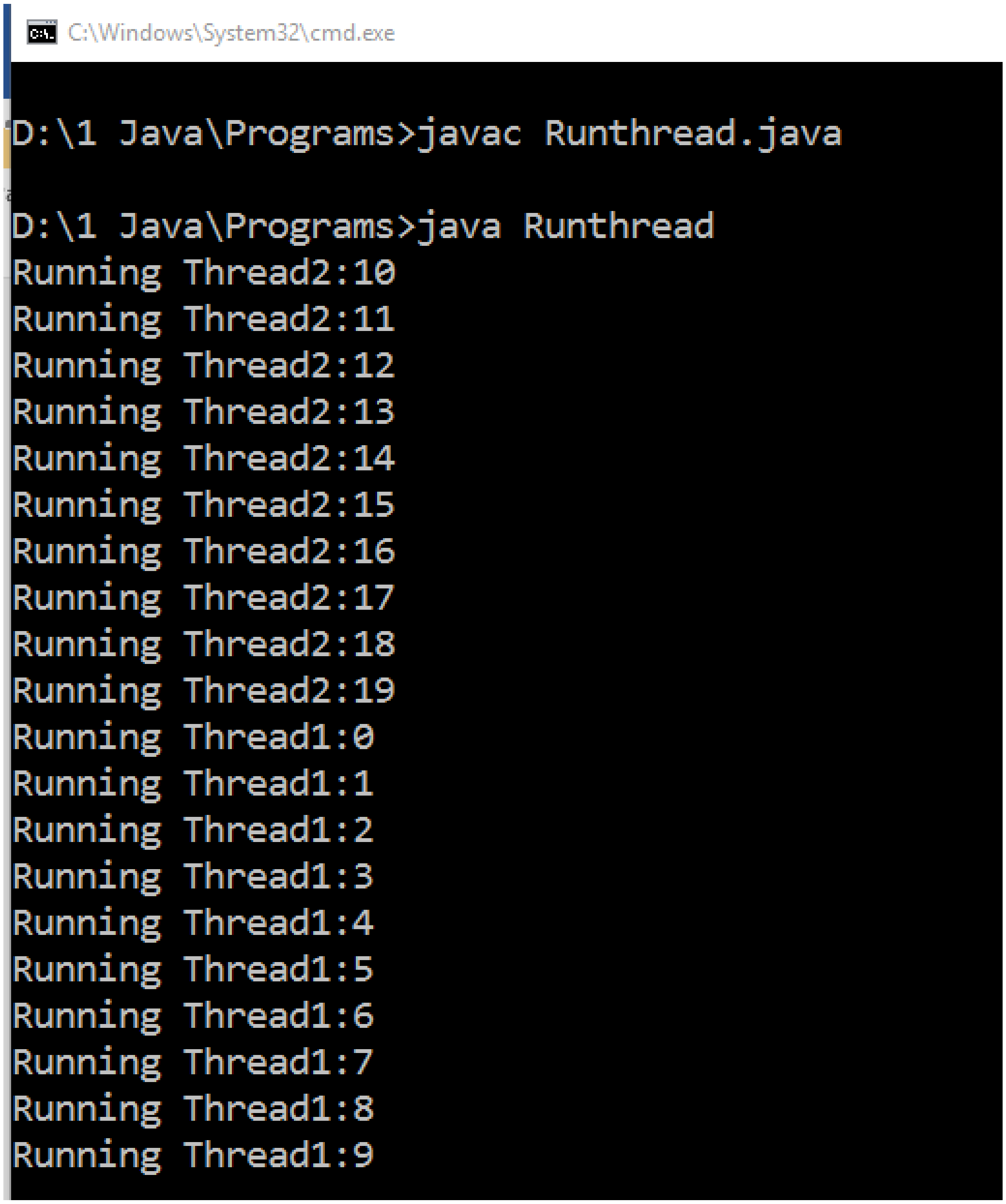
Important Methods of Thread class:

sleep() method:

sleep(): This method causes the currently executing thread to sleep for the specified number of milliseconds, subject to the precision and accuracy of system timers and schedulers. The sleep() method of Thread class is used to sleep a thread for the specified amount of time.

