Programming Iterative Loops

- for
- while

What was an iterative loop, again?

Recall this definition:

Iteration is when the same procedure is repeated multiple times.

Some examples were long division, the Fibonacci numbers, prime numbers, and the calculator game. Some of these used recursion as well, but not all of them.

Two Types of Iterative Loops

1. `for` loop: used when you want to plug in a bunch of successive integers, or repeat a procedure a given number of times

2. `while` loop: used when you want to iterate until a certain condition is met, or when you know in advance how many loops to run

Loops with “for”

`for` loops are used when want to plug in a bunch of successive integers, or repeat a procedure a given number of times.

The general structure is:

```
for x in ___:___
    (do something)
end
```

Loops with “for”

Let’s say you wanted to print out the values of $y = x^2$ from x = 5 to x = 12. Here’s the code:

```
for x in 5:12
    println(x^2)
end
```

Longer Loops with “for”

Now suppose you want to add up the values of $y = x^2 – 5x + 11$ from x = 0 to x = some number n.

This procedure is a bit longer:

1. Plug x into f(x)
2. Add that to the total

Repeat for x-values from 0 to n.

Longer Loops with “for”

(Program to add up the values of $y = x^2 - 5x + 11$ from x = 0 to x = n)

```
function Sum(n)
     f(x) = x^2 - 5x + 11
     S = 0
     for x in 0:n
          S = S + f(x)
     end
     println(S)
end
```

A Funny Feature of “for”

`for` can also let you do something a certain number of times; you don’t even need to apply the value of n. Try this:

```
for n in 1:10
     println("Mrs. Crabapple is perfect!")
end
```

For Loops With Recursion

Recall that the powers of two can be generated using recursion – each term is double the previous term. Here's an example using a for loop to print out the first ten powers of two:

```
x = 1
for n in 1:10
    println(x)
    x = 2x
end
```

Practice Problems 1 and 2

1. Use a “for” loop to generate a list of values of $y = 4x^2 – 12$ from $x = -6$ to $x = 6$.

2. Use a “for” loop to generate a sequence of 12 terms, starting with 20, where each term is the square root of the previous term.

Fibonacci Numbers, Revisited

It is possible to generate the Fibonacci numbers using a “for” loop, but the replacement is tricky. The first two terms are easy:

```
a = 1
b = 1
```

And the looped procedure is easy:

```
c = a + b
```

Bu how will you get the next term, and the next, without introducing any new variables?

Fibonacci Numbers, Revisited

```
a = 1      b = 1      c = a + b
```

We need to replace a and b with the next term:

`a = b` the value of b is now called “a”

`b = c` the value of c is now called “b”

and then repeat:

`c = a + b` the new value of c is a + b.

Practice Problem 3

3. Write a function including a “for” loop to generate the first *n* numbers in the Fibonacci sequence, including the first two. Use it to generate a list of the first 15 Fibonacci numbers.

While Loops

A `while` loop is used when you want to iterate until a certain condition is met, or when you know in advance how many loops to run. The format goes:

```
while (condition not met)
      (do a bunch of stuff)
end
```

While Loops with Conditions

The repeated square root of a positive number will always get closer to 1. Here is a function that will repeatedly take square roots until within 0.001 of 1:

```
function reproot(x)
    while abs(x - 1) > 0.001
        x = sqrt(x)
        println(x)
    end
end
```

While Loops with Counters

Let's say you wanted to know how many iterations it took for the reproot program to get to within 0.001 of 1. You could modify the program like this:

```
function reproot(x)
     n = 0
     while abs(x - 1) > .001
          x = sqrt(x)
          n = n + 1
     end
     println(n)
end
```

The number n is a counter – it just counts the loops.

While Loops with Counters, 2

Let's say you wanted to run the reproot program through 20 loops, then print the answer. This can also be done using counters:

```
function reproot(x)
     n = 0
     while n < 20
          x = sqrt(x)
          n = n + 1
     end
     println(x)
end
```

While Loops vs. For Loops

For a set number of iterations, you could use either “while” or “for”.

`for` loops are short and simple, but only work when you’re trying to increment by an even amount. You can make them increment by numbers other than 1 using `for n in 1:0.1:20`, for example.

`while` loops are a little longer, but will work even if you’re not incrementing evenly.

Practice Problem 4

4. Write a function, compound(P), that will count how many years it will take for an investment of P dollars, earning 5% interest, to grow over $1,000,000.

Don’t forget to test your code!

Extension Problems 5 and 6

5. Working in groups or on your own, write a function calcgame(x) that counts the number of iterations in the calculator game. Here are the rules again:

    if x is odd, multiply by 3 and add 1.

    if x is even, divide by 2.

    Stop when x = 1.

    Count the iterations.

Your function must include a “while” loop as well as an “if/else” test.

6. Use a “for” command to list the iterations of the calgame(x) function from x = 1 to x = 20.

Extension Problem 7

7. Working in groups or on your own, write a program to test if a number is prime.

Your output should be some version of, “n is prime” or “n is not prime”.

Don’t forget to test your code!