Python Programming for Novice

Malvika Sharan & Olav Vahtras

Workshop: Software Writing Skills for Young Researchers
2015.09.23 - 2015.09.25

GFZ Helmholtz-Zentrum Potsdam, Germany

Python Programming for Novice

GFZ
Helmholtz Centre
POTSDAM

software carpentry

FOSTER
FACILITATE OPEN SCIENCE TRAINING FOR EUROPEAN RESEARCH

Malvika Sharan & Olav Vahtras

Day – 2: Session – 1

2015-09-24

Recap

- Yesterday we covered the following topics:
  - Print "Hello World!", 'literal constants like "strings", "integers", "floats" etc.'
  - Variables: Example

    ```
    >>> book = 'Game of Thrones'
    >>> author = 'R. R. Martin'
    ```

  - Printing several strings by using string format operator

    ```
    >>> print "author of the book %s is %s" % (book,  author)
    ```

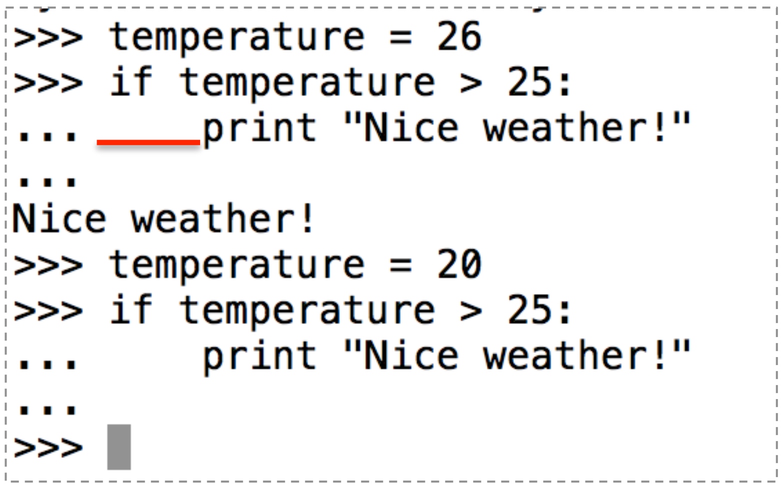
  - Using math operators: float((2 + 3)-11*(6/7))
  - Data structures: creating, accessing, manipulating by adding and removing
    - MyList = [], MyList = [1, 2, 'C'], MyList("string"), MyList[-1] etc.
    - MyDict = {}, MyDict = {"Location" : "Winterfell"}, MyDict["Head"] = "Eddard Stark", MyDict.pop("Head"), MyDict.items() etc.

Control Flow

- So far we commanded Python to do stuffs (print, add etc.)
- Control-flow allows Python to take a decision and do different things depending on different situations
- In Python are 3 control flow statements: If, for and while

- The if statement is used to check a condition
  - Looks for answer in true or false
  - if the condition is true, block of codes inside the if statement will be executed

4 spaces or a tab



```
>>> temperature = 26
>>> if temperature > 25:
...     print "Nice weather!"
...
Nice weather!
>>> temperature = 20
>>> if temperature > 25:
...     print "Nice weather!"
...
>>> 
```

Checks if the value of temperature (>25)
Since its true, it executes the block of code

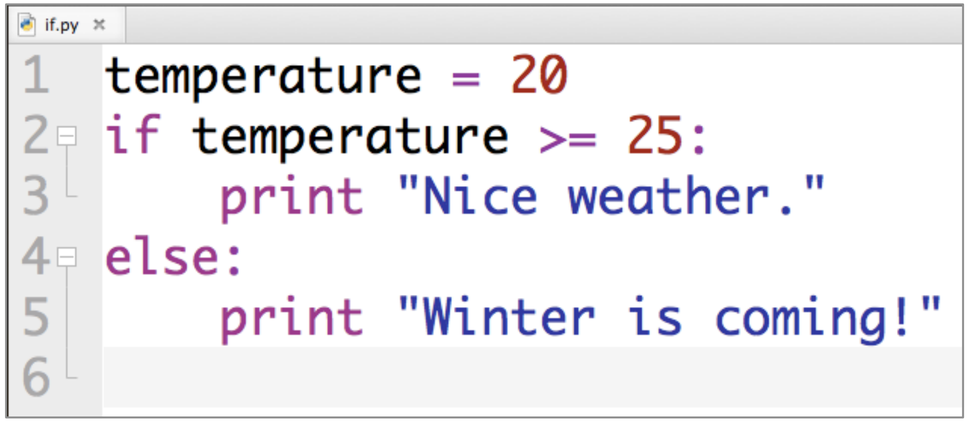
Since its false, it does not read the block of code that belongs to our if-statement

Editors

- We will continue with the if-statement but let’s first address the problem of writing codes on terminal
  - No one wants to type the same code again and again
- We can solve this by writing our codes/snippets using an editor and saving them for future use
  - There are several powerful editors like: Vim, Emacs, Komodo etc.
  - And some easy to use editors: gedit, nano, PyCharm and Notepad++
- The codes are saved by using name of file followed by ‘.py’
- Please create a python script called if.py and repeat the last task
- Using ‘python if.py’, we can execute the python codes

If Statement

- When if statement is false, Python executes the anther block of codes in the else statement

