# rrPython Basics Tutorial

Go through the Lean the Basics section at <https://www.learnpython.org/> Run each code block in a separate cell in this Jupyter Notebook. An easy shortcut is to hit Shift-Enter after entering your code to run the code in the cell and create a new cell below.

In [1]:

print("hello world!")

hello world!

In [2]:

myint =7

print(myint)

7

In [3]:

myfloat = 7.0

print(myfloat)

myfloat = float(7)

print(myfloat)

7.0

7.0

In [4]:

mystring = "Don't worry about apostrophes"

print(mystring)

Don't worry about apostrophes

In [5]:

one = 1

two = 2

three = one + two

print(three)

hello = "hello"

world = "world"

helloworld = hello + " " + world

print(helloworld)

3

hello world

In [6]:

a, b = 3, 4

print(a,b)

3 4

In [7]:

*#changes to code*

mystring = "hello"

myfloat = 10.0

myint = 20

*# testing code*

**if** mystring == "hello":

print("String: **%s**" % mystring)

**if** isinstance(myfloat, float) **and** myfloat == 10.0:

print("Float: **%f**" % myfloat)

**if** isinstance(myint, int) **and** myint == 20:

print("Integer: **%d**" % myint)

String: hello

Float: 10.000000

Integer: 20

In [8]:

mylist = []

mylist.append(1)

mylist.append(2)

mylist.append(3)

print(mylist[0]) *# prints 1*

print(mylist[1]) *# prints 2*

print(mylist[2]) *# prints 3*

*# prints out 1,2,3*

**for** x **in** mylist:

print(x)

1

2

3

1

2

3

In [9]:

numbers = []

strings = []

names = ['John', 'Eric', 'Jessica']

numbers.append(1)

numbers.append(2)

numbers.append(3)

strings.append('Hello')

strings.append('World')

second\_name = names[1]

print(numbers)

print(strings)

print("The second name on the names list is **%s**" % second\_name)

[1, 2, 3]

['Hello', 'World']

The second name on the names list is Eric

In [10]:

x = 'hello'

y = 'world'

*# TODO: change this code*

x\_list = ([x] \* 10)

y\_list = ([y] \* 10)

big\_list = x\_list + y\_list

print("x\_list contains **%d** objects" % len(x\_list))

print("y\_list contains **%d** objects" % len(y\_list))

print("big\_list contains **%d** objects" % len(big\_list))

*# testing code*

**if** x\_list.count(x) == 10 **and** y\_list.count(y) == 10:

print("Almost there...")

**if** big\_list.count(x) == 10 **and** big\_list.count(y) == 10:

print("Great!")

x\_list contains 10 objects

y\_list contains 10 objects

big\_list contains 20 objects

Almost there...

Great!

In [11]:

data = ("John", "Doe", 53.44)

format\_string = "Hello"

print("Hello, **%s** **%s**. Your current balance is $**%s**" % data)

Hello, John Doe. Your current balance is $53.44

In [12]:

data = ("John", "Doe", 53.44)

format\_string = "Hello **%s** **%s**. Your current balance is $**%s**."

print(format\_string % data)

Hello John Doe. Your current balance is $53.44.

In [13]:

astring = "Strings are awesome!"

s = astring

*# Length should be 20*

print(len(astring))

*# First occurrence of "a" should be at index 8*

print(astring.index("a"))

*# Number of a's should be 2*

print(astring.count("a"))

*# Slicing the string into bits*

print(astring[ :5]) *# Start to 5*

print(astring[5:10]) *# 5 to 10*

print(astring[12:12]) *# Just number 12*

print(astring[1::2]) *#(0-based indexing)*

print(astring[-5:]) *# 5th-from-last to end*

*# Convert everything to uppercase*

print(astring.upper())

*# Convert everything to lowercase*

print(astring.lower())

*# Check how a string starts*

**if** astring.startswith("Str"):

print("String starts with 'Str'. Good!")

*# Check how a string ends*

**if** astring.endswith("ome!"):

print("String ends with 'ome!'. Good!")

*# Split the string into three separate strings,*

*# each containing only a word*

print(astring.split())

20

8

2

Strin

gs ar

tig r wsm!

some!

STRINGS ARE AWESOME!

strings are awesome!

String starts with 'Str'. Good!

String ends with 'ome!'. Good!

['Strings', 'are', 'awesome!']

In [14]:

s = "Strings are awesome!"

*# Length should be 20*

print("Length of s = **%d**" % len(s))

*# First occurrence of "a" should be at index 8*

print("The first occurrence of the letter a = **%d**" % s.index("a"))

*# Number of a's should be 2*

print("a occurs **%d** times" % s.count("a"))

*# Slicing the string into bits*

print("The first five characters are '**%s**'" % s[:5]) *# Start to 5*

print("The next five characters are '**%s**'" % s[5:10]) *# 5 to 10*

print("The thirteenth character is '**%s**'" % s[12]) *# Just number 12*

print("The characters with odd index are '**%s**'" %s[1::2]) *#(0-based indexing)*

print("The last five characters are '**%s**'" % s[-5:]) *# 5th-from-last to end*

*# Convert everything to uppercase*

print("String in uppercase: **%s**" % s.upper())

*# Convert everything to lowercase*

print("String in lowercase: **%s**" % s.lower())

*# Check how a string starts*

**if** s.startswith("Str"):

print("String starts with 'Str'. Good!")

