enactus
University of Sheffield

CODE CREATORS
SHEFFIELD

2024 PYTHON BEGINNER COURSE

Lesson 5

SHALEV LAU
/in/shalevl
/mushroomhater07

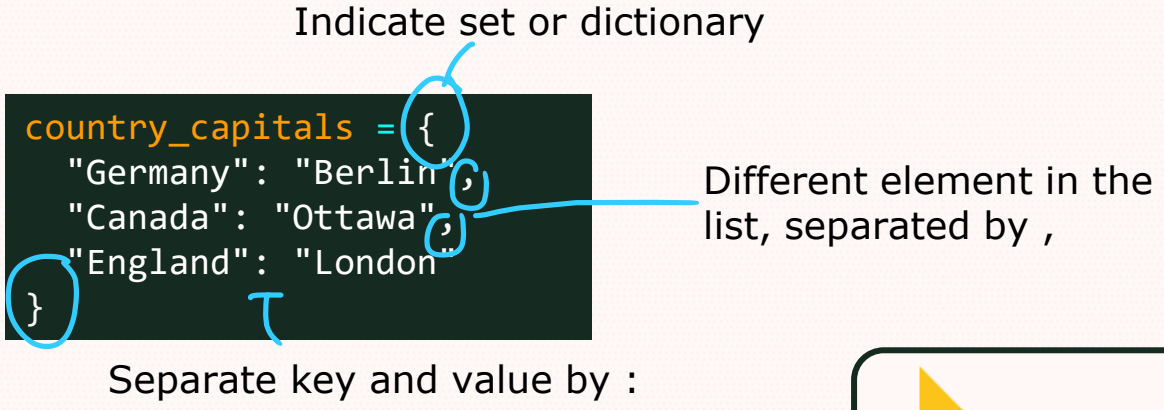
Functions

Dictionary
Function
Parameter & Argument
Scope
Return
Recursion
VS infinite loop

Dictionary

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered, changeable and do not allow duplicates.



enactus University of Sheffield

CODE CREATORS SHEFFIELD

Dictionary operation

```
country_capitals = {}
}
```

Access

```
print(country_capitals["Germany"])

Bestplace = country_capitals["England"])
```

```
# Berlin
```

```
Bestplace = London
```

Add item

```
country_capitals["Italy"] = "Rome"
```

Delete item

```
del country_capitals["Canada"]
```

Delete all items

```
country_capitals.clear()
```

enactus University of Sheffield

CODE CREATORS SHEFFIELD

DT comparsion

Collection data type	Ordered	Allowing duplicates	Changeable	When useful
List	Yes	Yes	Yes	When you care about exact order of elements `zoo = ["panda", "zebra", "panda", "monkey"]`
Set	No	No	Yes	When you just need to know what elements are in the zoo `animals = {"panda", "monkey", "zebra"}`
Tuple	Yes	Yes	No	When elements are unchangeable `must_have = ("elephant", "zebra", "lion")`
Dictionary	No	Keys - unique Values - may be duplicated	Yes	When you want to store the relations between some elements: `no_of_animals = { "panda":2, "zebra":1, "monkey":1 }`

enactus University of Sheffield

CODE CREATORS SHEFFIELD

What is functions?

```
print("Hello world!")
```

```
len("word")
```

```
arr = [1, 2, 3, 4, 5]
sum(arr)
```

```
import time
time.sleep(1)
```

Fun Fact: You've already used several functions without even knowing it.

enactus University of Sheffield

CODE CREATORS SHEFFIELD

Functions

‘Out of line’ block of code that may be executed (called) by simply writing its name in a program statement.

can be called **with** or **without parameters**.

enactus University of Sheffield

CODE CREATORS SHEFFIELD

Creating Functions - def

Make sure:

def keyword to define

: after a statement

4 spacebar in front of a statement

```
def hello():
    name = input("Enter your name")
    print('Nice to meet you, ', name)

hello()
```

Calling statement after define (only in Python)

() indicate function

Parameters

Share variable/ value between function and your main code

Set default parameter
(Not necessary but can be prevenet error)

