Programming in Python
Today…

We will:

✓ Re-cap Python constructs.
✓ Cover functions further.
✓ Program a reading game.
✓ Look at random numbers.
✓ Program “rock, paper, scissors” (if time).
✓ To finish: Editors
Re-cap
Editing Files

To Edit Files: **TextWrangler.**

‘Source code’ is stored in .py files, e.g. “program.py”
Create one file per program.

To Compile Files: **Terminal.**

“cd” – change directory, e.g. cd Desktop
“ls” – list files, e.g. ls Desktop
“python” – to run python, e.g. python MyProg.py

Notes:

• PyTHon iS CaSe SeNsItiVe.
• Comments: # comment.
## Printing

### Syntax

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print()</code></td>
<td>Print without any arguments.</td>
</tr>
<tr>
<td><code>print(value_1, value_2, ..., value_n)</code></td>
<td>Print all the given values, separated by blank spaces.</td>
</tr>
</tbody>
</table>

All arguments are optional. If no arguments are given, a blank line is printed.

```python
print("The answer is", 6 + 7, "!")
```

The values to be printed, one after the other, separated by a blank space.

---

Remember: No Brackets!
Variables

A variable is defined the first time it is assigned a value.

```
total = 0
  .
  .
  total = bottles * BOTTLE_VOLUME
      .
      .
      .
  total = total + cans * CAN_VOLUME
```

Names of previously defined variables

The expression that replaces the previous value

The same name can occur on both sides. See Figure 2.
Types of Variables

1) A whole number (no fractional part)  7 (integer)
2) A number with a fraction part  8.88 (float)
3) A sequence of characters  "Bob" (string)

Conversion examples:

String to Int: number1 = int("4")

String to Float: price = float("9.0")

String Concatenation: “Hello” + “World”
User Input

**CODE:**

```python
answer = raw_input("How old are you?")
print "You are", answer, "years old"
```

**OUTPUT:**

How old are you?

*user types 17*

You are 17 years old
Warning!

Remember to convert ints and floats!

Strings that look like numbers are not ints or floats!

For example:

```python
number = int(raw_input("Enter a number"))
number2 = float(raw_input("Enter a float"))
```
Conditional Statements

my_var = 37
if my_var < 50:
    print “It is less than 50!”
else if my_var < 80:
    print “It is more than 50 but less than 80!”
else:
    print “It is more than 80!”

Many comparisons, see notes from last week.

Condition operators: and, or, not
Loops: WHILE

**CODE:**

```python
counter = 0
while (counter < 10):
    print "The counter is", counter
    counter = counter + 1

print "The End."
```

**OUTPUT:**

```
0
1
2
3
4
5
6
7
8
9
```
Functions

Syntax

```python
def functionName(param1, param2, ...):
    statements
```

- **Function header**
  - Name of function
  - Name of parameter variable

- **Function body**, executed when function is called.

```python
def cubeVolume(sideLength):
    volume = sideLength ** 3
    return volume
```

- **Return statement**
  - Exits function and returns result.
Example: Name Age Function
Further with Functions
Main Functions

By convention, main is the starting point of the program.

```python
def main():
    result = cubeVolume(2)
    print("A cube with side length 2 has volume", result)

def cubeVolume(sideLength):
    volume = sideLength ** 3
    return volume

main()
```

The cubeVolume function is defined below.

This statement is outside any function definitions.
Recursive Functions

A function that calls itself. Tricky! 😊

Example:

```python
def sum(x):
    if (x == 0):
        return 0;
    else:
        return x + sum(x-1);
```

What does this function do? Consider: print `sum(5)`

Notice the “return” statement is important!
Function Tasks

Task 1:
Create a function for computing the volume of a cube. Then create a main function and call this function with different values. What happens if you enter very large numbers?

Task 2:
Write a “recursive” function that calls itself. The function should compute the factorial of a number! Then start this function from your main method.

Hint: factorial(n) = n * (n-1). Also, factorial(0) = 0.

Try calling the function with a large number. What happens?

Challenge Task:
Try to write factorial using a while loop and run it with large numbers. What happens?
Games
The Game

You're walking down a path and there are 3 paths to choose from. Which path do you take?

Choose a path...

OPTION 1

OPTION 2

OPTION 3
Example Function

def page1():
    answer = raw_input("You leave your house to begin your day, do you:
    a) Take the car,
    b) Walk,
    c) catch the bus.")

    if answer == "a":
        page2()
    elif answer == "b":
        page3()
    else:
        page4()
Game Tasks

Task 1:
Create your own games with a storyline and series of pages! Think about the ending?
Hint: write one function per page.

Challenge Task:
Introduce a score for your game: better choices score more? Bad choices subtract from score? Hint: use a global variable for the score (put the variable at the top of the file).
A Random Library

To make games more interesting, people often make the computer a player.

To do this for our next game we will use randomly generated numbers. These can be obtained from a library!

Example:

```python
from random import randint

x = randint(5,10)
print(x)
```
Rock, Paper, Scissors

A) Write a program that asks the user to select a number (from main function!):
   1. Rock
   2. Paper
   3. Scissors
   Store their choice for later.

B) Randomly generate a choice for the computer using the `randint` function. Store the number for later.

C) Implement a function (2 parameters) that computes the game winner:
   -- Rock beats scissors,
   -- Scissors beats paper,
   -- Paper beats rock.

D) Print whether the result is a win, lose or draw for the user.
Editors: Idle, Eclipse etc.
Summary

Today we have looked at combining constructs to form a larger program.

Further coding: try code academy?

http://www.codecademy.com/tracks/python

Next Week: Lists and storing data.
Homework (as you asked for it)

1. Please let me know of a few examples of large programs you would like to create towards the end of the course.

2. Write a program that prompts the user for a integer $n$. The program should then print a hollow square of asterisks $n$ by $n$ in size.

For example: if the user enters 5, print:

```
*****
*   *
*   *
*   *
*****
```