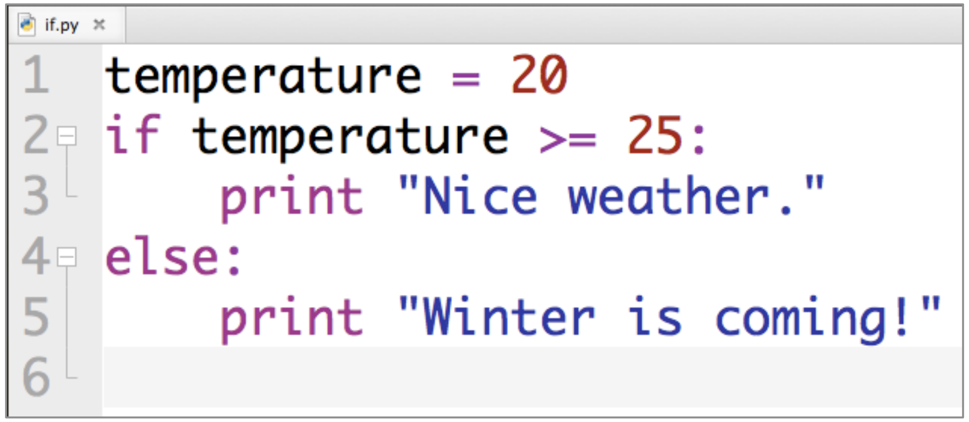


if.py

```
temperature = 20
if temperature >= 25:
    print "Nice weather."
else:
    print "Winter is coming!"
```

If Statement

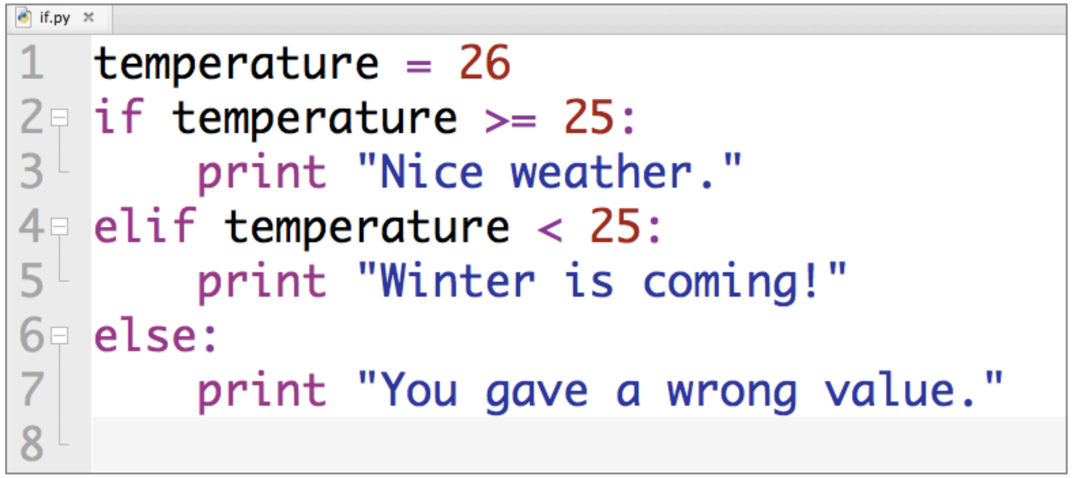
- When if statement is false, Python executes the anther block of codes in the else statement



if.py

```
temperature = 20
if temperature >= 25:
    print "Nice weather."
else:
    print "Winter is coming!"
```

- Multiple conditions can be given by introducing elif statement

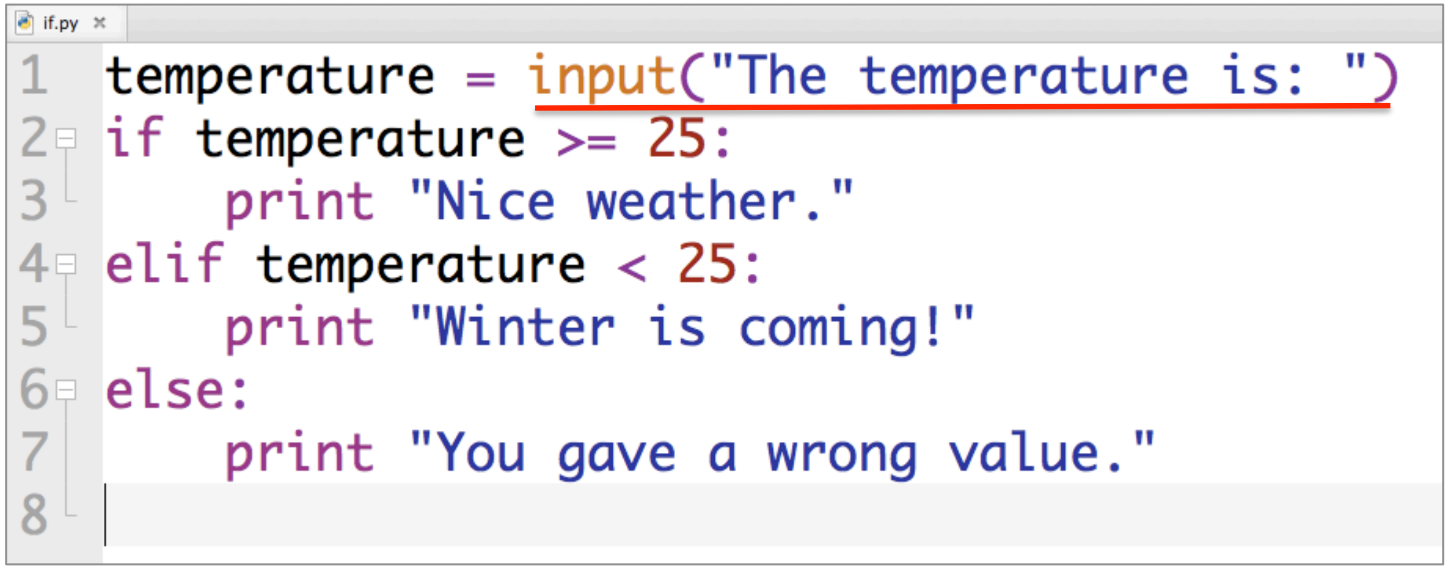


if.py

```
temperature = 26
if temperature >= 25:
    print "Nice weather."
elif temperature < 25:
    print "Winter is coming!"
else:
    print "You gave a wrong value."
```

