Recognizing Patterns

- Counting: continually updating a value by a fixed amount
- Counting raindrops

```java
int dropCount = 0; // Raindrop counter
while (dropCount < MAX) {
    new Raindrop(...);
    dropCount++;
}
```

Counting Bricks

```java
while (count < TOTAL) {
    new Brick(...);
    count++;
}
```

The Counting while Loop

```java
int i = initialValue; // initialize
while (i < stopVal) { // test
    ... // do stuff
    i++; // increment
}
```

The for loop

- Especially useful for counting
- Ex:

```java
for (int i=initialVal; // initialize
    i<stopVal; // test
    i++) { // increment
    ... // do stuff
}
```

Counting Raindrops with for Loop

```java
for (int dropCount = 0; // initialize
    dropCount < MAX;
    dropCount++) {
    new Raindrop (...);
}
```
More General Start and End Points

- Loops can take whatever starting point, end point, and increment
  Ex:
  ```java
  for (int i=23; i <= 1728; i=i+591;){
      //do stuff
  }
  ```

- But one should avoid using a double for any of the three values

Counting Backwards with for Loop

Ex: Printing a countdown
```java
for (int count = 10; count >= 1; count--) {
    System.out.println(count);
}
```

Update Values

- Can increment loop index by any value
- Ex: Drawing grass blades
  ```java
  for (int pos = 0;
      pos < WIDTH;
      pos = pos + GAP) {
      new Line (pos, TOP, pos, GROUND, canvas);
  }
  ```

General Syntax of for Loop

- for (initialization; condition; update) {
  //Do something
}

Initialization: gen’ly creates a counting variable
Condition: a boolean expression to stop the loop
Updating: updates the variable created

Nested Loops

- Any loop body can contain another loop
  Ex: for ( … ) {
      while (…) {
          while (…) {
              for(…) {
              }
          }
          }
      }
  }

The do while Loop

- Syntax:
  ```java
  do {
      <code to repeat>
  } while (<condition>)
  ```

  Craps Example
do while Loop vs while Loop

- **do while**
  - Condition checked at the end
  - Loop body executed at least once
- **while**
  - Condition checked at the beginning
  - Loop body may never execute

Avoiding Loop Errors

- Easier to find errors if you know where to look

  - Common loop errors include:
    - Off by 1 in counting loops
    - Infinite loops

Off by one errors

Suppose we want to run a for loop 5 times:

```java
for(int i=0;i<=5;i++) {
}
```

The left hand version will run it 6 times, not 5.

Infinite Loops

Ex:

```java
while (count< TOTAL) {
  new Brick (...);
}
```

Since value of count is not updated, the condition in while will stay true forever.