public static void sleep(long milliseconds) throws InterruptedException

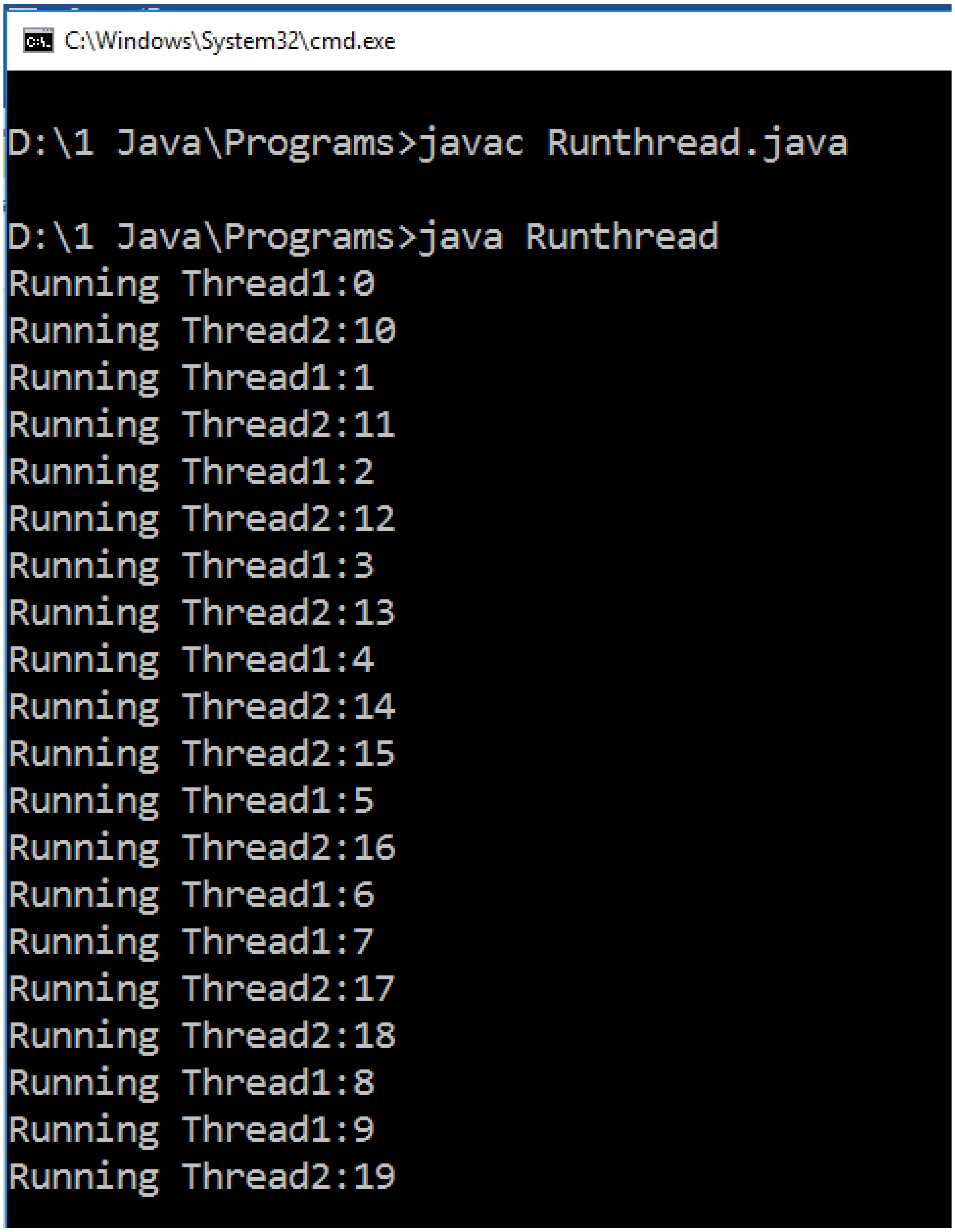
```
class Mythread1 implements Runnable {
        public void run() {
                for(int i=0;i<10;i++) {
                         try{
       Thread.sleep(1000); // sleeps thread for 1 sec
     } catch(InterruptedException e){
        System.out.println(e);
     }
                        System.out.println("Running Thread1:"+i); }
        } }
class Mythread2 extends Thread {
        public void run() {
                for(int i=10;i<20;i++)
                {
        System.out.println("Running Thread2:"+i); }
        } }
class Runthread {
        public static void main(String arg[]){
   Mythread1 r1=new Mythread1();
         Thread t1=new Thread(r1,"thread1");
     t1.start();
   Mythread2 r2=new Mythread2();
         r2.start();
                }
}
```



```
D:\1 Java\Programs>javac Runthread.java

D:\1 Java\Programs>java Runthread
Running Thread2:10
Running Thread2:11
Running Thread2:12
Running Thread2:13
Running Thread2:14
Running Thread2:15
Running Thread2:16
Running Thread2:17
Running Thread2:18
Running Thread2:19
Running Thread1:0
Running Thread1:1
Running Thread1:2
Running Thread1:3
Running Thread1:4
Running Thread1:5
Running Thread1:6
Running Thread1:7
Running Thread1:8
Running Thread1:9
```