If Statement Exercise - 1

- Take input from terminal by using input() or raw_input()

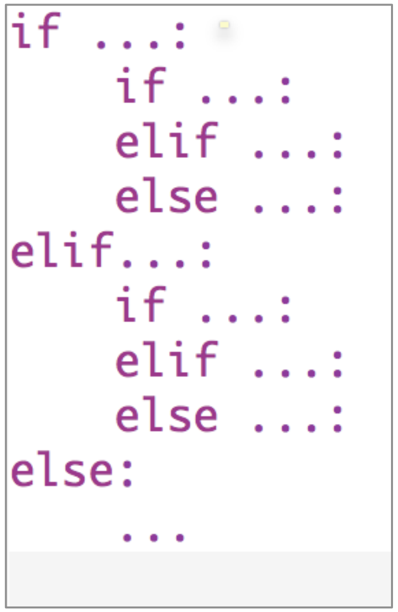


```python
temperature = input("The temperature is: ")
if temperature >= 25:
    print "Nice weather."
elif temperature < 25:
    print "Winter is coming!"
else:
    print "You gave a wrong value."

```

- Great! but temperature above 40 is not nice
  - How to test multiple conditions before executing code?

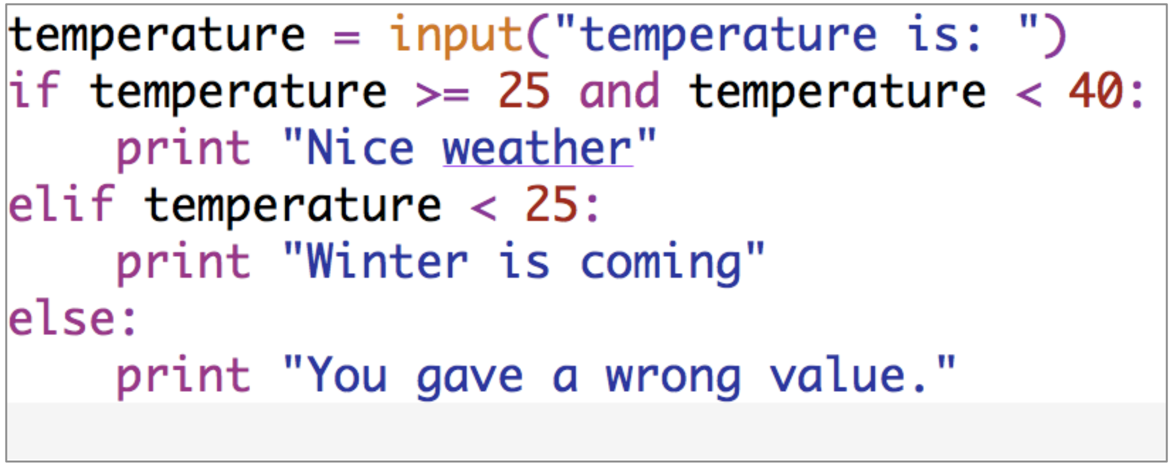
If Statement

1. By not so elegant nested if statements



```
if ...:
    if ...:
    elif ...:
    else ...:
elif...:
    if ...:
    elif ...:
    else ...:
else:
    ...
```

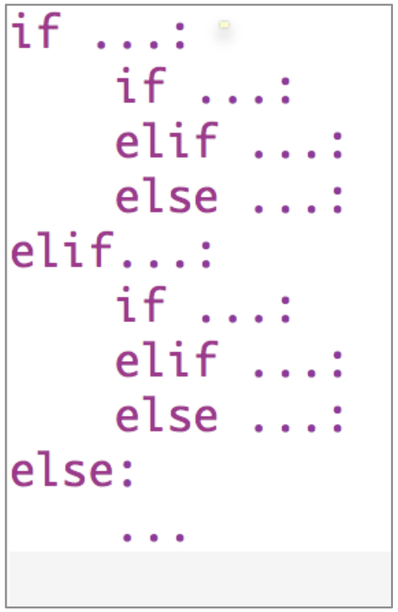
2. Connecting conditions by Boolean (and, or, not) and extend your code



```
temperature = input("temperature is: ")
if temperature >= 25 and temperature < 40:
    print "Nice weather"
elif temperature < 25:
    print "Winter is coming"
else:
    print "You gave a wrong value."
```

