# While

# Question 1

What does the function **something** do?

```
def something(mystring,character):
        output = ""
        for i in range(len(mystring)):
                output += mystring[i]+character
        return output


print(something("OOOOO","X"))
```

**A. Prints some text.**

**B. Returns some text.**

# Question 2

What does this code print?

```
def something(mystring,character):
        output = ""
        for i in range(len(mystring)):
                output += mystring[i]+character
        return output


print(something("OOOOO","X"))
```

A. OOOOOXXXXX
B. OXOXOXOXOX
C. OOOOOX
D. OX

# Question 3

What does this code print?

```
def something2(mystring,character):
        output = ""
        for i in range(len(mystring)):
                output += mystring[i]+character
                return output


print(something2("OOOOO","X"))
```

A. OOOOOXXXXX
B. OXOXOXOXOX
C. OOOOOX
D. OX

# An Interactive Turtle

# Readings Review

Why might we use a **while** loop instead of a **for** loop?

If you want an action to repeat itself until a certain condition is met

Good for iterating through a list or sequence, or repeating the same code a fixed number of times.

# Question 4

What is the issue with the code below?

```
milkshakes = 10
while milkshakes > 0:
    milkshakes += 1
print(milkshakes)
```

# Question 5

How would you translate this comment into Python?

```
# As long as a is less than b, do XYZ
```

# Turtle Inter-action

```
1   # Interactive drawing turtle (one command only)
2   import turtle
3   anna = turtle.Turtle()
4
5   print("Commands accepted: (f)orward, (s)tamp")
6
7   # Get user's input
8   command = input("What should I do? [f, s]: ")
9
10  # Process the command
11  if command == "f":
12      anna.forward(100)
13  elif command == "s":
14      anna.stamp()
15  else:
16      print("Unknown command.")
17
18  turtle.exitonclick()
```

But this only asks once and then the program is over :(
What can we do?

# Looping Interaction

- Change the code to:
  - Loop until the user types "stop"
  - Create a variable named keep_looping

# The While Loop

Initialize Boolean

while <Boolean>:

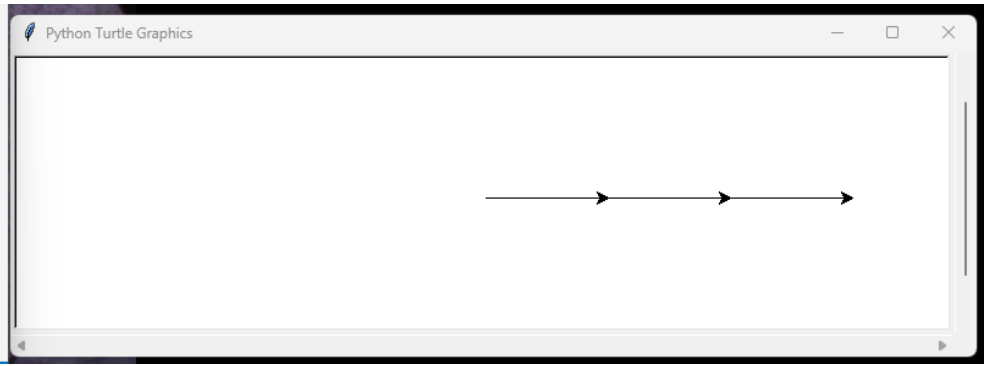
Update Boolean

Remember to indent!

```
1   # Interactive drawing turtle
2   import turtle
3   anna = turtle.Turtle()
4
5   print("Commands accepted: (f)orward, (s)tamp, (stop)")
6
7   # Setup our loop control variable
8   keep_looping = True
9   while keep_looping:
10
11      # Get user's input
12      command = input("What should I do? [f, s, stop]: ")
13
14      # Process the command
15      if command == "stop":
16          # If ending, update our loop controller
17          keep_looping = False
18      elif command == "f":
19          anna.forward(100)
20      elif command == "s":
21          anna.stamp()
22      else:
23          print("Unknown command.")
```

# Try it!

```
gpy\adapter/../..\debugpy\launcher' '56990' '-
-' 'c:\all-my-code\CMPT120-Code\ExWeek9\intera
ctive-turtle.py'
Commands accepted: (f)orward, (s)tamp, (stop)
What should I do? [f, s, stop]: f
What should I do? [f, s, stop]: s
What should I do? [f, s, stop]: f
What should I do? [f, s, stop]: s
What should I do? [f, s, stop]: f
What should I do? [f, s, stop]: s
What should I do? [f, s, stop]: stop
```



# While

**while** loop vs. **for** loop

```python
# Different ways of looping
# Angelica Lim
# March 5, 2021
# Print out all elements in a list,
# one element per line

vowels = ["a","e","i","o","u"]

# Use a for loop
for vowel in vowels:
    print(vowel)

# Use a for loop with range() #[0,1,2,3,4]
for i in range(len(vowels)):
    print(vowels[i])

# Use a while loop
i = 0
while i < len(vowels):
    print(vowels[i])
    i += 1
```

We saw this last time.

Beware! If you place the i+=1 **before** the print statement, you will get an error. Why?

# While to validate input

```
1  # While Loop Waiting for Valid Input
2  # Angelica Lim
3  # Feb. 24, 2021
4  # Asks whether you would like tea and keeps asking until
5  # you reply Y/y or N/n
6
7  # Initialize boolean variable
8  needs_reply = True
9
10 while needs_reply:
11   answer = input("Would you like tea (Y/N)? ").lower()
12   if answer == "y":
13     print("Here you go!")
14     needs_reply = False
15   elif answer == "n":
16     print("No worries, I'll ask again later.")
17     needs_reply = False
18   else:
19     print("Please reply Y or N :)")
```

```
Would you like tea (Y/N)? hmm
Please reply Y or N :)
Would you like tea (Y/N)? y
Here you go!
>
```

If you want an action to repeat itself until a certain condition is met

http://interactivepython.org/runestone/static/thinkcspy/MoreAboutIteration/SentinelValuesAndValidation.html

# More on loops

Reminder: the **`while`** statement can include any kind of Boolean expression.

See Diana Cukierman's excellent video here!

# While Summary

The **three** elements you need:

- **Initialize** a **control variable** before the while loop
- Write a **while** statement based on that **control variable**
- **Modify** the **control variable** in the while loop such that eventually the while loop will terminate

# Let's review some concepts

What are the components necessary to make a good while loop?

What is wrong with the code below?

```
i = 0

while i < 4:

  print(i)
```

# Exercise

Write a **Number Guessing** Game that randomly picks a number from 1 to 100. The game should continue to ask you to guess a number until your guess is equal to the randomly picked number.

If your guess is too low, the program should say “Higher!”. If the guess is too high, the bot should say “Lower!”

When you get it right, it should output “You got it!"

Also, you only get 8 tries.

Here is a sample run:

```
Please guess a number between 1-100! 50
Higher! Try again: 75
Higher! Try again: 83
Higher! Try again: 90
Higher! Try again: 95
Higher! Try again: 98
Higher! Try again: 99
You got it!
```

```
Please guess a number between 1-100! 1
Higher! Try again: 1
Higher! Try again: 1
Higher! Try again: 1
Higher! Try again: 1
Higher! Try again: 1
Higher! Try again: 1
Higher! Try again: 1
Higher! Try again: 1
Sorry, you ran out of guesses!
```

# While with 2 Conditions

```
1  # Number guessing game
2  # Angelica Lim
3  # Nov. 4, 2022
4  import random
5
6  # Choose a random number
7  secret_number = random.choice(range(1,101))
8
9  # Init control variables
10 guess = -1
11 guesses_left = 8
12
13 while guess != secret_number and guesses_left > 0:
14     # Ask user for a number
15     guess = int(input("Guess a number between 1-100: "))
16     guesses_left -= 1
17
18     # Depending on guess, say higher or lower
19     if guess < secret_number:
20         print("Higher!")
21     elif guess > secret_number:
22         print("Lower!")
23
24 if guess == secret_number:
25     print("You got it!")
26 else:
27     print("You ran out of guesses!")
28
```

Check loop continuation using control variables

Initialize control variables

Update control variables to make the while condition be False eventually!

Final result depending on which variable caused the loop to terminate