

STIWK3014 REAL TIME PROGRAMMING
TUTORIAL/EXERCISE 4: Thread States

TASK 1: ThreadNew

```
package Week_04;

class ThreadNew {

    public static void main(String[] args) {
        Thread t = new Thread();
        System.out.println(t.getState());
    }
}
```

TASK 2: ThreadRunnable

```
package Week_04;

class ThreadRunnable {

    public static void main(String[] args) {
        Thread t = new Thread();
        t.start();
        System.out.println(t.getState());
    }
}
```

TASK 3: ThreadStates

```
package Week_04;

class ThreadStates {

    public static void main(String[] args) {
        Thread t = new Thread();
        Thread.State e = t.getState();
        Thread.State[] ts = e.values();    for
        (int i = 0; i < ts.length; i++) {
```

```

        System.out.println(ts[i]);
    }
}
}

```

TASK 4: ThreadStatesInJava

```

package Week_04;

class ThreadStatesInJava {

    public static void main(String[] args) {
        Thread.State[] states = Thread.State.values();

        for (Thread.State state : states) {
            System.out.println(state);
        }
    }
}

```

TASK 5: ThreadWaiting

```

package Week_04;

class ThreadWaiting {

    public static void main(String[] args) {

        Thread t1 = new Thread() {
            public void run() {
try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        };

        Thread t2 = new Thread() {
            public void run() {
try {

```

```

        t1.join();
    } catch (InterruptedException e) {
e.printStackTrace();
    }
}
};
t2.start();
t1.start();

try {
    Thread.sleep(100);    } catch
(InterruptedException e) {
e.printStackTrace();
}

    System.out.println(t2.getState());
}
}

```

TASK 6: ThreadTimedWaiting

```

package Week_04;

class ThreadTimedWaiting {

    public static void main(String[] args) {

        Thread t = new Thread() {
            public void run() {
try {
                Thread.sleep(5000);    }
catch (InterruptedException e) {
e.printStackTrace();
            }
        }
    };

    t.start();

    try {

```

```

        Thread.sleep(2000);    }
catch (InterruptedException e) {
e.printStackTrace();
    }

    System.out.println(t.getState());
}
}

```

TASK 7: Shared (This is Blocked States feature)

```

package Week_04;

class Shared {

    synchronized void methodOne(Shared s) {
        try {
            Thread.sleep(2000);    }
catch (InterruptedException e) {
e.printStackTrace();
    }
        s.methodTwo(this);
    }

    synchronized void methodTwo(Shared s) {
        try {
            Thread.sleep(2000);    }
catch (InterruptedException e) {
e.printStackTrace();
    }
        s.methodOne(this);
    }
}

class ThreadBlocked {

    public static void main(String[] args) {

        final Shared s1 = new Shared();
final Shared s2 = new Shared();

```

```

        Thread t1 = new Thread() {
@Override          public void
run() {
            s1.methodOne(s2);
        }
    };

        Thread t2 = new Thread() {
@Override          public void
run() {
            s2.methodTwo(s1);
        }
    };

        t1.start();
t2.start();

        try {
            Thread.sleep(3000);
        } catch (InterruptedException e) {
e.printStackTrace();
        }

        System.out.println("state t1 = " + t1.getState());
        System.out.println("state t2 = " + t2.getState());
    }
}

```

TASK 8: ThreadTerminated

```

package Week_04;

class ThreadTerminated {

    public static void main(String[] args) {
        Thread t = new Thread() {
@Override          public void run()
{
            for (int i = 0; i <= 10; i++)
            {
                System.out.println(i);
            }
        }
    }
}

```

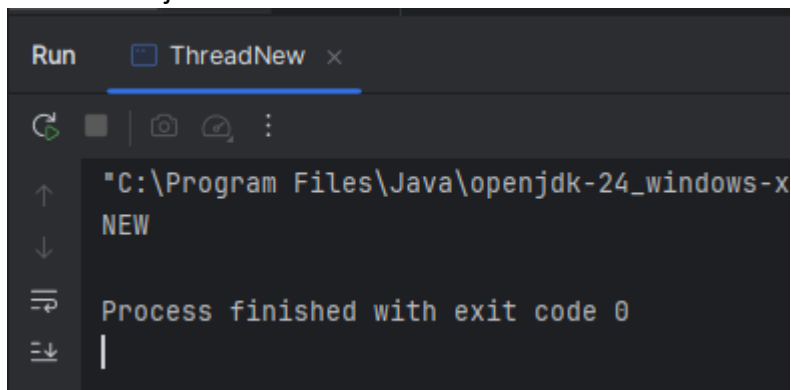
```
};

t.start();

try {
    Thread.sleep(2000);    }
catch (InterruptedException e) {
    e.printStackTrace();
    }
    System.out.println(t.getState());
}
}
```

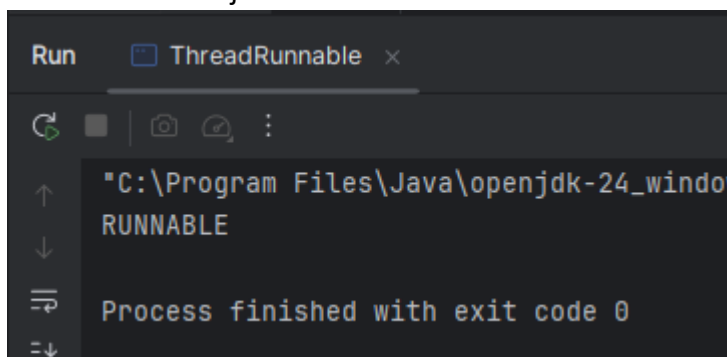
Output

ThreadNew.java



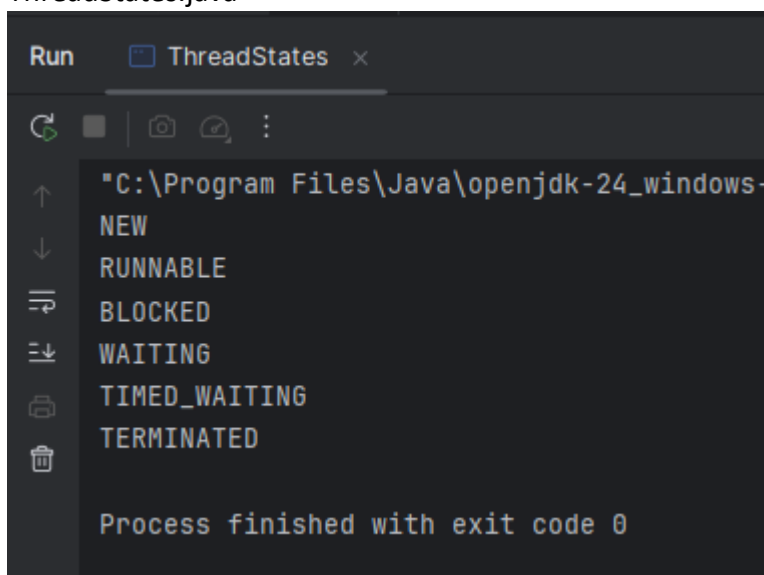
```
Run ThreadNew x
"C:\Program Files\Java\openjdk-24_windows-x
NEW
Process finished with exit code 0
```