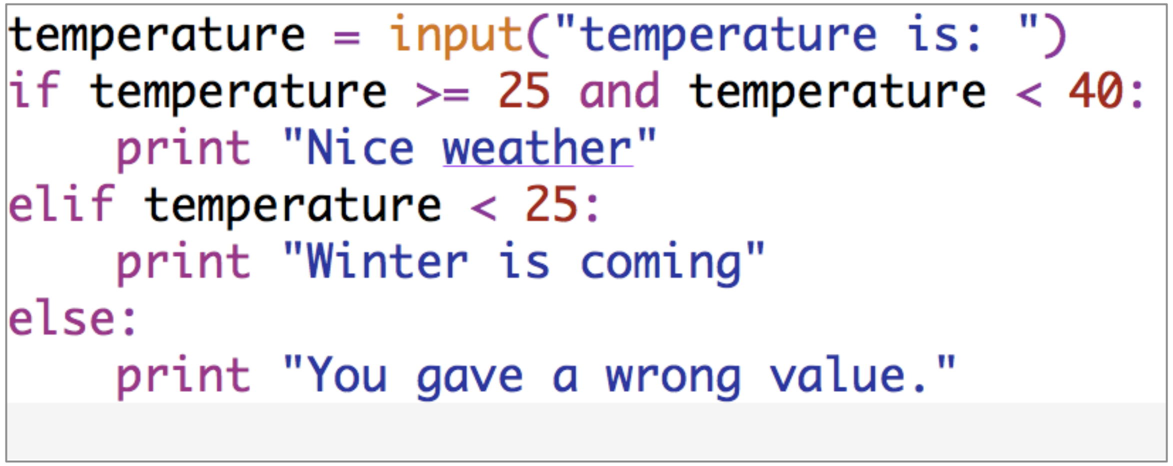
If Statement

1. By not so elegant nested if statements



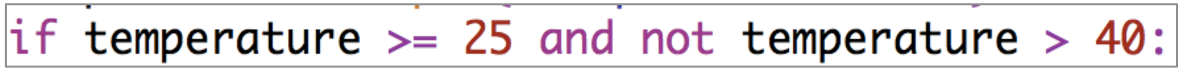
```
if ...:
    if ...:
    elif ...:
    else ...:
elif...:
    if ...:
    elif ...:
    else ...:
else:
    ...
```

2. Connecting conditions by Boolean (and, or, not) and extend your code

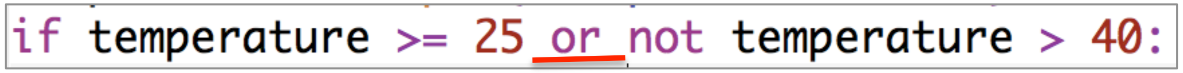


```
temperature = input("temperature is: ")
if temperature >= 25 and temperature < 40:
    print "Nice weather"
elif temperature < 25:
    print "Winter is coming"
else:
    print "You gave a wrong value."
```

Check following statements (line-2):



```
if temperature >= 25 and not temperature > 40:
```

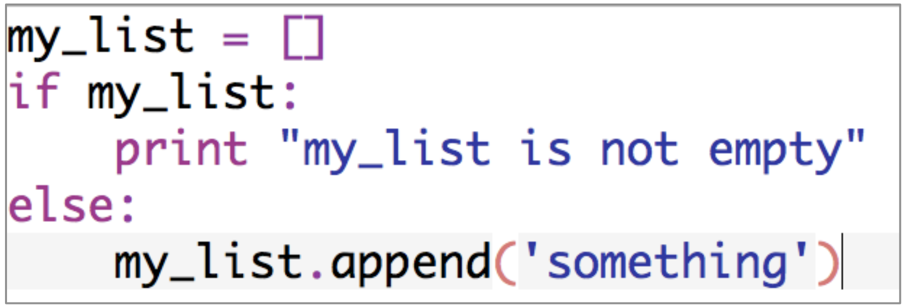


```
if temperature >= 25 or not temperature > 40:
```

Warning: English’s ‘or’ and Python’s ‘or’ are not always the same

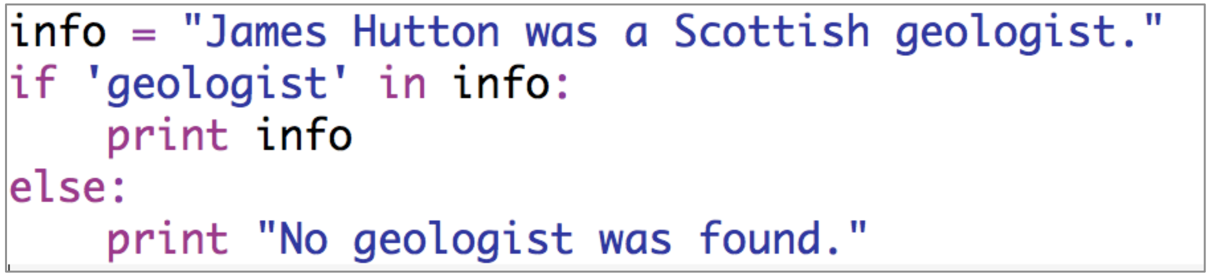
If Statement

- Some more useful use of if statements:
  1. Check if a variable or data type exists (not empty)



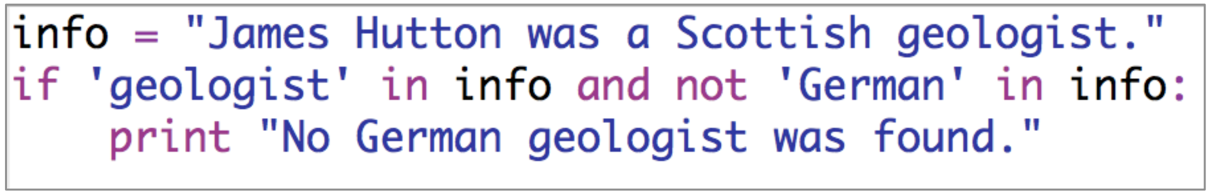
```
my_list = []
if my_list:
    print "my_list is not empty"
else:
    my_list.append('something')
```

  2. Checks if an item exists in a string or data structure (if ‘x’ in list:)



```
info = "James Hutton was a Scottish geologist."
if 'geologist' in info:
    print info
else:
    print "No geologist was found."
```

