

Control Structures - Repetition

3 Forms of Repetition Control Structures:

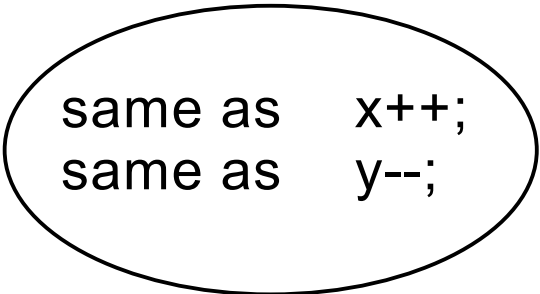
- > for loops
- > do/while loops
- > while loops

Each have unique uses and required knowledge.

Often Repetition Structures need to 'count'. This is done by counters. Counters are often (not always) found within a repetition structure, and keep track of how many times something has occurred.

Syntax for Counters:

<code>x = x + 1;</code>	same as	<code>x += 1;</code>
<code>y = y - 1;</code>	same as	<code>y -= 1;</code>
<code>z = z + 3;</code>	same as	<code>z += 3;</code>
<code>m = m * 6;</code>	same as	<code>m *= 6;</code>



same as `x++;`
same as `y--;`

Format of the
experienced
programmer!

Control Structures - Repetition (for loop)

For loop

Syntax:

```
for (int identifier = startValue; identifier <= endValue; identifier increment)
{ ... commands to repeat with the for loop ...
}
```

Counts from 1 through 10
Note: no semi-colon (not end of structure yet)

Examples:

```
total = 0;
for (int x= 1; x <= 10; x++)
{ c.println ( "Count: " + x);
  c.println ( "Enter a number: ");
  num = c.readInt();
  total = total + num;
}
c.println ( "Your 10 numbers total to: " + total);
```

Accumulator (totals)
(total must be set first to 0)

```
for (int x = 1; x <= 7000000000; x++);
```

Delay loop has ;
(special situation)

Control Structures - Repetition (for loop)

Special Notes about the FOR loop:

- Always pick the FOR loop when you are 'counting' or 'incrementing'
- The control identifier (int x) ... must be of integer type
- The control identifier uses memory temporarily ... since it declared WITHIN the control structure, it exists only within the structure ... after the FOR loop is completed the identifier no longer 'lives' (this is called local effect, thus a local identifier)
- The control identifier use a special form of memory, called INDEX memory ... it is the fastest type of memory
- Increments can go up or down
 - ▶ for (int x=1; x<=100; x++)
 - ▶ for (int x=100; x>=1; x--)