Default parameter

```
def multiply(a=2, b=4):
    num = a * b
    print("Result:", num)

multiply(a=5)
```

argument

enactus
University of Sheffield

CODE CREATORS
SHEFFIELD

Scope - global scope

```
def random():
    num = 5

random()
print(num)
```

Local scope: variable is available only in one part of the program.

```
def random():
    global num
    num = 5

random()
print(num)
```

```
num = 0;

def random():
    num = 5

random()
print(num)
```

Global scope: variable is available within any scope, so can be used anywhere in the program

enactus University of Sheffield

CODE CREATORS SHEFFIELD

Scope - naming issues

Why not use global at the whole time?

Global scope: variable is available within any scope, so can be used anywhere in the program

Used up memory
Affect other program/function

```
num = 0;

def random():
    num = 5
def add(add1, add2):
    num = add1 + add2

random()
add(5,5)
print(num)
```

enactus University of Sheffield

CODE CREATORS SHEFFIELD

Return

```
def add(num1, num2):
    sum = num1 + num2
    return sum

result = add(2, 3)
print(result)
```

```
5
```

Another way to pass variable to main program – using return keyword

Function benefit?

- Re-use code: create a common routine once and re-use as many times as you want
- Clean code: Make your code structured
- Share code with other programs
- Independent: Test function separate from the rest of the code
- Reduces need for global variables as local variables used instead

Difference

```
name = ("John", 15)
name = hello()
name = hello("John", 15)
name = hello(("John", 15))
```

Tuple (datatype)

Function, with no argument

Function, with 2 argument (string & int)

Function, with 1 argument (tuple)

enactus
University of Sheffield

CODE CREATORS
SHEFFIELD

Recursion

A function that called itself (again and again)

```
def run():
    run()
```

```
def run(numbers):
    run(numbers)
```

Recursion

```
count = 3
def run(numbers):
    if(count == 0):
        return
    else:
        return run(numbers - 1)
```

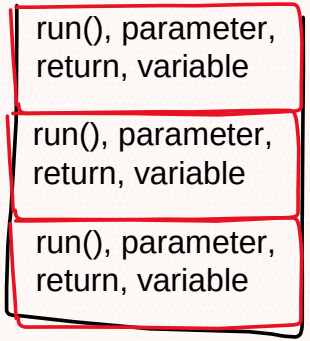
Infinite loop

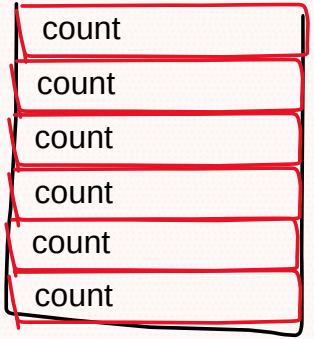
```
count = 3
while True:
    if(count == 0):
        break
    else:
        count -= 1
```

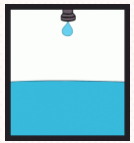
enactus University of Sheffield

CODE CREATORS SHEFFIELD

Recursion VS infinite loop







```
count = 3
def run(numbers):
    if(count == 0):
        return
    else:
        return run(numbers - 1)
```

```
count = 3
while True:
    if(count == 0):
        break
    else:
        count -= 1
```

enactus University of Sheffield

CODE CREATORS SHEFFIELD

Scope – naming issues

A primary school teacher requires a program that will allow pupils to practise their multiplication tables (times tables). The program must allow them to choose the table they want displayed and the start and end numbers to multiply by.

The program will then print a message followed by the table selected.
Create a solution for this using a subroutine called **multiples()** which takes **table**, **startnum**, **endnum** and **pupilName** as parameters. The subroutine will output the message and multiplication table. The main program will prompt the user to enter the values and will then pass them to the routine.

```
What is your name:  Joe
Which multiply table? (2-9): 5
Start from(1-15): 4
End at (large than start): 12

Joe, this is your personalised multiplication
table
5 x 4 = 20
5 x 5 = 25
…
…
5 x 12 = 60
```