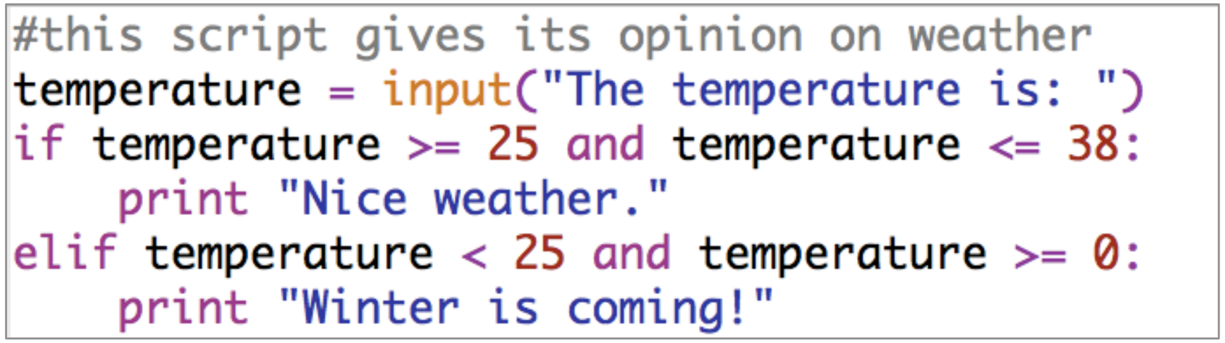
  3. Or does not exist



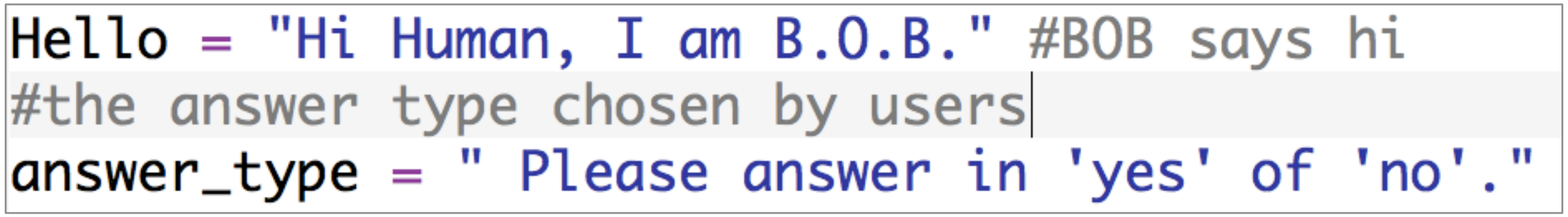
```
info = "James Hutton was a Scottish geologist."
if 'geologist' in info and not 'German' in info:
    print "No German geologist was found."
```

Commenting and Annotating Codes

- “#” for writing comments or commenting out you codes
  - Annotate your codes so other’s can learn what your code is doing



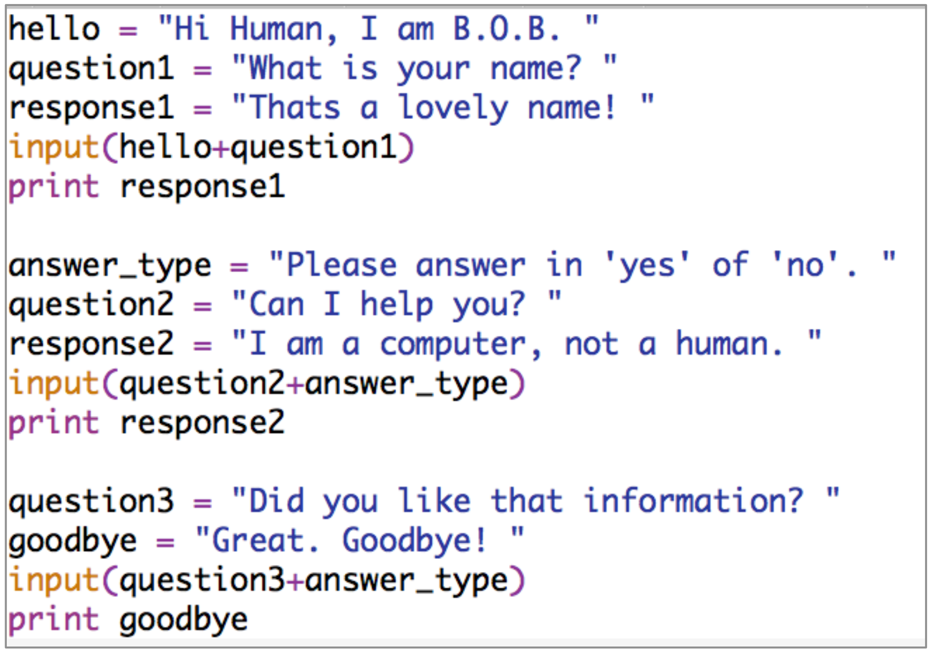
```
#this script gives its opinion on weather
temperature = input("The temperature is: ")
if temperature >= 25 and temperature <= 38:
    print "Nice weather."
elif temperature < 25 and temperature >= 0:
    print "Winter is coming!"
```



```
Hello = "Hi Human, I am B.O.B." #BOB says hi
#the answer type chosen by users
answer_type = " Please answer in 'yes' of 'no'."
```

If Statement Exercise - 2

Problem 1 (bob.py)

