

# Learning Objectives Chapter 2

March 5, 2020

## 1 Learning Objectives for Chapter 2

To prepare for the test on this chapter, you should verify that you are able to do each of the following. Create code and/or markdown cells below each objective to demonstrate your skill.

### 1.1 Learning Objective 1

Observe a constant and determine its type when the type is one of string, int, and float.

“123” is a string since it is enclosed in quotes. 123 is a int since it is not inclosed in quotes. 12.3 is a float since it involves a decimal point.

```
[1]: type(12.3)
```

```
[1]: float
```

```
[2]: type(123)
```

```
[2]: int
```

```
[3]: type("123")
```

```
[3]: str
```

### 1.2 Learning Objective 2

Predict the type of a variable based on the expression used to define it.

```
[4]: p = 11  
q = 121  
r = q/p  
print(r)
```

```
11.0
```

The variable “r” is a float because it has a decimal paoint.

### 1.3 Learning Objective 3

Use the `type()` function to determine the type of a constant or variable.

```
[5]: type(123)
```

```
[5]: int
```

### 1.4 Learning Objective 4

Use the functions `int()`, `float()` and `str()` appropriately.

```
[6]: int(12.3)
```

```
[6]: 12
```

```
[7]: str(123)
```

```
[7]: '123'
```

```
[8]: float("123")
```

```
[8]: 123.0
```

### 1.5 Learning Objective 5

Use and explain the two different methods of dividing a pair of integers.

To get a floating result with a decimal, use a single `/`.

```
[9]: x = 2/5  
print(x)  
print(type(x))
```

```
0.4
```

```
<class 'float'>
```

To get a quotient use `//` and to get the remainder use `%`.

```
[10]: y = 2//5  
z = 2%5  
print(y)  
print(z)
```

```
0
```

```
2
```

## 1.6 Learning Objective 6

Use and explain the rules of precedence of operators for numerical expressions.

Depending on where the operators are located numerical expressions will execute differently.

```
[11]: x = 1 + 3 * 5
      print(x)
      x = 1 * 3 + 5
      print(x)
```

16

8

## 1.7 Learning Objective 7

Describe the consequences of including a comma within a number.

```
[12]: x = 1,000,000
      print(x)
      print(type(x))
      y = 1000000
      print(y)
```

(1, 0, 0)

<class 'tuple'>

1000000

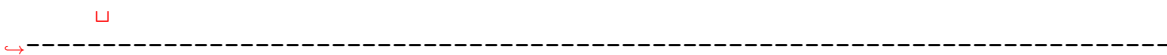
## 1.8 Learning Objective 8

Use the operator + to concatenate strings.

```
[13]: First = "Tyler "
      Last = "Moore "
      FullName = First + Last
      print(FullName)
```

Tyler Moore

```
[14]: First = "Tyler "
      Age = 26
      Full = First + age
      print(Full)
```



```

NameError                                Traceback (most recent call
↳last)

<ipython-input-14-f529fee40259> in <module>
      1 First = "Tyler "
      2 Age = 26
----> 3 Full = First + age
      4 print(Full)

NameError: name 'age' is not defined

```

## 1.9 Learning Objective 9

Prompt a user for input and convert the result to an integer or float if necessary.

```

[15]: xstr = input("Give me a number")
      xnum = float(xstr)
      print(xnum)

```

Give me a number

```

↳-----

```

```

ValueError                                Traceback (most recent call
↳last)

<ipython-input-15-b6237134ddab> in <module>
      1 xstr = input("Give me a number")
----> 2 xnum = float(xstr)
      3 print(xnum)

ValueError: could not convert string to float:

```

## 1.10 Learning Objective 10

Describe the response of the python interpreter when a programmer tries to use a reserved word as a variable name.

else is a reserved variable name

```

[16]: else = 3

```

```
File "<ipython-input-16-47c7b804d5e4>", line 1
else = 3
    ^
SyntaxError: invalid syntax
```

### 1.11 Learning Objective 11

Give an example of a non-mnemonic variable name and a mnemonic alternative.

### 1.12 Learning Objective 12

List the three components of an assignment statement and describe the result of executing one.

1. Variable name is located on the left.
2. A single equal sign assigns the variable a expression.
3. Some kind of expression goes to the left of the “=”.

```
[0]: Variable = "Some kind of expression"
```

### 1.13 Learning Objective 13

Distinguish between valid and invalid variable names