C24 - Basic Syntax

author & date

- author: Akshar Patel
- date: 4/11/2022

Variable assignment

```
#R
tomato <- 1

#Python
tomato = 1
```

```
---------------------------------------------------------------------------
NameError                                 Traceb
<ipython-input-43-2f4fba34dbf0> in <module>()
      1 #R
----> 2 tomato <- 1
      3
      4 #Python
      5 tomato = 1

NameError: name 'tomato' is not defined
```

SEARCH STACK OVERFLOW

Akshar Patel
6:12 PM Today
Resolve

Shows difference in between R and Python

```
#R
watermelon <- "popular in summer"

#Python
watermelon = "popular in summer"


x, y, z = "apple", "water", "house"

print(x, y, z)
```

```
apple water house
```

Indentation

```
# correct

if 5 > 2:
  print("Five is greater than two!")
```

```
Five is greater than two!
```

```
#incorrect

if 5 > 2:
print("Five is greater than two!")
```

Data types

- Text type: str
- Numeric type: int, float
- Sequence type: list, tuple, range, dict
- Boolean type: bool (True, False)
- command to check data type: type()

```
#String

watermelon = "popular in summer"
type(watermelon)
```

```
str
```

```
#Strings can be indexed

print(watermelon[0])
print(watermelon[11])
print(watermelon[-1]) #to start counting from the right
```

```
p
s
r
```

```
#Strings can be sliced

print(watermelon[0:8]) #from position 0 (included) to 8 (excluded)
print(watermelon[:4]) #from the beginning to position 4 (excluded)
print(watermelon[4:]) #from position 4 (included) to the end
```

```
popular
popu
lar in summer
```

```
#int

type(123)
```

```
int
```

```
#float

type(4.567)
```

```
float
```

List

- A list of comma-separated items between square brackets [ ]
- Lists might contain items of different types
- Mutable: it is possible to change their content

```
#Sequence type: list

tomatos = [1, 4, 52, 12]
```

```
print(tomatos)
```

```
[1, 4, 52, 12]
```

```
type(tomatos)
```

```
list
```

```
# A list can be indexed and sliced

print(tomatos[0])
print(tomatos[0:3])
print(tomatos[:])
```

```
1
[1, 4, 52]
[1, 4, 52, 12]
```

```
#if you want  to replace the wrong value

tomatos[3] = "red"
print(tomatos)
```

```
[1, 4, 52, 'red']
```

```
# using append

tomatos.append("green")
print(tomatos)
```

```
[1, 4, 52, 'red', 'green']
```

```
# remove items using blank square brackets []

tomatos[3:5] = []
print(tomatos)
```

```
[1, 4, 52]
```

Tuple

- A list of comma-separated items between parentheses ( )
- Immutable: it is not possible to change their content directly

```
price = (2, 4, 5)
```

```
type(price)
```

```
tuple
```

Range

- Structure: (`start`, `stop`, `step`)
- Commonly used for looping a specific number of times in for loops
- Immutable sequence of numbers

`start`

> The value of the start parameter (or 0 if the parameter was not supplied) (included)

`stop`

> The value of the stop parameter (excluded)

`step`

> The value of the step parameter (or 1 if the parameter was not supplied)

```
range(10)
```

```
range(0, 10)
```

```
list(range(10))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
list(range(0, 30, 5))
```

```
[0, 5, 10, 15, 20, 25]
```

```
r = range(0, 20, 2)
```

```
list(r)
```

```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

```
r.index(10) #the position at the first occurrence of the specified value (10)
```

```
5
```

Dict

- Dictionary
- Mapping object with braces like {'iphone': 100, 'ipad': 200, 'imac': 300}
- Structure: {key: value}
- Mutable

```
#first method

apple = {'iphone': 100, 'ipad': 200, 'imac': 300}


print(apple)
type(apple)
```

```
{'iphone': 100, 'ipad': 200, 'imac': 300}
dict
```

```
#second method

apple2 = dict(iphone=100, ipad=200, imac=300)


print(apple2)
type(apple2)
```

```
{'iphone': 100, 'ipad': 200, 'imac': 300}
dict
```

```
#return values only

apple.values()
```

```
dict_values([100, 200, 300])
```

```
#return keys only

apple.keys()
```

```
dict_keys(['iphone', 'ipad', 'imac'])
```

```
# return a value using a key

apple["iphone"]
```

```
100
```

```
# length of items
```

```
len(apple)
```

```
3
```

```
# update (1)

apple.update({"imac studio": 400})
```

```
print(apple)
```

```
{'iphone': 100, 'ipad': 200, 'imac': 300, 'imac
```

```
# update (2)

apple["magic mouse"] = 50
print(apple)
```

```
{'iphone': 100, 'ipad': 200, 'imac': 300, 'imac
```

```
# remove (1)

apple.pop("magic mouse")
print(apple)
```

```
{'iphone': 100, 'ipad': 200, 'imac': 300, 'imac
```

```
del apple["iphone"]
print(apple)
```

```
{'ipad': 200, 'imac': 300, 'imac studio': 400}
```

loop of dict

```
#only keys (1)

for i in apple:
  print(i)
```

```
ipad
imac
imac studio
```

```
#only keys (2)

for i in apple.keys():
  print(i)
```

```
ipad
imac
imac studio
```

```
#only values(1)

for i in apple:
  print(apple[i])
```

```
200
300
400
```

```
#only values(2)

for i in apple.values():
  print(i)
```

```
200
300
400
```

```
#keys and values

for i in apple.items():
  print(i)
```

```
('ipad', 200)
('imac', 300)
('imac studio', 400)
```

```
import os
os.getcwd()
```

```
'/content'
```

0s completed at 6:13 PM