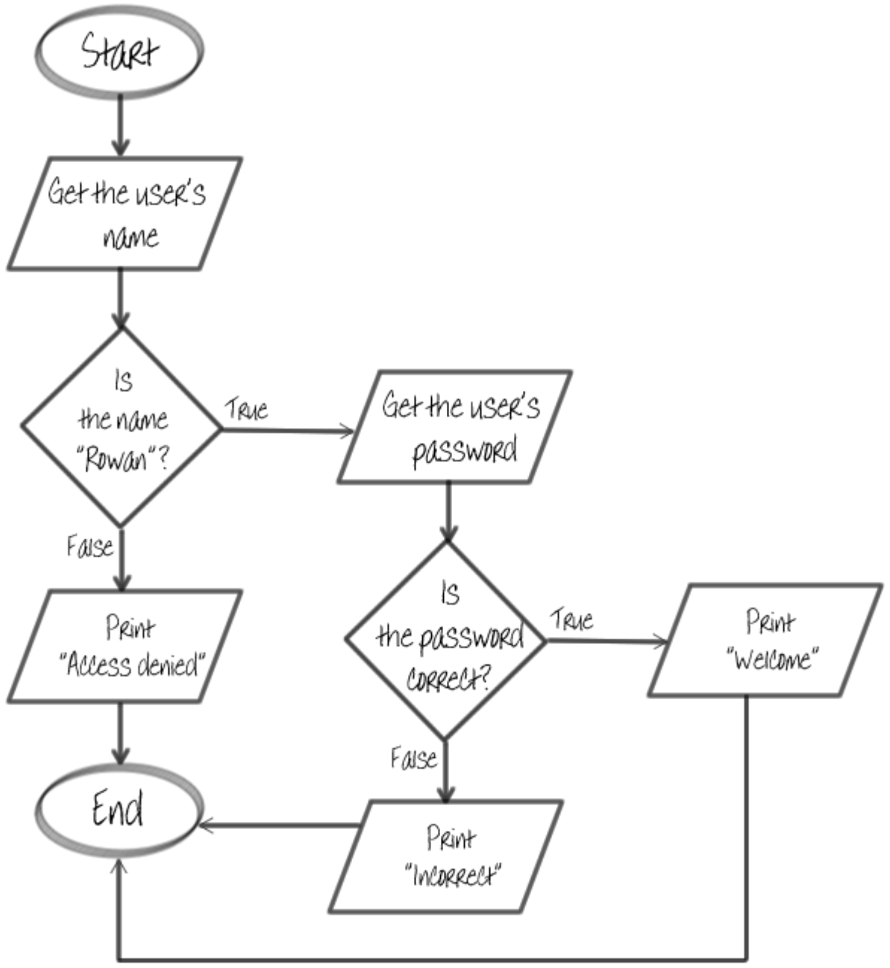


```
hello = "Hi Human, I am B.O.B. "
question1 = "What is your name? "
response1 = "Thats a lovely name! "
input(hello+question1)
print response1

answer_type = "Please answer in 'yes' of 'no'. "
question2 = "Can I help you? "
response2 = "I am a computer, not a human. "
input(question2+answer_type)
print response2

question3 = "Did you like that information? "
goodbye = "Great. Goodbye! "
input(question3+answer_type)
print goodbye
```

Make B.O.B (Basic Output Being) smarter by letting it differentiate "yes" and "no" and respond to the user accordingly.

For example: if the answer of question2 is in yes, then let B.O.B. help user somehow and if the answer is no say goodbye already!

Problem 2 (user.py)

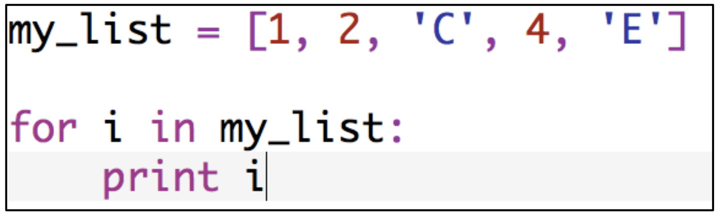


http://usingpython.com/python-if-else-statements/

For-loop

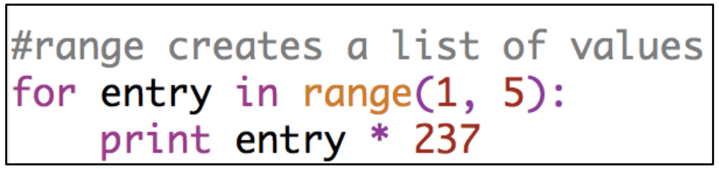
- Most often we carry out the same task repeatedly
  - Reading each items in list or dict
  - Reading several files of same file format
  - Extracting information
  - Correcting a misprinted word in several files

- The For loop of for … in statement is the most powerful way to tackle the repeated tasks

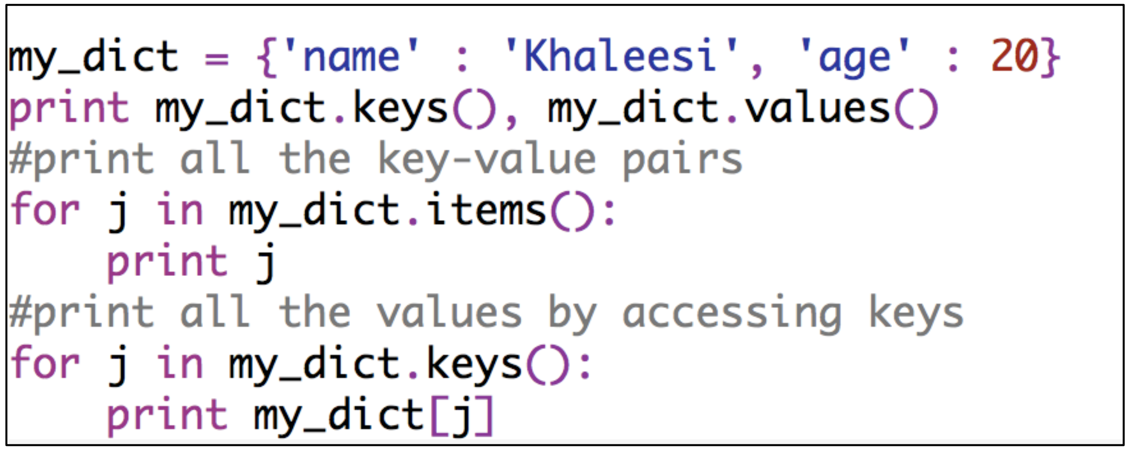


```
my_list = [1, 2, 'C', 4, 'E']

for i in my_list:
    print i
```