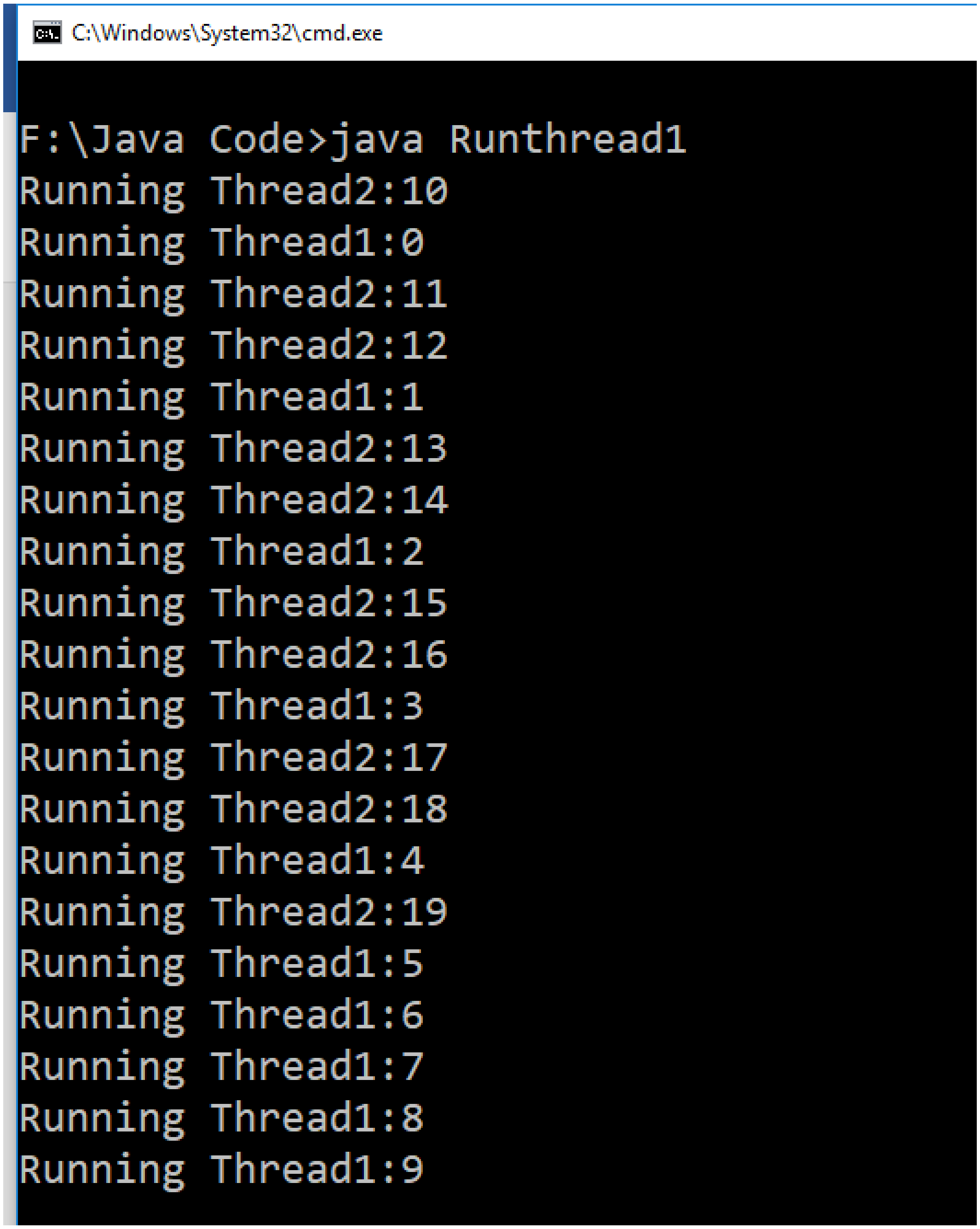
```
class Mythread1 implements Runnable {
    public void run() {
        for(int i=0;i<10;i++) {
            try{
    Thread.sleep(1000); // sleeps thread for 1 sec
   } catch(InterruptedException e){
    System.out.println(e);
   }
            System.out.println("Running Thread1:"+i); }
    } }
class Mythread2 extends Thread {
    public void run() {
        for(int i=10;i<20;i++)
        {
    try{
    Thread.sleep(1000); // sleeps thread for 1 sec
   } catch(InterruptedException e){
    System.out.println(e);
   }
    System.out.println("Running Thread2:"+i); }
    } }
class Runthread {
    public static void main(String arg[]){
  Mythread1 r1=new Mythread1();
    Thread t1=new Thread(r1,"thread1");
   t1.start();
  Mythread2 r2=new Mythread2();
    r2.start();
        }
}
```



```
D:\1 Java\Programs>javac Runthread.java

D:\1 Java\Programs>java Runthread
Running Thread1:0
Running Thread2:10
Running Thread1:1
Running Thread2:11
Running Thread1:2
Running Thread2:12
Running Thread1:3
Running Thread2:13
Running Thread1:4
Running Thread2:14
Running Thread2:15
Running Thread1:5
Running Thread2:16
Running Thread1:6
Running Thread1:7
Running Thread2:17
Running Thread2:18
Running Thread1:8
Running Thread1:9
Running Thread2:19
```

```
class Mythread1 implements Runnable {
        public void run() {
                for(int i=0;i<10;i++) {
                         try{
       Thread.sleep(2000); // sleeps thread for 2 sec
     } catch(InterruptedException e){
        System.out.println(e);
     }
                        System.out.println("Running Thread1:"+i); }
        } }
class Mythread2 extends Thread {
        public void run() {
                for(int i=10;i<20;i++)
                {
        try{
       Thread.sleep(1000); // sleeps thread for 1 sec
     } catch(InterruptedException e){
        System.out.println(e);
     }
        System.out.println("Running Thread2:"+i); }
        } }
class Runthread1 {
        public static void main(String arg[]){
   Mythread1 r1=new Mythread1();
         Thread t1=new Thread(r1,"thread1");
     t1.start();
   Mythread2 r2=new Mythread2();
         r2.start();
                }
}
```



