Programming in Python

For Loops

More Loops

- There are 2 types of loops in Python
  - A **`while`** loop repeats based on a condition
  - A **`for`** loop repeats based on a sequence

for Loop

- Use the reserved words **for** and **in**.

Indentation matters!

```
for variable in sequence:
    statement(s) (using variable)
    # there may be more statements
# code after the loop
```

```
counter = 0
while counter < 30:
    print(counter)
    counter = counter + 1
```

```
for counter in range(30):
    print(counter)
```

What is a sequence?

- A sequence is an "ordered set of things".
  - A group of items stored together in a collection
- Examples in Python
  - a "range" of numbers
  - string variables
  - lists (in future)

```
2, 4, 6, 8, 10
```

```
"Hello World"
```

```
["science", "technology", "engineering", "math"]
```

`range()` function

- The **`range()`** function will generate a sequence of numbers based on some parameters.
  - **`stop`** is an example of a parameter and allows us to pass information into a function.
  - This function returns a sequence of integers that start at 0, increase each time by 1, and count up to (but not including) the **`stop`** parameter.

```
range(stop)
```

```
for counter in range(6):
    print(counter)
```

```
0
1
2
3
4
5
```

`range()` function

- We can add a **`start`** parameter.
  - **start** must be before **stop**.

```
range(start, stop)
```

```
for counter in range(1, 11):
    print(counter)
```

```
1
2
3
4
5
6
7
8
9
10
```

`range()` function

- We can also add a **`step`** parameter.
  - **start** must be the first parameter.
  - **step** must be the last parameter.

```
range(start, stop, step)
```

```
for counter in range(10, 0, -1):
    print(counter)
```

```
10
9
8
7
6
5
4
3
2
1
```

Strings are sequences too!

- Sequences have order, and so do strings.

```
for variable in string:
    statement
    # there may be more statements
# code after the loop
```

```
for letter in "STEM":
    print(letter)
```

```
S
T
E
M
```

`for` Loop

- Use the reserved words **`for`** and **`in`**.

```
for variable in sequence:
    statement
    # there may be more statements
# code after the loop
```

```
for counter in range(30):
    print(counter)
```

```
for letter in "STEM":
    print(letter)
```

The `in` Operator

- Use **`in`** within a Boolean expression
  - › Returns **`True`** if the item is a member of the sequence.
- Branching

```
if item in sequence:
    statement
```

- While loop

```
while item in sequence:
    statement
```

Using the `in` Operator

- Branching and sequence is a string

```
if string1 in string2:
    statement
```

```
message = input("Enter a message: ")
subStr = input("Enter a string: ")
if subStr in message:
   print("Found")
else:
   print("NOT Found")
```

```
Enter a message: Be the sunshine!
Enter a string: sun
Found
```

Using the `in` Operator

- While loop and sequence is a string

```
while string1 in string2:
    statement
```

```
letter = input("Enter a vowel: ")
while letter not in "aeiou":
    letter = input("Enter a vowel: ")
print("You entered the vowel", letter)
```

```
Enter a vowel: b
Enter a vowel: q
Enter a vowel: t
Enter a vowel: e
You entered the vowel e
```

Using the `in` Operator

- Branching and use range() to get a sequence of ints

```
if intVar in range(start, stop, step):
    statement
```

```
num = int(input("Enter an integer: "))
if num in range(0, 11, 2):
    print("You entered an even number between 0 and 10")
```

```
Enter an integer: 6
You entered an even number between 0 and 10
```

len() Function

- The len() function returns the length of any sequence.

```
var = len(sequence)
```

```
message = "Happy Birthday!"
num = len(message)
print(message, "has", num, "characters.")
```

```
Happy Birthday! has 15 characters.
```

- What type of variable is num?

```
int
```

Nested Loops

- You can use loops inside of other loops.
- Let's use a for loop inside of a for loop.

```
for num in range(3):
  for letter in "abc":
    print(num, letter)
```

```
0 a
0 b
0 c
1 a
1 b
1 c
2 a
2 b
2 c
```

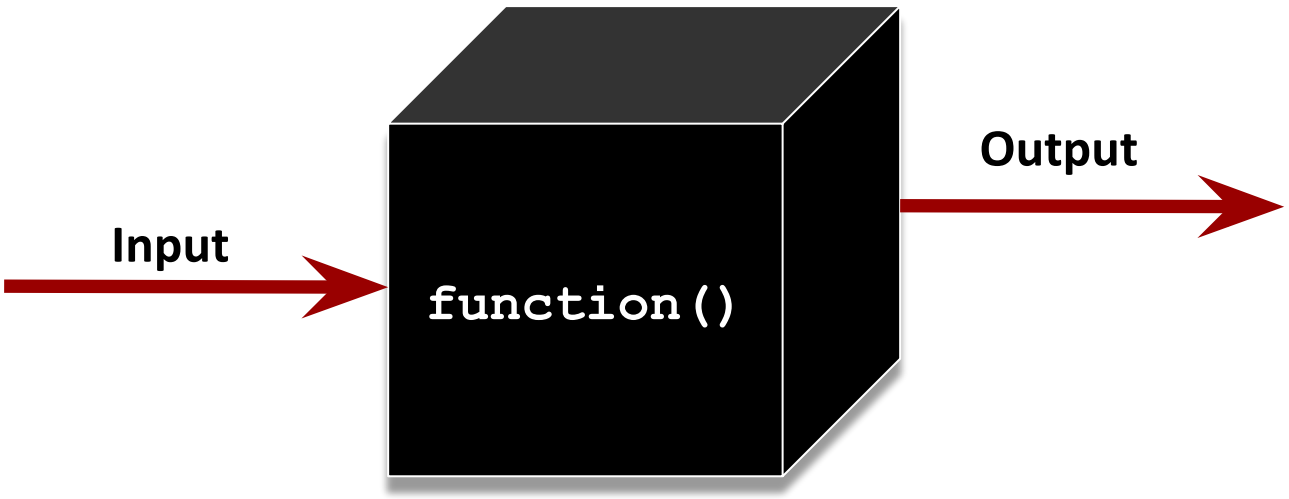
- Do you see a pattern?

