1 if, elif, else, while, and logical operators

Like many programming languages, Python enables code to be run based on testing whether a condition is true. Python uses indented blocks to run code according to these tests.

1.1 if, else, elif

emphif statements run a block of code if a particular condition is true. emphelif or 'else if' statements can subsequently run to test more conditions, but these run only if none of the precious emphif or emphelif statements were true. emphelse statements may be used to run code if none of the previous emphif or emphelif were true.

emphif statements may be run alone, with emphelse statements or with emphelif statements. emphelse and emphelif statements require a previous emphif statement.

In the following example we use the input command to ask a user for a password and test against known value.

Here we use just one elif statement, but multiple statements could be used.

Note that the tests of equality uses double = signs

```
password = input ("Password? ") # get user to enter password
if password == "secret":
    print ("Well done")
    print ("You")
elif password=="Secret":
    print ("Close!")
else:
    print("Noodle brain")

OUT:
Password? Secret
Close!
```

1.2 while statements

emphwhile statements may be used to repeat some actions until a condition is no longer true. For example:

```
x = 0
while x <5:
    x +=1 # This is shorthand for x = x + 1
    print (x)

OUT:
1
2
3
4
5
```

1.3 Logical operators

Logical operators

The following are commonly used logical operators:

== Test of identity

!= Not equal to

>greater than

<less than

>= equal to or greater than

<= less than or equal to

in test whether element is in list/tuple/dictionary

not in test whether element is not in list/tuple/dictionary

and test multiple conditions are true

or test one of alternative conditions are true

any test whether all elements are true

all test whether all elements are true

When Python test conditions they are evaluated as either True or False. These values may also be used directly in tests:

```
x = 10
y = 20
z = 30

# using and/or
print (x>15)
print (x>15 and y>15 and z>15)
print (x>15 or y>15 or z>15)
print ()

# Using any and all
print (any([x>20, y>20, z>20]))
test_array = [x>20, y>20, z>20]
print (test_array)
print (any(test_array))
print (all(test_array))

OUT:
False
False
True

True
[False, False, True]
True
False
```