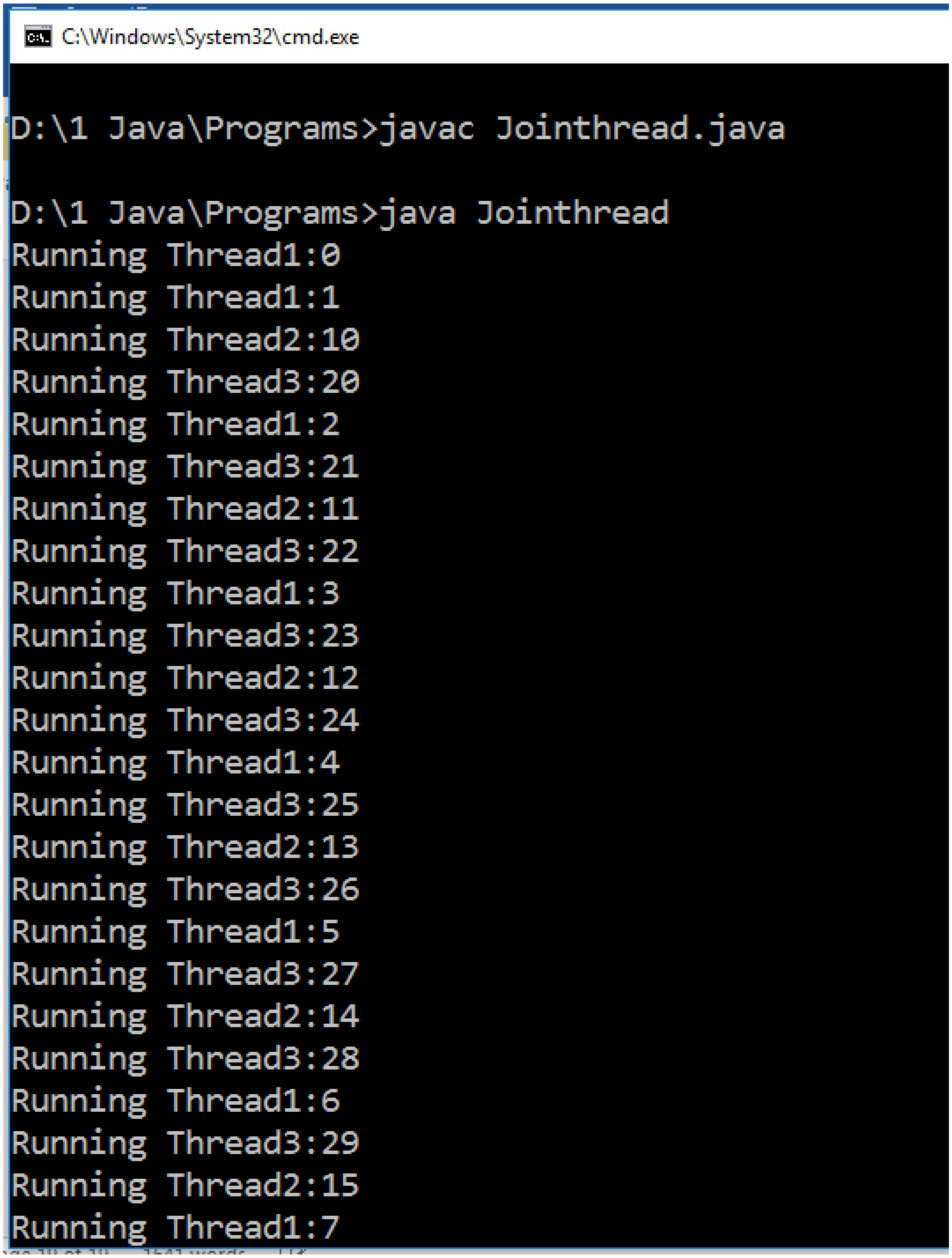
```
F:\Java Code>java Runthread1
Running Thread2:10
Running Thread1:0
Running Thread2:11
Running Thread2:12
Running Thread1:1
Running Thread2:13
Running Thread2:14
Running Thread1:2
Running Thread2:15
Running Thread2:16
Running Thread1:3
Running Thread2:17
Running Thread2:18
Running Thread1:4
Running Thread2:19
Running Thread1:5
Running Thread1:6
Running Thread1:7
Running Thread1:8
Running Thread1:9
```

//This program show the multi threading between 3 threads without join().

```
class Mythread1 implements Runnable {
        public void run() {
                for(int i=0;i<10;i++) {
System.out.println("Running Thread1:"+i); }
        } }
class Mythread2 extends Thread {
        public void run() {
                for(int i=10;i<20;i++)

                {
        System.out.println("Running Thread2:"+i); }
        } }

        class Mythread3 extends Thread {
        public void run() {
                for(int i=20;i<30;i++)
                {
        System.out.println("Running Thread3:"+i); }
        } }
class Jointhread {
        public static void main(String arg[]){
   Mythread1 r1=new Mythread1();
         Thread t1=new Thread(r1,"thread1");
   t1.start();
   Mythread2 r2=new Mythread2();
         r2.start();
         Mythread3 r3=new Mythread3();
         r3.start();
                }
}
```



```
D:\1 Java\Programs>javac Jointhread.java

D:\1 Java\Programs>java Jointhread
Running Thread1:0
Running Thread1:1
Running Thread2:10
Running Thread3:20
Running Thread1:2
Running Thread3:21
Running Thread2:11
Running Thread3:22
Running Thread1:3
Running Thread3:23
Running Thread2:12
Running Thread3:24
Running Thread1:4
Running Thread3:25
Running Thread2:13
Running Thread3:26
Running Thread1:5
Running Thread3:27
Running Thread2:14
Running Thread3:28
Running Thread1:6
Running Thread3:29
Running Thread2:15
Running Thread1:7
```

.