ThreadRunnable.java



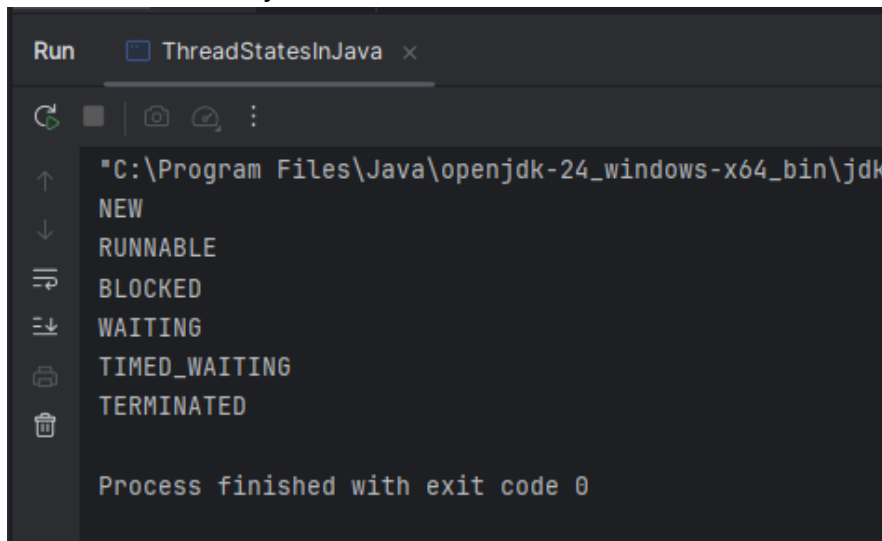
```
Run ThreadRunnable x
"C:\Program Files\Java\openjdk-24_window
RUNNABLE
Process finished with exit code 0
```

ThreadStates.java



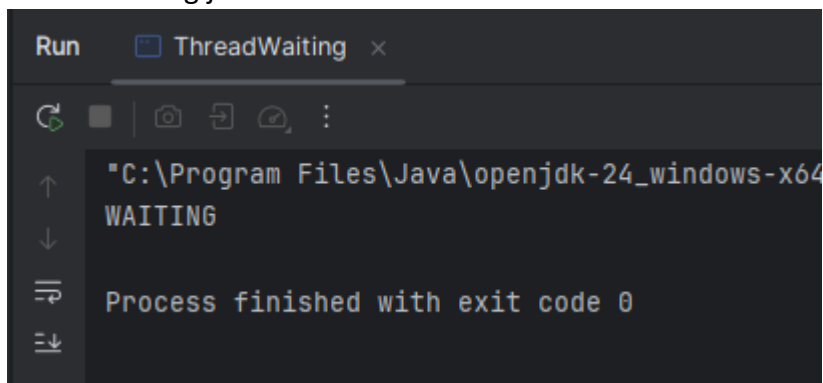
```
Run ThreadStates x
"C:\Program Files\Java\openjdk-24_windows-
NEW
RUNNABLE
BLOCKED
WAITING
TIMED_WAITING
TERMINATED
Process finished with exit code 0
```

ThreadStatesInJava.java



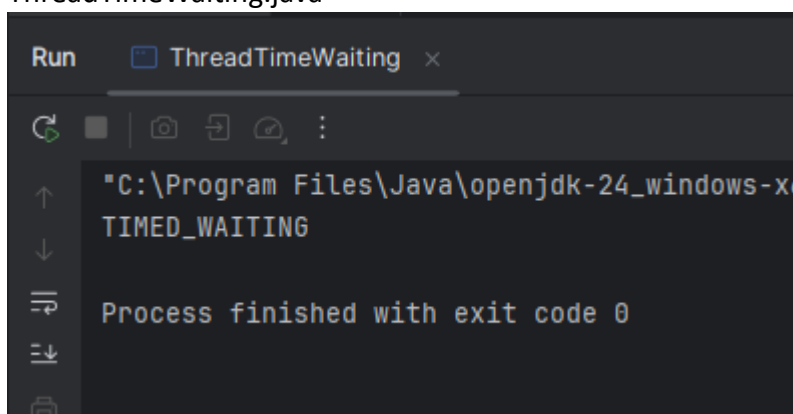
```
Run ThreadStatesInJava x
"C:\Program Files\Java\openjdk-24_windows-x64_bin\jdk-24\bin\java.exe"
NEW
RUNNABLE
BLOCKED
WAITING
TIMED_WAITING
TERMINATED
Process finished with exit code 0
```