Programming in Python

Modules

Functions

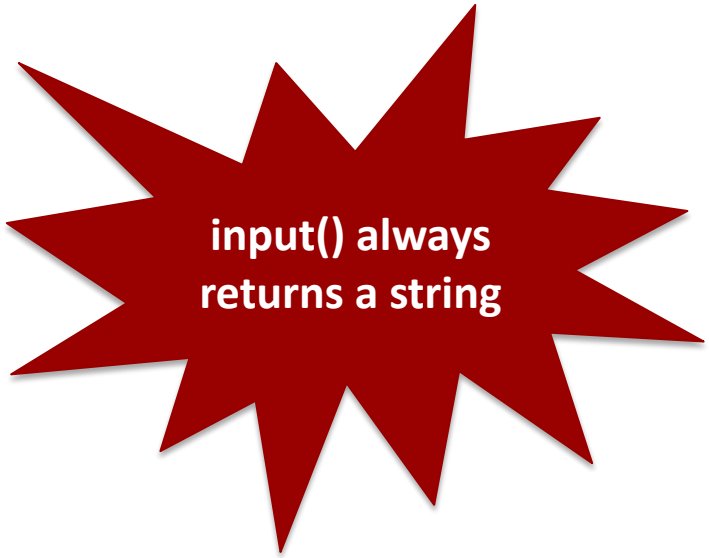
- Perform a task and then return control to your program
- Reuse code
  - Write once, use many times
- Many functions have been defined that we can use



Calling Functions

- Call input() function

```
str = input("Prompt: ")
```



- Since input() returns a string, need to create a variable
- Examples

```
name = input("Enter your name: ")
print("Hi, " + name + "!")
```

```
Enter your name: Quinn
Hi, Quinn!
```

Modules

- A module is a file containing Python definitions and statements.
- Many modules have been created by other developers with various functions that we want to use.
- To use the function, we have to tell PyCharm about them.
- We need to import a module.

```
import module
```

```
import random
```

- A module is NOT modulo, which is the math operator %.

Accessing Functions inside Modules

- To access a function, you must use the module name

```
module.functionName()
```

```
floatNum = random.random()
# float between 0.0 and 1.0
print(floatNum)

intNum = random.randrange(0, 10)
# int between 0 and 9 (inclusive)
print(intNum)
```

```
0.8293787843926647
6
```

Random Module

- Generates pseudo-random numbers
- Contains various functions that can be called

Function	Description
`choice(sequence)`	Returns a random element from the given sequence
`randint(start, stop)`	Returns a random number between the given range inclusive of the stop parameter
`randrange(start, stop)`	Returns a random number between the given range
`random()`	Returns a random float number between 0 and 1
`seed(a, version)`	Initialize the random number generator Both parameters are optional

- Need to import the random module at the top of your code
  - › Right below the top comment block

Random Examples

```
import random  # import the random module

random_num = random.randrange(10)
# random number between 0 and 9
print(random_num)

random_num = random.randrange(1, 11)
# random number between 1 and 10
# random number between 1 and 11 (exclusive)
print(random_num)

random_num = random.randrange(0, 101)
# random number between 0 and 100
# random number between 0 and 101 (exclusive)
print(random_num)
```

```
0
10
85
```

Random Examples

```
import random  # import the random module

random_num = random.randint(1, 10)
# random number between 1 and 10 (inclusive)
# random_num could equal 10
print(random_num)

random_char = random.choice("abcdef")
# random character from the "abcdef" string
print(random_char)

message = "hello world"
random_char = random.choice(message)
# random character from the message variable
# random_char could equal the space character
print(random_char)
```

```
8
d
l
```

Math Module

- Contains various mathematical functions

Function	Description
`fabs(x)`	Returns the absolute value of x
`sqrt(x)`	Returns the square root of x
`cos(x)`	Returns the cosine of x radians
`sin(x)`	Returns the sine of x radians
`tan(x)`	Returns the tangent of x radians

```
import math  # import the math module
num = math.sqrt(16)
print(num)
```

```
4.0
```