Use range:



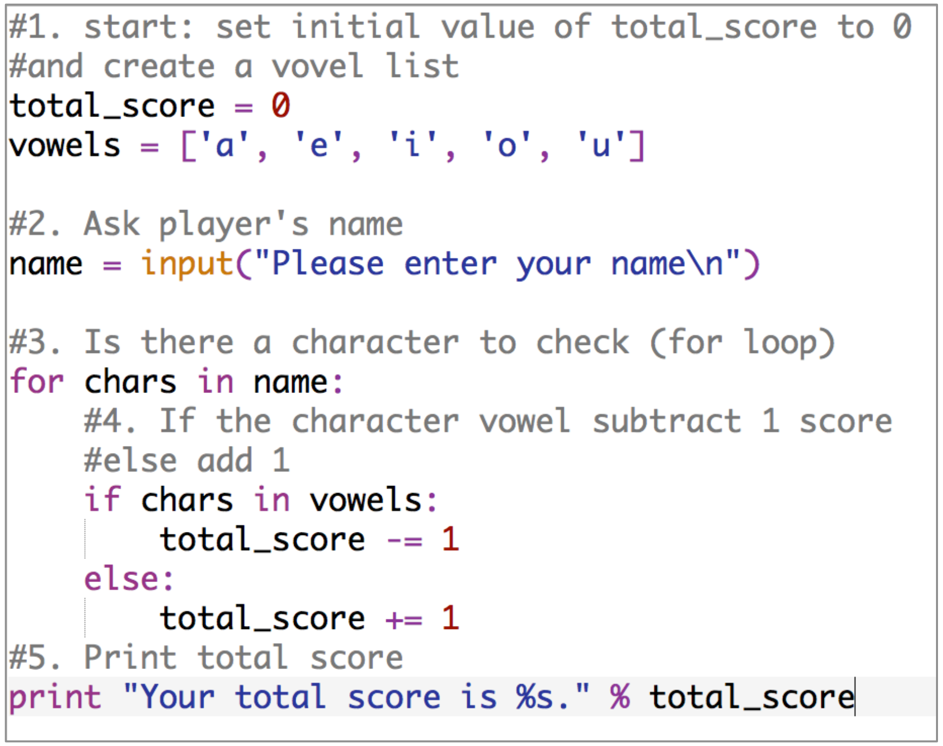
```
#range creates a list of values
for entry in range(1, 5):
    print entry * 237
```



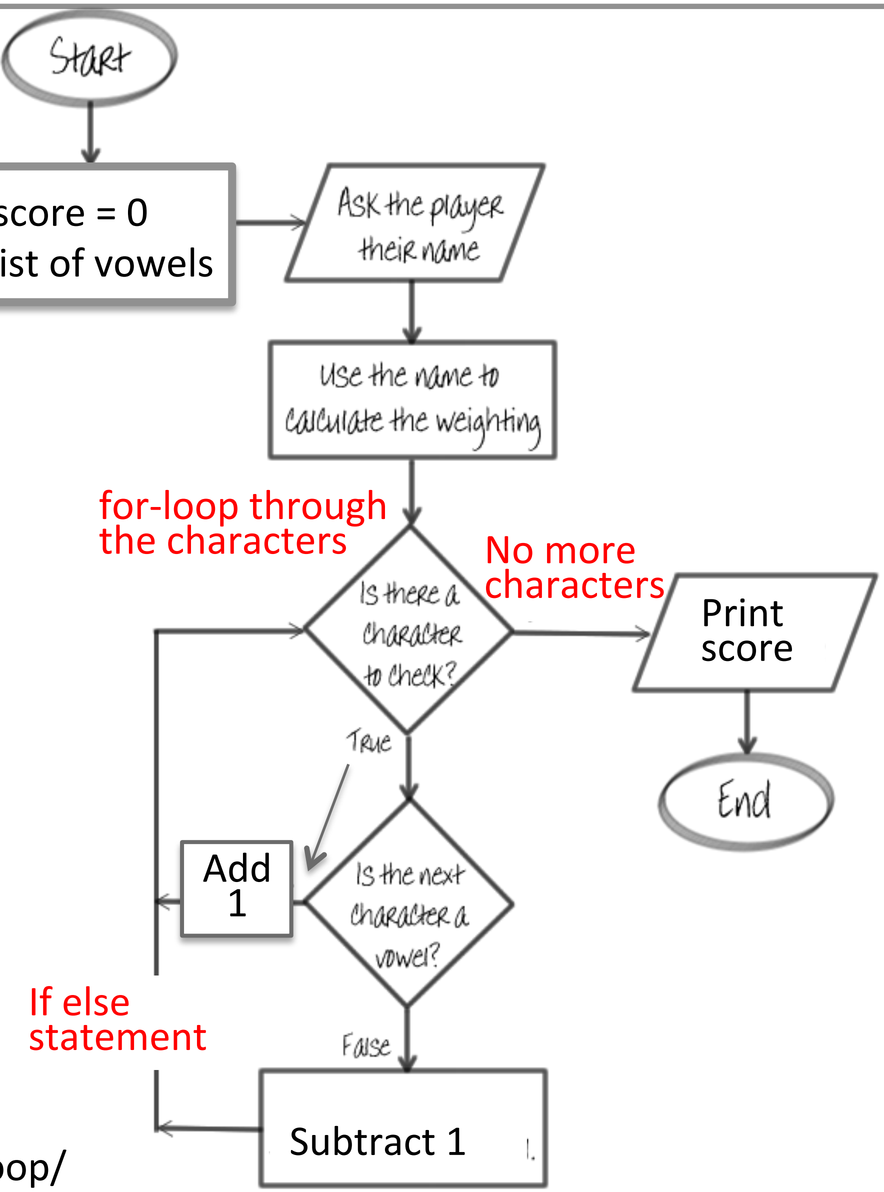
```
my_dict = {'name' : 'Khaleesi', 'age' : 20}
print my_dict.keys(), my_dict.values()
#print all the key-value pairs
for j in my_dict.items():
    print j
#print all the values by accessing keys
for j in my_dict.keys():
    print my_dict[j]
```