ThreadWaiting.java



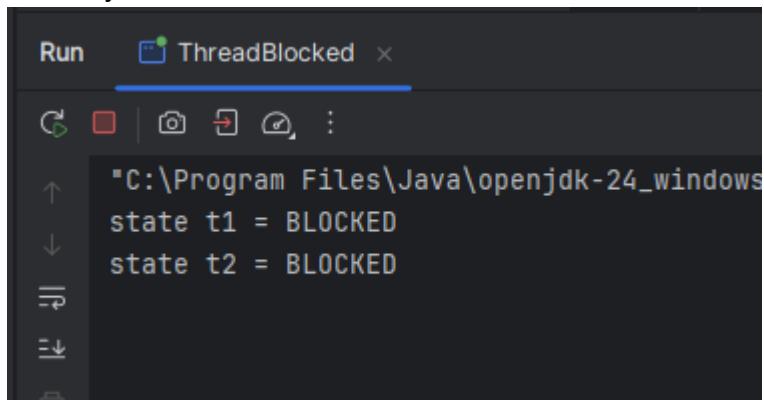
```
Run ThreadWaiting x
"C:\Program Files\Java\openjdk-24_windows-x64_bin\jdk-24\bin\java.exe"
WAITING
Process finished with exit code 0
```

ThreadTimeWaiting.java



```
Run ThreadTimeWaiting x
"C:\Program Files\Java\openjdk-24_windows-x64_bin\jdk-24\bin\java.exe"
TIMED_WAITING
Process finished with exit code 0
```

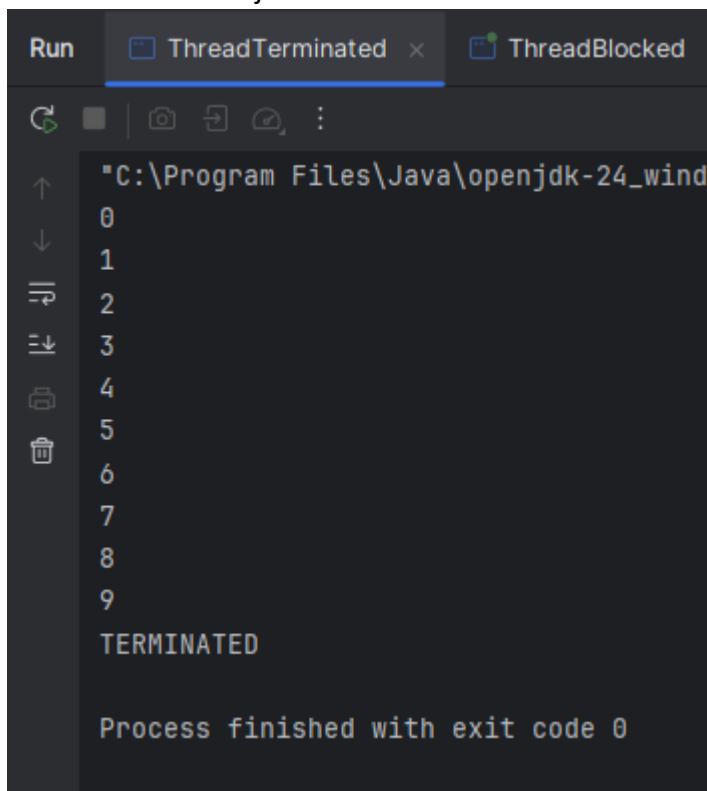

Shared.java



The screenshot shows an IDE console window titled "Run" with a tab for "ThreadBlocked". The console output displays the file path "C:\Program Files\Java\openjdk-24_windows" followed by two lines of text: "state t1 = BLOCKED" and "state t2 = BLOCKED". The left sidebar contains standard IDE navigation icons.

```
"C:\Program Files\Java\openjdk-24_windows
state t1 = BLOCKED
state t2 = BLOCKED
```

ThreadTerminated.java



The screenshot shows an IDE console window titled "Run" with tabs for "ThreadTerminated" and "ThreadBlocked". The console output for "ThreadTerminated" shows the file path "C:\Program Files\Java\openjdk-24_windows" followed by a list of numbers from 0 to 9, then the word "TERMINATED", and finally "Process finished with exit code 0". The left sidebar contains standard IDE navigation icons.

```
"C:\Program Files\Java\openjdk-24_windows
0
1
2
3
4
5
6
7
8
9
TERMINATED

Process finished with exit code 0
```