14 is running
Thread 12 is running
Thread 10 is running
Thread 16 is running
Thread 11 is running
Thread 13 is running
Thread 15 is running
```

## 2. Implementing the Runnable Interface

```
// Java code for thread creation by implementing
// the Runnable Interface
class MultithreadingDemo implements Runnable {
    public void run()
    {
        try {
            // Displaying the thread that is running
            System.out.println(
                "Thread " + Thread.currentThread().getId()
                + " is running");
        }
        catch (Exception e) {
```

```
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}

// Main Class
class Multithread {
    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            Thread object
                = new Thread(new MultithreadingDemo());
            object.start();
        }
    }
}
```

## OUTPUT

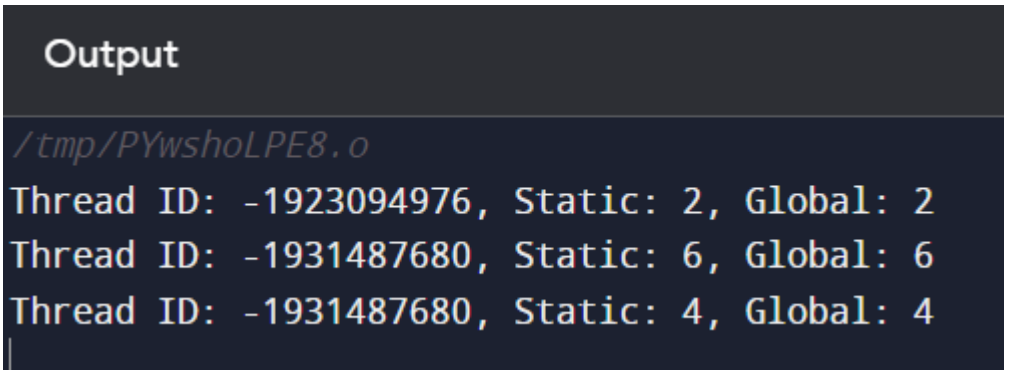
```
Output

java -cp /tmp/Lypf7NQJfg Multithread
Thread 15 is running
Thread 17 is running
Thread 11 is running
Thread 14 is running
Thread 13 is running
Thread 12 is running
Thread 16 is running
Thread 10 is running
```

# Multithreading in C

## 1. Using P threads

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>

// Let us create a global variable to change it in threads
int g = 0;

// The function to be executed by all threads
void *myThreadFun(void *vargp)
{
    // Store the value argument passed to this thread
    int *myid = (int *)vargp;

    // Let us create a static variable to observe its changes
    static int s = 0;

    // Change static and global variables
    ++s; ++g;

    // Print the argument, static and global variables
    printf("Thread ID: %d, Static: %d, Global: %d\n", *myid, ++s,
++g);
}

int main()
{
    int i;
    pthread_t tid;

    // Let us create three threads
    for (i = 0; i < 3; i++)
        pthread_create(&tid, NULL, myThreadFun, (void *)&tid);

    pthread_exit(NULL);
    return 0;
}
```

## OUTPUT


Output
/tmp/PYwshoLPE8.o
Thread ID: -1923094976, Static: 2, Global: 2
Thread ID: -1931487680, Static: 6, Global: 6
Thread ID: -1931487680, Static: 4, Global: 4