For-loop Example

Script: name_score.py



```
#1. start: set initial value of total_score to 0
#and create a vovel list
total_score = 0
vowels = ['a', 'e', 'i', 'o', 'u']

#2. Ask player's name
name = input("Please enter your name\n")

#3. Is there a character to check (for loop)
for chars in name:
    #4. If the character vowel subtract 1 score
    #else add 1
    if chars in vowels:
        total_score -= 1
    else:
        total_score += 1
#5. Print total score
print "Your total score is %s." % total_score
```



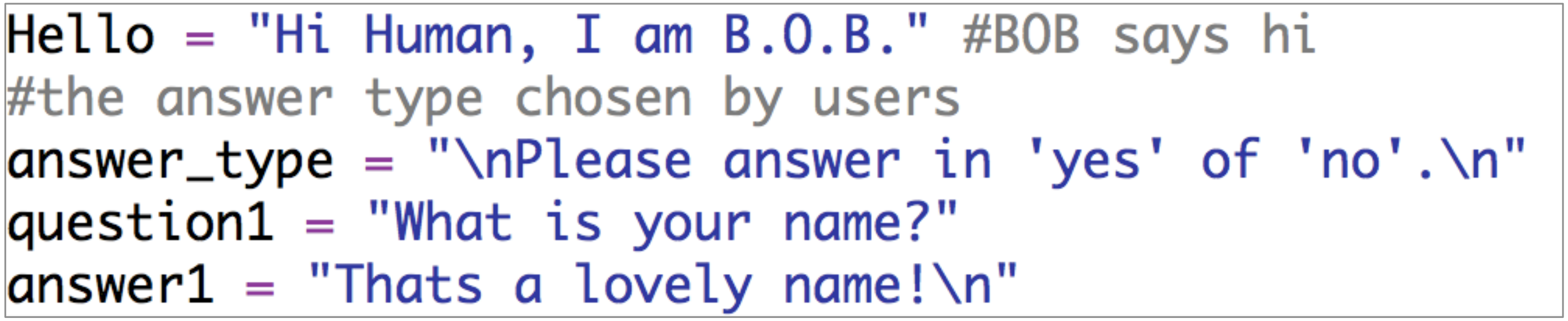
Start
i. total_score = 0
ii. Create a list of vowels
Ask the player their name
Use the name to calculate the weighting
for-loop through the characters
Is there a character to check?
No more characters
Print score
End
True
Is the next character a vowel?
Add 1
False
Subtract 1
If else statement

Adapted from: http://usingpython.com/python-for-loop/

Printing Newline and Tab

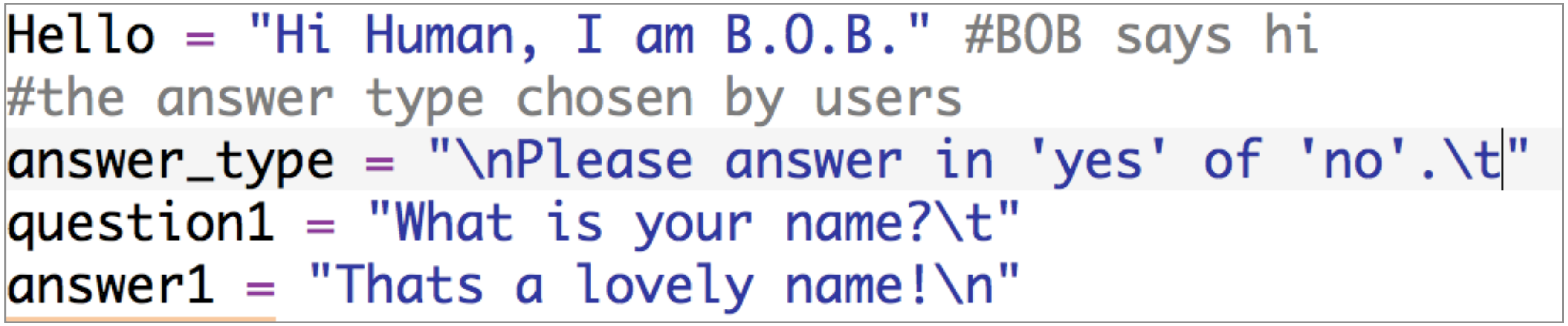
Most commonly used in creating files and tables

- “\n” for newline



```
Hello = "Hi Human, I am B.O.B." #BOB says hi
#the answer type chosen by users
answer_type = "\nPlease answer in 'yes' of 'no'.\n"
question1 = "What is your name?"
answer1 = "Thats a lovely name!\n"
```

- “\t” for tab ()



```
Hello = "Hi Human, I am B.O.B." #BOB says hi
#the answer type chosen by users
answer_type = "\nPlease answer in 'yes' of 'no'.\t"
question1 = "What is your name?\t"
answer1 = "Thats a lovely name!\n"
```

File Handling

- File handling refers to creating, opening, reading and writing files
- Please see the script “file_handle.py”
- fh = open(): open a file and assign it to a variable fh
- open(filename, ‘a’): create an empty file
- fh = open(filename, ‘r’): read an existing file
- fh = open(filename, ‘w’): write in a new file
- fh.close(): close the open file
- Learn how to use if-statements and for-loops in file handling