.

.

join() method.

java.lang.Thread class provides the join() method which allows one thread to wait until another thread completes its execution. If **t** is a Thread object whose thread is currently executing, then **t.join()** will make sure that **t** is terminated before the next instruction is executed by the program.

If there are multiple threads calling the join() methods that means overloading on join allows the programmer to specify a waiting period. However, as with sleep, join is dependent on the OS for timing, so you should not assume that join will wait exactly as long as you specify. There are three overloaded join functions.

1. **join():** It will put the current thread on wait until the thread on which it is called is dead. If thread is interrupted then it will throw InterruptedException.
   Syntax:

   ```
   public final void join()
   ```

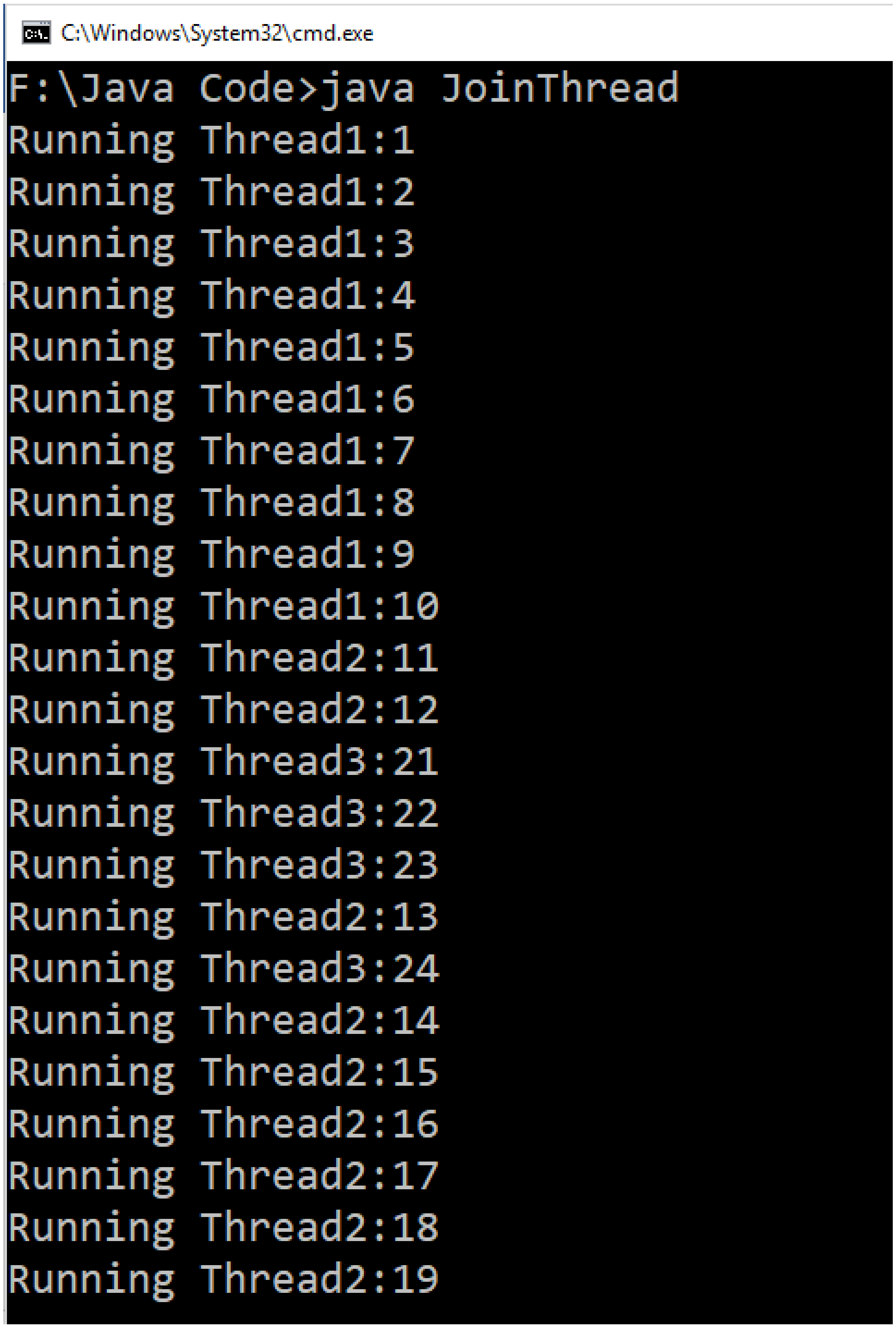
2. **join(long millis)** :It will put the current thread on wait until the thread on which it is called is dead or wait for specified time (milliseconds).
   Syntax:
   ```
   public final synchronized void join(long millis)
   ```