*# Check how a string ends*

**if** s.endswith("ome!"):

print("String ends with 'ome!'. Good!")

*# Split the string into three separate strings,*

*# each containing only a word*

print("Split the words of the string: **%s**" % s.split(" "))

Length of s = 20

The first occurrence of the letter a = 8

a occurs 2 times

The first five characters are 'Strin'

The next five characters are 'gs ar'

The thirteenth character is 'a'

The characters with odd index are 'tig r wsm!'

The last five characters are 'some!'

String in uppercase: STRINGS ARE AWESOME!

String in lowercase: strings are awesome!

String starts with 'Str'. Good!

String ends with 'ome!'. Good!

Split the words of the string: ['Strings', 'are', 'awesome!']

In [15]:

*# change this code*

number = 20

second\_number = 0

first\_array = [1,2]

second\_array = [1,2,3]

**if** number > 15:

print("1")

**if** first\_array:

print("2")

**if** len(second\_array) == 3:

print("3")

**if** len(first\_array) + len(second\_array) == 5:

print("4")

**if** first\_array **and** first\_array[0] == 1:

print("5")

**if** **not** second\_number:

print("6")

1

2

3

4

5

6

In [16]:

*# change this code*

number = 16

second\_number = 0

first\_array = [1,2,3]

second\_array = [1,2]

**if** number > 15:

print("1")

**if** first\_array:

print("2")

**if** len(second\_array) == 2:

print("3")

**if** len(first\_array) + len(second\_array) == 5:

print("4")

**if** first\_array **and** first\_array[0] == 1:

print("5")

**if** **not** second\_number:

print("6")

1

2

3

4

5

6

In [17]:

numbers = [951, 402, 984, 651, 360, 69, 408, 319, 601, 485, 980, 507, 725, 547, 544, 615, 83, 165, 141, 501, 263, 617, 865, 575, 219, 390, 984, 592, 236, 105, 942, 941, 386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615, 953, 345, 399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687, 217, 815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742, 717, 958, 609, 842, 451, 688, 753, 854, 685, 93, 857, 440, 380, 126, 721, 328, 753, 470, 743, 527]

**for** number **in** numbers:

**if** number == 237:

**break**

**if** number % 2 == 1:

**continue**

print(number)

402

984

360

408

980

544

390

984

592

236

942

386

462

418

344

236

566

978

328

162

758

918

In [26]:

*# Modify this function to return a list of strings as defined above*

**def** list\_benefits():

**return** "More organized code", "More readable code", "Easier code reuse", "Allowing programmers to share and connect code together"

*# Modify this function to concatenate to each benefit - " is a benefit of functions!"*

**def** build\_sentence(benefit):

**return** "**%s** is a benefit of functions!" % benefit

**def** name\_the\_benefits\_of\_functions():

list\_of\_benefits = list\_benefits()

**for** benefit **in** list\_of\_benefits:

print(build\_sentence(benefit))

name\_the\_benefits\_of\_functions()

More organized code is a benefit of functions!

More readable code is a benefit of functions!

Easier code reuse is a benefit of functions!

Allowing programmers to share and connect code together is a benefit of functions!

In [27]:

*# define the Vehicle class*

**class** **Vehicle**:

name = ""

kind = ""

color = ""

value = 100.00

**def** description(self):

desc\_str = "**%s** is a **%s** **%s** worth $**%.2f**." % (self.name, self.color, self.kind, self.value)

**return** desc\_str

car1 = Vehicle()

car1.name = "Super Duty"

car1.color = "Grey"

car1.kind = "truck"

car1.value = 80000.00

car2 = Vehicle()

car2.name = "Ram"

car2.color = "white"

car2.kind = "truck"

car2.value = 60000.00

*# test code*

print(car1.description())

print(car2.description())

Super Duty is a Grey truck worth $80000.00.

Ram is a white truck worth $60000.00.

In [28]:

phonebook = {

"John" : 938477566,

"Jack" : 938377264,

"Jill" : 947662781,

"Jake" : 938273443

}

phonebook.pop("Jill")

*# testing code*

**if** "Jake" **in** phonebook:

print("Jake is listed in the phonebook.")

**if** "Jill" **not** **in** phonebook:

print("Jill is not listed in the phonebook.")

Jake is listed in the phonebook.

Jill is not listed in the phonebook.

In [29]:

phonebook = {

"John" : 938477566,

"Jack" : 938377264,

"Jill" : 947662781

}

*# write your code here*

phonebook["Jake"] = 938273443

**del** phonebook["Jill"]

*# testing code*

**if** "Jake" **in** phonebook:

print("Jake is listed in the phonebook.")

**if** "Jill" **not** **in** phonebook:

print("Jill is not listed in the phonebook.")

Jake is listed in the phonebook.

Jill is not listed in the phonebook.

In [30]:

**import** **re**

find\_members = []

**for** member **in** dir(re):

**if** "find" **in** member:

find\_members.append(member)

print(sorted(find\_members))

['findall', 'finditer']

In [ ]: