

Week 5

Intro to Python

Please sign-in:
cadtx.pw/week5

Objectives

Chapter 5 (part 2)

- For loops
- Nested loops
- Break/continue

Recap

Last week...

- While loops

- Counter-controlled

- Sentinel value controlled

- User confirmation controlled

For loops

A *for* loop iterates through each value in a sequence.

Last class we looked at a counter-controlled loop

```
data = eval(input("Enter an integer: "))
i = 0
# Read 4 more numbers, then stop
sum = data
while i < 4:
    data = eval(input("Enter an integer: "))

    sum += data

    i += 1

print("The sum is", sum)
```

Counter-controlled while loops \mapsto for loop

```
data = eval(input("Enter an integer: "))  
  
sum = data  
  
for i in range(4):  
    data = eval(input("Enter an integer: "))  
    sum += data  
  
print("The sum is", sum)
```

One important difference: (Hint: look at the loop condition)

i is NOT initialized before the loop condition!

However, *sum* still is.

Exercise 1

Convert while loop to a for loop.

```
i = 1
```

```
sum = 0
```

```
while sum < 10000:
```

```
    sum = sum + i
```

```
    i += 1
```

For loop syntax

```
for var in sequence:  
    # Loop body
```

range(a, b) returns sequence of integers $a, a+1, a+2, \dots, b-2, b-1$ (but NOT b).

range(b) is the same as range(0, b).

range(a, b, k) is the same as range(a, b) but with k as your step value

We can also move across elements in a sequence-type object (e.g. string, list).

```
sequence = ['cat', 'dog', 'ant']  
for var in sequence:  
    print(var)
```

What will this print?

Exercise 2

What's the output?

```
for i in range(1, 5):  
    j = 0  
    while j < i:  
        print(j, end = " ")  
        j += 1
```

```
i = 0  
while i < 5:  
    for j in range(i, 1, -1):  
        print(j, end = " ")  
    print("*****")  
    i += 1
```


String and lists example

```
st = 'Python'
l = ['P', 'y', 't', 'h', 'o', 'n']

for char in st:
    print(char)

for letter in l:
    print(letter)
```

```
# this output twice
>>> P
>>> y
>>> t
>>> h
>>> o
>>> n
```

Nested loops

A *nested loop* consists of an outer loop (m iterations) and inner loop(s) (n iterations). Each pass through the outer loop, all n passes through the inner loop are made.

Classic example: matrices

```
mat = [[2,3,4],[1,5,7],[8,9,6]]
```

```
for i in range(0,len(mat)):
    for j in range(0,len(mat[i])):
        if(i==j):
            print(mat[i][j])
```

```
for row in mat:
    for col in row:
        if(mat.index(row)==row.index(col)):
            print(col)
```

Syntax is similar to nested *if* statements.

Notice you can nest *for*, *while*, and *if* statements together.

Exercise 3

How would we print the first letter of each word in the following list using nested for loops:

```
sequence = ['cat', 'dog', 'ant']
```


Terminating loops

You can instantly terminate a loop using *break*.

The loop ends when the sum is greater or equal to 100.

Without *break*, the output would be the display the number as 20 and the sum as 210.

```
1      sum = 0
2      number = 0
3
4      while number < 20:
5          number += 1
6          sum += number
7          if sum >= 100:
8              break
9
10     print("The number is", number)
11     print("The sum is", sum)
```



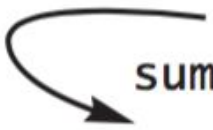
Number is 14 and sum is 105.

Terminating an iteration

Using *continue*, we can skip an iteration without ending the loop.

In this loop, the number 10 and 11 are not added to the sum.

```
1  sum = 0
2  number = 0
3
4  while number < 20:
5      number += 1
6      if number == 10 or number == 11:
7          continue
8      sum += number
9
10 print("The sum is", sum)
```



Exercise 4

What will happen in these loops?

```
balance = 1000
while True:
    if balance < 9:
        break
    balance = balance - 9
print("Balance is", balance)
```

```
balance = 1000
while True:
    if balance < 9:
        continue
    balance = balance - 9
print("Balance is", balance)
```

Multiplication Table

```
print("\tMultiplication Table")
print("  ",end='')
for j in range(1,10):
    print(" ",j,end='')
print()
print("-----")
# Display table body
for i in range(1,10):
    print(i,"|",end='')
    for j in range(1,10):
        # Display the product and align
        print(format(i*j,"4d"),end='')
    print()
```

	Multiplication Table								
	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

Exercise 5

```
# We are writing a program to check how long
# it will take for a bank account to grow to a
# desired value based on an annual interest rate

initB = eval(input("Enter the initial balance: "))
intRate = eval(input("Enter the interest rate (%): "))
desiredB = eval(input("Enter the final balance: "))

intRate = intRate/100

years = 0
# while loop? for loop?

# Example:
# Initial: $500, interest rate: 4%
# Math: after one year, the bank account will have
# grown to  $(500 * 1.04) = \$520$ 

print("Amount of years until desired balance:", years)
```