//This program shows the working of join() method.

```
class Mythread1 implements Runnable {
        public void run() {
                for(int i=1;i<11;i++) {
                        System.out.println("Running Thread1:"+i); }
        } }
class Mythread2 extends Thread {
        public void run() {
                for(int i=11;i<21;i++)
                {
        System.out.println("Running Thread2:"+i); }
        } }
        class Mythread3 extends Thread {
        public void run() {
                for(int i=21;i<31;i++)
                {
        System.out.println("Running Thread3:"+i); }
        } }

class JoinThread {
        public static void main(String arg[]){
        Mythread1 r1=new Mythread1();
         Thread t1=new Thread(r1,"thread1");
     t1.start();
                try{
```

```
                        t1.join();  //it force to complete the task of t1 thread.
                        }
                catch(Exception e)
                {
                        System.out.println(e);
                }
   Mythread2 t2=new Mythread2();
        t2.start();
        Mythread3 t3=new Mythread3();
        t3.start();
                }
}
```

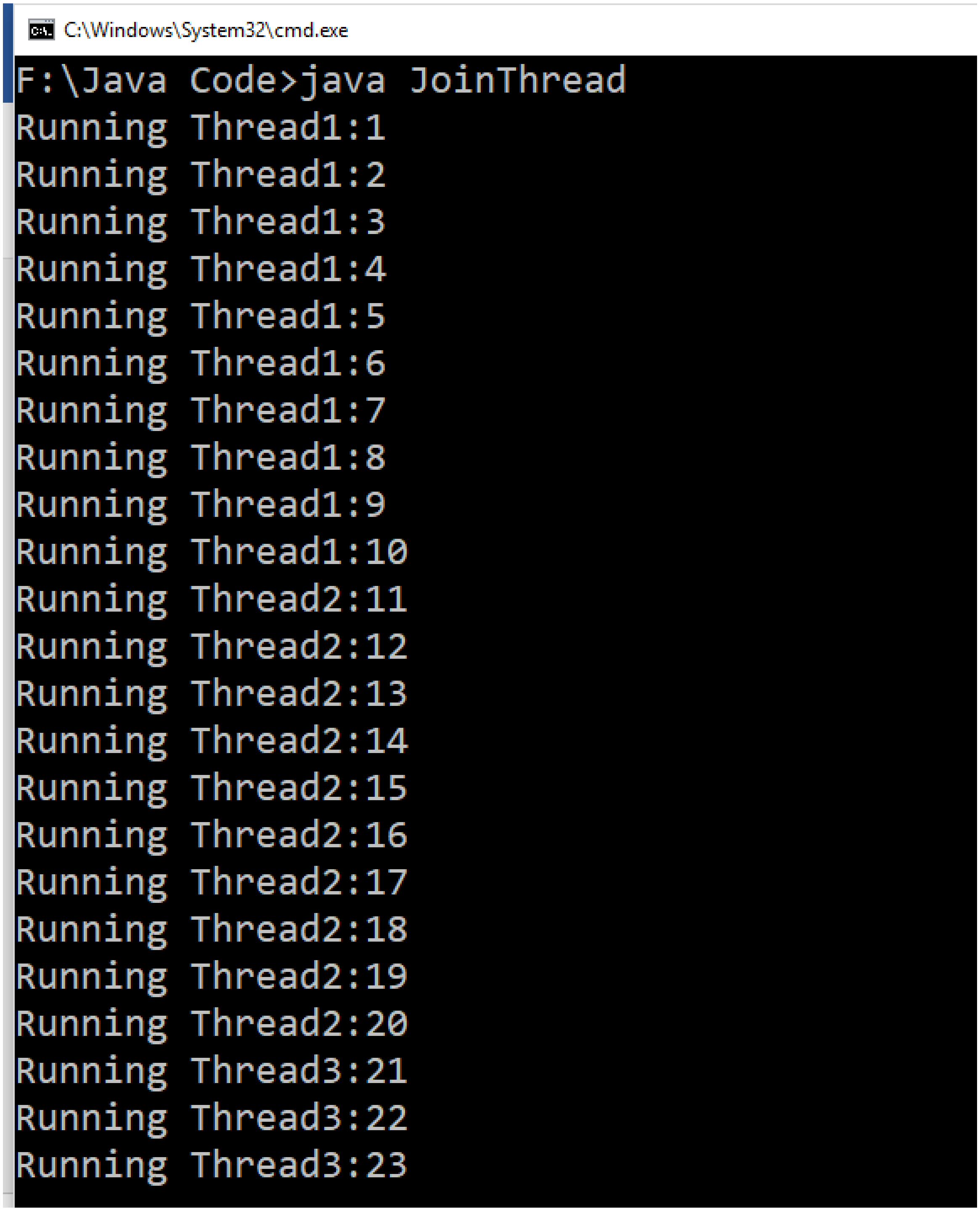


```
F:\Java Code>java JoinThread
Running Thread1:1
Running Thread1:2
Running Thread1:3
Running Thread1:4
Running Thread1:5
Running Thread1:6
Running Thread1:7
Running Thread1:8
Running Thread1:9
Running Thread1:10
Running Thread2:11
Running Thread2:12
Running Thread3:21
Running Thread3:22
Running Thread3:23
Running Thread2:13
Running Thread3:24
Running Thread2:14
Running Thread2:15
Running Thread2:16
Running Thread2:17
Running Thread2:18
Running Thread2:19
```

```
class Mythread1 implements Runnable {
        public void run() {
                for(int i=1;i<11;i++) {
                        System.out.println("Running Thread1:"+i); }
        } }
class Mythread2 extends Thread {
        public void run() {
                for(int i=11;i<21;i++)
                {
        System.out.println("Running Thread2:"+i); }
        } }
        class Mythread3 extends Thread {
        public void run() {
                for(int i=21;i<31;i++)
                {
        System.out.println("Running Thread3:"+i); }
        } }

class JoinThread {
        public static void main(String arg[]){
        Mythread1 r1=new Mythread1();
         Thread t1=new Thread(r1,"thread1");
     t1.start();
                try{
                        t1.join();  //it force to complete the task of t1 thread.
                        }
                catch(Exception e)
                {
                        System.out.println(e);
                }

   Mythread2 t2=new Mythread2();
         t2.start();
         try{
                        t2.join();  //it force to complete the task of t2 thread.
                        }
                catch(Exception e)
                {
                        System.out.println(e);
                }
         Mythread3 t3=new Mythread3();
         t3.start();
                }
}
```



