

<http://www.cs.cornell.edu/courses/cs1110/2020sp>

## Lecture 23: while Loops

(Sections 7.3, 7.4)

CS 1110  
Introduction to Computing Using Python

[E. Andersen, A. Bracy, D. Fan, D. Gries, L. Lee,  
S. Marschner, C. Van Loan, W. White]

### Recall: For Loops

```
for x in grades:
    print(x)
```

- **loop sequence:** grades
- **loop variable:** x
- **body:** print(x)

To execute the for-loop:

1. Check if there is a "next" element of **loop sequence**
2. If so:
  - **assign** next sequence element to **loop variable**
  - Execute all of **the body**
  - Go back to Line 1
3. If not, terminate execution<sub>2</sub>

### Different types of Repetition

1. Process each item in a sequence
  - Compute statistics for a dataset
  - Send all your contacts an email

```
for x in sequence:
    process x
```
2. Do something *n* times
  - Draw a checkers board
  - Run a protein-folding simulation for 10<sup>6</sup> time steps

```
for x in range(n):
    do something
```
3. Do something an unknown number of times
  - Play word guessing game until 6 strikes
  - Go in current direction until edge is detected

```
???
```

<https://www.flickr.com/photos/janirene/albums/72157642146435575/with/3058966193/>

### Beyond Sequences: The while-loop

```
while <condition>:
    statement 1
    ...
    statement n
```

body

- Relationship to **for-loop**
  - Broader notion of "keep working until done"
  - Must explicitly ensure condition becomes false
  - *You* explicitly manage what changes per iteration

4

### While-Loops and Flow

```
import random

num = random.randint(0,10)
guessed_it = False
print("I'm thinking of a number.")

while not guessed_it:
    guess = int(input('Guess it: '))
    guessed_it = (num == guess)
    print('Well done!')
```

I'm thinking of a number.  
Guess it: 6  
Guess it: 2  
Guess it: 1  
Guess it: 4  
Well done!

Continuation condition, not stopping condition

5

### Q1: What gets printed?

<pre>a = 0 while a &lt; 1:     a = a + 1  print(a)</pre>	<pre>a = 0 while a &lt; 2:     a = a + 1  print(a)</pre>	<pre>a = 0 while a &gt; 2:     a = a + 1  print(a)</pre>
<input type="text"/>	<input type="text"/>	<input type="text"/>

6

### Q2: What gets printed?

```
a = 4
while a > 0:
    a = a - 1
```

```
print(a)
```



```
a = 0
while a < 3:
    if a < 2:
        a = a + 1
```

```
print(a)
```



9

### Q3: What gets printed?

```
a = 8
b = 12
while a != b:
    if a > b:
        a = a - b
    else:
        b = b - a
print(a)
```

A: *Infinite loop*  
B: 8  
C: 12  
D: 4  
E: *I don't know*

This is Euclid's Algorithm for finding the greatest common factor of two positive integers.  
Trivia: It is one of the *oldest* recorded algorithms (~300 B.C.)

12

Start next video:  
while or for ?

14

### for vs. while

- You can almost always use either
- Sometimes **for** is better
  - Do something a **fixed** (pre-determined) number of times
- Sometimes **while** is better
  - Do something an **indefinite** (not infinite) number of times
  - E.g., do something until some event happens, i.e., **until a stopping condition is reached**

Called "definite iteration"

Called "indefinite iteration"

15

### for vs. while

do something n times

```
for k in range(n):
    # do something
```

```
k = 0
while k < n:
    # do something
    k = k+1
```

Must remember to increment

My preference? for-loop

16

### for vs. while

do something an unknown number of times

```
for k in range(??BIG_NUM):
    # do something
    if time to stop:
        break
```

```
while not time to stop:
    # do something
```

Do NOT use **break** in any work you submit in CS1110.  
Practice using **while**-loop in situations where **while**-loop is well suited

My preference? while-loop

17

## for vs. while

do something to each element of a sequence

```
for k in range(len(seq)):
    seq[k] = seq[k]+1

k = 0
while k < len(seq):
    seq[k] = seq[k]+1
    k = k+1
```

while is more flexible, but sometimes requires more code

My preference? for-loop

18

## for vs. while

do something until a limit is reached  
e.g., make a table of squares up to N

```
seq = []
sqn = math.floor(sqrt(N))
for k in range(sqn+1):
    seq.append(k*k)

seq = []
k = 0
while k*k < N:
    seq.append(k*k)
    k = k+1
```

for-loop requires you to know how many iterations you want **ahead of time**

can use complex expressions to check if a task is done

My preference? while-loop

19

## for vs. while

change a sequence's length  
e.g., remove all 3's for list nums

```
for i in range(len(nums)):
    if nums[i] == 3:
        del nums[i]

while 3 in nums:
    nums.remove(3)
```

IndexError: list index out of range

is this not beautiful?

My preference? while-loop

20

## for vs. while

find 1<sup>st</sup> n Fibonacci numbers

Fibonacci numbers:  
 $F_0 = 1$   
 $F_1 = 1$   
 $F_n = F_{n-1} + F_{n-2}$

```
fib = [1, 1]
for k in range(2,n):
    fib.append(fib[-1] + fib[-2])

fib = [1, 1]
while len(fib) < n:
    fib.append(fib[-1] + fib[-2])
```

Last item in list

Second-last item in list

loop variable not always **used**

loop variable not always **needed** at all

My preference? while-loop

21

## Using while-loops Instead of for-loops

### Advantages

- Better for **modifying data**
  - More natural than range
  - Works better with deletion
- Better for **convergent tasks**
  - Loop until calculation done
  - Exact steps are unknown
- Easier to **stop early**
  - Just set loop var (keep\_going) to False

### Disadvantages

- **Infinite loops** more likely
  - Easy to forget loop vars
  - Or get continuation condition wrong
- **Require** more management
  - Initialize the condition?
  - Update the condition?

22

Start next video:  
**How to set up a while loop**

23

## Setting up a while-loop

0. Situation is to do something until an event happens
1. Write the **continuation condition**
  - Create var names as necessary to express condition
  - May be easier to **negate stop** condition to get **continuation condition**
2. **Initialize loop vars** (vars in loop condition) as necessary
3. In loop body: **update loop vars**  
to possibly change loop condition from True to False
4. Write the rest of the loop body

24

## Improve number guessing game

```
import random
min_num= 1
max_num= 10
max_chances= 5
secret_num= random.randint(min_num, max_num)
print("I have a number from "+str(min_num)+" to "+str(max_num))
print("You have "+str(max_chances)+" chances to guess it")
```

**# User guesses until all chances used up or guessed correctly**

1. Allow fixed number of guesses

*For you to add later:*

2. If a guess is wrong, tell player whether it was too high or too low.

25