Carlos J. Costa

PYTHON PROGRAMMING FUNDAMENTALS

Index

- Learning Objectives
- Conditions and Branching
- Loops
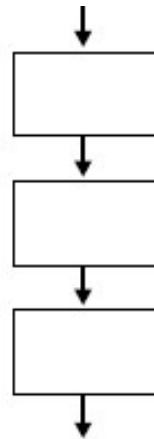- Functions

Learning Objectives

- Know key control structures
- Know how to do small programs
- Know what a function is
- Know how to create simple functions

Control Structure

- Sequencing
- Decision
- Loops

Sequencing

- By default, in a structured language code is interpreted from right to left and top to bottom.

Decision
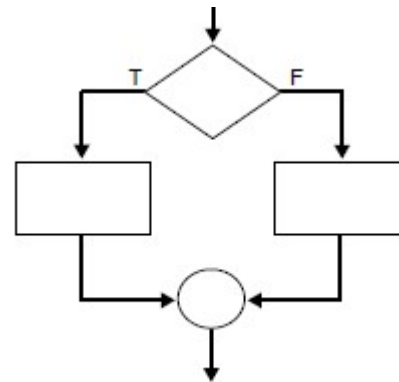
```
IF-THEN Statement

# A simple decision
age=?
if (age >= 18):
    print("You can vote!")
```

Decision

- IF-THEN statements test for only one action.
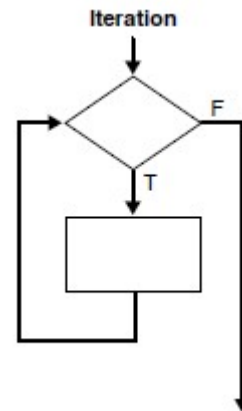- If the condition is true, then an action occurs. If the condition is false, take an alternate action.

Decision

```
#If-elif-else structure
if age < 3:
    ticketPrice= 0
elif age < 18:
    ticketPrice = 5
else: ticketPrice = 10
```

Loops

- A WHILE loop is a process in which a loop is initiated until a condition has been met.
- This structure is useful when performing iterative instructions to satisfy a certain parameter.

Loops

- **The loop ends as a result of value entered by user**

```
msg = ''
while msg != 'exit':
    msg = input("Write a message: ")
    print(msg)
```

Functions

- Function is a named blocks of code that is used to accomplish a specific purpose.
- Information passed to a function is an argument
- The information received by the function is a parameter

Function

- **Simplified Function**

```
def greet ():
  # greeting
  print ("Hello!")
```

- **To call the function, it's called by name:**

```
greet ()
```

Function

- For the function to have greater use there is need to pass arguments

```
#argument pass
def greetUser (name):
   "" "Show a custom message." ""
   print ("Hello," + name + "!")
```

- Call Example

```
greetUser ("John")
```

Function

- We can have default parameter

```
#argument pass
def greetUser (name = 'Zézinho'):
  "" "Show a custom message." ""
  print ("Hello," + name + "!")
```

- welcome_user ()

```
greetUser ("John")
```

Function

- ## Return a value

```
def addValues (x, y):
"""
    takes two values and returns the sum.
"""
  return x + y
```

- ## sum = addValues (3,5)

```
print (sum)
```

Function

- ## Sum with default values

```
def addValues (x = 0, y = 0, z = 0):
    """

    add multiple values and return the sum.
    """

    return x + y + z
```

- ## Call function

```
sum = addValues (3,4)
print (sum)
```