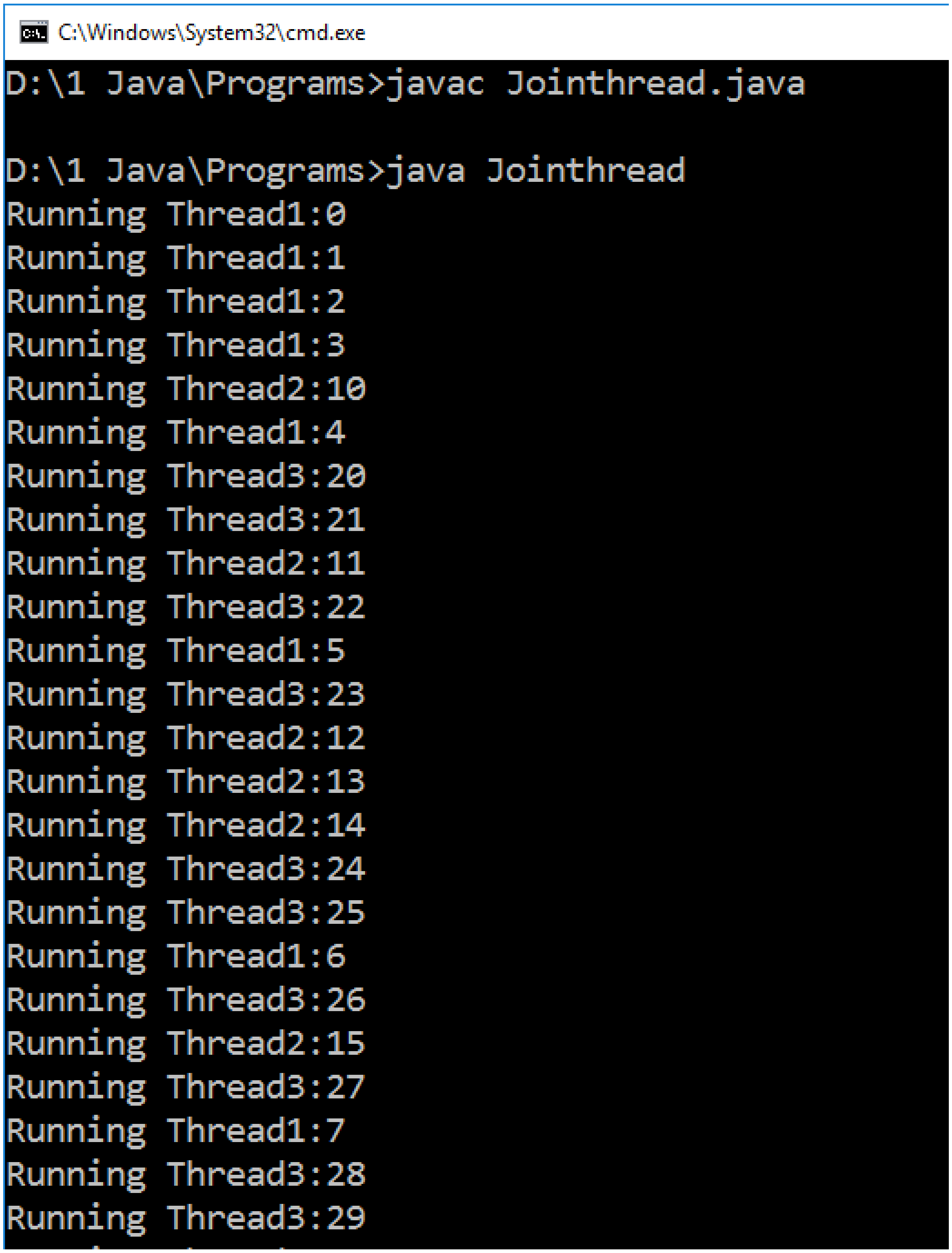
```
F:\Java Code>java JoinThread
Running Thread1:1
Running Thread1:2
Running Thread1:3
Running Thread1:4
Running Thread1:5
Running Thread1:6
Running Thread1:7
Running Thread1:8
Running Thread1:9
Running Thread1:10
Running Thread2:11
Running Thread2:12
Running Thread2:13
Running Thread2:14
Running Thread2:15
Running Thread2:16
Running Thread2:17
Running Thread2:18
Running Thread2:19
Running Thread2:20
Running Thread3:21
Running Thread3:22
Running Thread3:23
```

