Basic Python ¶

As you learn Python throughout this course, there are a few things you should keep in mind. Python is case sensitive. Spacing is important. Use error messages to help you learn.

In [ ]:
```

```

Data Types and Operators

As we know that data is classified into different so they are called data types they are very importand for any programming language.
Data Types: Integers, Floats, Booleans, Strings, Lists, Tuples, Sets, Dictionaries
Operators: Arithmetic, Assignment, Comparison, Logical, Membership, Identity
Built-In Functions, Compound Data Structures, Type Conversion

Arithmetic Operators

```
+ Addition
- Subtraction
* Multiplication
/ Division
% Mod (the remainder after dividing)
** Exponentiation (note that ^ does not do
 this operation, as you might have seen in
 other languages)
// Divides and rounds down to the nearest i
nteger
```

In [ ]:
```
5+3
```

In [ ]:
```
5-3
```

In [ ]:
```
5 * 3
```

In [ ]:
```
5/3
```

In [ ]:
```
5 % 3
```

In [ ]:
```
5**3 #5^3
```

In [ ]:
```
5//3
```

```
()  Parentheses
()  Parentheses
**  Exponentiation
*   Multiplication
/   Division
+   Addition
–   Subtraction
```

In [ ]:
```
15 + 4
```

In [ ]:
```
5+3**2
```

Variables

They hold some space in memory and are referenced with the name they are given
Besides writing variable names that are descriptive, there are a few things to watch out for when naming variables in Python.

1. Only use ordinary letters, numbers and underscores in your variable names. They can’t have spaces, and need to start with a letter or underscore.
2. You can’t use reserved words or built-in identifiers that have important purposes in Python, which you’ll learn about throughout this course.

In [ ]:
```
x = 3
y = 4
z = 5
```

In [ ]:
```
z
```

In [ ]:
```
x, y, z = 3, 4, 5
```

In [ ]:
```
# correct way of intializing the vari
ables
myHeight = 58
my_height = 58
my_lat = 40
my_long = 105
```

Integers and Floats

In [ ]:
```
x = 4.7
y = 4
```

In [ ]:
```
type(x)
```

In [ ]:
```
print(type(x))
print(type(y))
```

In [ ]:
```
x = int(4.7)    # x is now an integer
```

```
 4
y = float(4)   # y is now a float of
 4.0
```

In [ ]:
```
y
```

In [ ]:
```
print(type(x))
print(type(y))
```

In [ ]:
```
print(.1 + .1 + .1 == .3)
#Because the float, or approximation,
for 0.1 is actually slightly more tha
n 0.1, when we add several of them to
gether we can see the difference betw
een the mathematically correct answer
and the one that Python creates.
```

In [ ]:
```
.1 + .1 + .1
```

Boolean Comparison and Logical Operators

The bool data type holds one of the values True or False, which are often encoded as 1 or 0, respectively. 5 < 3 False Less Than
5 > 3 True Greater Than
3 <= 3 True Less Than or Equal To
3 >= 5 False Greater Than or Equal To
3 == 5 False Equal To
3 != 5 True Not Equal To
Logical Use Bool Operation
5 < 3 and 5 == 5 False and - Evaluates if all provided statements are True
5 < 3 or 5 == 5 True or - Evaluates if at least one of many statements is True
not 5 < 3 True not - Flips the Bool Value

In [ ]:
```
5 == 3
```

In [ ]:
```
5 < 3 or 5 == 5
```

In [ ]:
```
not 5 < 3
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Strings

Strings in Python are shown as the variable type str. You can define a string with either double quotes " or single quotes '. If the string you are creating actually has one of these two values in it, then you need to be careful to assure your code doesn't give an error.

In [10]:
```
my_string = 'this is a \t string!'
print(my_string)
```
```
this is a        string!
```

In [7]:
```
this is a string's!
```

In [8]:
```
my_string2 = "this is also a strin
g!!!"
this_string = 'Simon\'s skateboard is
in the garage.'
print(this_string)
```
```
Simon's skateboard is in the garage.
```

In [ ]:

In [14]:
```
first_word = 'Hello'
second_word = 'There'
print(first_word +" "+ second_word)
```
```
Hello There
```

In [15]:
```
print(first_word * 5)
```
```
HelloHelloHelloHelloHello
```

In [16]:
```
print(len(first_word))
```
```
5
```

In [17]:
```
# Unicode Character 'DEVANAGARi'
name = u'\u0915\u094b\u0930 \u092a\u0
948\u0925\u0964\u0928'
print(name)
```
```
कोर पैथ।न
```

String Methods

In [ ]:
```
my_string = "Tanishka"
```