Implementing Processes

Use Java threads:

1) extend `Thread`, or

2) implement `Runnable`

Both have member `public void run()`.


```
class MyThread extends Thread {
 // ...
}

class MyRun implements Runnable {
 // ...
}

Thread a = new MyThread();
Thread b = new Thread(new MyRun());

a.start(); // causes a.run() to be called
b.start();
```


Lifecycle

- Create.
- `start()` causes `run()` to be called.
- Terminate on return from `run()`.
- Thread can give up processor using `yield()`.
- `sleep(n)` causes thread to be suspend for n milliseconds.


```
public class Shared {

  public static void main(String[] args) {
    Counter c = new Counter();
    Thread a = new Thread(new Increment(c));
    Thread b = new Thread(new Increment(c));
    a.start();
    b.start();
    try {
      while (a.isAlive()) Thread.sleep(50);
      while (b.isAlive()) Thread.sleep(50);
    }
    catch (InterruptedException e) {}

    System.out.println("Counter = " + c.val);
    System.exit(0);
  }
}
```

```
class Increment implements Runnable {
  private Counter cnt;

  Increment(Counter c) {
    cnt = c;
  }

  public void run() {
    try {
      int tmp;
      for (int i = 1; i <= 10; i++) {
        tmp = cnt.val;
        Thread.sleep((int)Math.round(Math.random()*10));
        cnt.val = tmp + 1;
        Thread.sleep((int)Math.round(Math.random()*20));
      }
    }
    catch (InterruptedException e) {}
```



```
  }
}
```



```
class Counter
{
  public int val = 0;
}
```