//show the working of join(long miliseconds) method. In this example,when t1 is completes its task for 10 miliseconds then r2 and r3 starts executing.

```
class Mythread1 implements Runnable {
    public void run() {
        for(int i=0;i<10;i++) {

            System.out.println("Running Thread1:"+i); }
    } }
class Mythread2 extends Thread {
    public void run() {
        for(int i=10;i<20;i++)

        {
    System.out.println("Running Thread2:"+i); }
    } }
    class Mythread3 extends Thread {
    public void run() {
        for(int i=20;i<30;i++)
        {
    System.out.println("Running Thread3:"+i); }
    } }
class Jointhread {
    public static void main(String arg[]){
  Mythread1 r1=new Mythread1();
     Thread t1=new Thread(r1,"thread1");
   t1.start();
        try{
    t1.join(10);  //it force to complete the task of t1 thread for 10 milli sec.
            }
        catch(Exception e)
        {
            System.out.println(e);
        }
  Mythread2 r2=new Mythread2();
     r2.start();
     Mythread3 r3=new Mythread3();
     r3.start();
        }
}
```



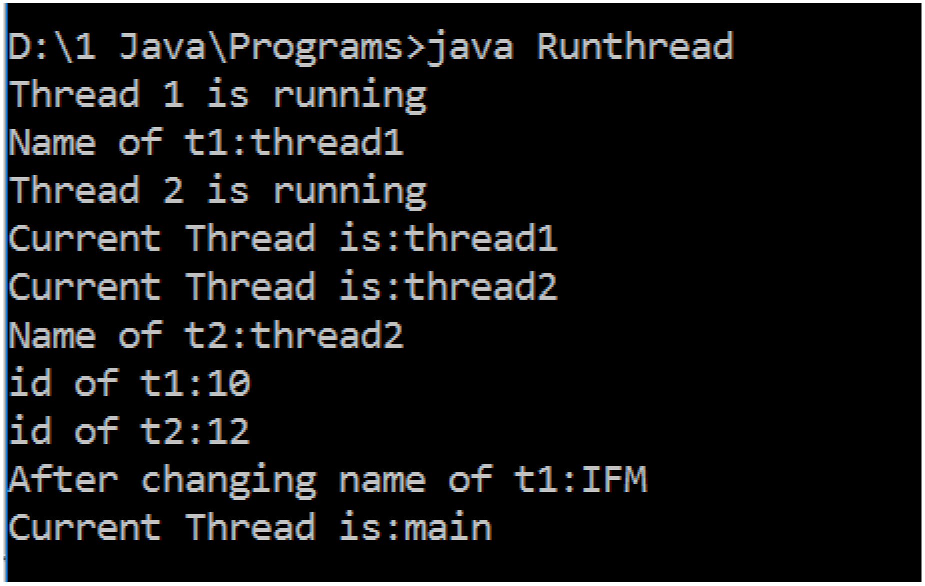
```
D:\1 Java\Programs>javac Jointhread.java

D:\1 Java\Programs>java Jointhread
Running Thread1:0
Running Thread1:1
Running Thread1:2
Running Thread1:3
Running Thread2:10
Running Thread1:4
Running Thread3:20
Running Thread3:21
Running Thread2:11
Running Thread3:22
Running Thread1:5
Running Thread3:23
Running Thread2:12
Running Thread2:13
Running Thread2:14
Running Thread3:24
Running Thread3:25
Running Thread1:6
Running Thread3:26
Running Thread2:15
Running Thread3:27
Running Thread1:7
Running Thread3:28
Running Thread3:29
```

getName(),setName(String) and getId() ,currentThread() methods:

```
class Mythread1 implements Runnable {
            public void run() {
                        System.out.println("Thread 1 is running");
System.out.println("Current Thread    is:"+Thread.currentThread().getName());
            } }
class Mythread2 extends Thread {
            public void run() {
                        System.out.println("Thread 2 is running");
System.out.println("Current Thread is:"+Thread.currentThread().getName());
          } }


class Runthread3 {
   public static void main(String arg[]){
            Mythread1 r1=new Mythread1();
             Thread t1=new Thread(r1,"thread1");
              t1.start();
             Mythread2 r2=new Mythread2();
             Thread t2=new Thread(r2,"thread2");
            t2.start();
            System.out.println("Name of t1:"+t1.getName());
            System.out.println("Name of t2:"+t2.getName());
            System.out.println("id of t1:"+t1.getId());
            System.out.println("id of t2:"+t2.getId());
            t1.setName("IFM");
           System.out.println("After changing name of t1:"+t1.getName());
           System.out.println("Current Thread is:"+Thread.currentThread().getName());
                        }}
```



```
D:\1 Java\Programs>java Runthread
Thread 1 is running
Name of t1:thread1
Thread 2 is running
Current Thread is:thread1
Current Thread is:thread2
Name of t2:thread2
id of t1:10
id of t2:12
After changing name of t1:IFM
Current Thread is:main
```