Chp03A Selection using if structure

References

1. https://www.w3schools.com/python/python_conditions.asp w3school Python Tutorial (if structure)

2. https://www.tutorialspoint.com/python3/python_decision_making.htm Decision making

3. https://docs.python.org/3/tutorial/controlflow.html *Read sections 4.1-4.4. Can also be obtained from Python manual*
4. *Learning Python, 5th (mark Lutz Oreilly 2013) – **Chapter 12 if Tests and Syntax Rules***
5. *Python for Everybody – **Chapter 3 Conditional execution***

Control structures

Control structures falls into two categories:

1. Selection structure
    - The **if** statement
    - The **if..else** statement
    - The **if..elif..elif..else** statement

2. Repetition/Loops structure
    - The **while** repetition structure
    - The **for** repetition structure

Relational (Comparison) Operators - I

Operator	Meaning	Description	Example
==	Is equal	If the values of two operands are equal, then the condition becomes true.	10==20 gives False
!=	Is not equal	If values of two operands are not equal, then condition becomes true.	10 != 20 gives True
>	Greater than	If the value of left operand is greater than the value of right operand, then condition becomes true.	10 > 20 gives False
<	Less than	If the value of left operand is less than the value of right operand, then condition becomes true.	10 < 20 Gives True

Relational (Comparison) Operators - II

Operator	**Meaning**	**Description**	**Example**
>=	Greater or Equal	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	10 >=20 gives False
<=	Less or Equal	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	10 <= 20 gives True

Relational operators examples

```
>>> type(True) #<class 'bool'>
>>> type(False) # <class 'bool'>
# Relational operators
>>> x=6; y=5; z=6;
>>> x==y  # False
>>> x>y   # True
>>> x<y   # False
>>> x>=y  # True
>>> y>= x # False
>>> x!=y  # True
```

Demo: chp03Aex00ARelationalOperators

Logical Operators

Are used to form compound statements

Operator	**Meaning**
and	If both the operands are true then condition becomes True.
or	If any of the two operands are non-zero then condition becomes True.
not	Used to reverse the logical state of its operand

Note: Do NOT use &&, ||, !, as in C language

Logical operators examples

Truth Table

A	B	A and B	A or B	Not A
F	F	F	F	T
F	T	F	T	T
T	F	F	T	F
T	T	T	T	F

```
>>> x=6
>>> x>5 and x<10  # True
>>> x< 5 and x<10  # False
>>> x%2==0         # True
>>> x%2==1         # False
```

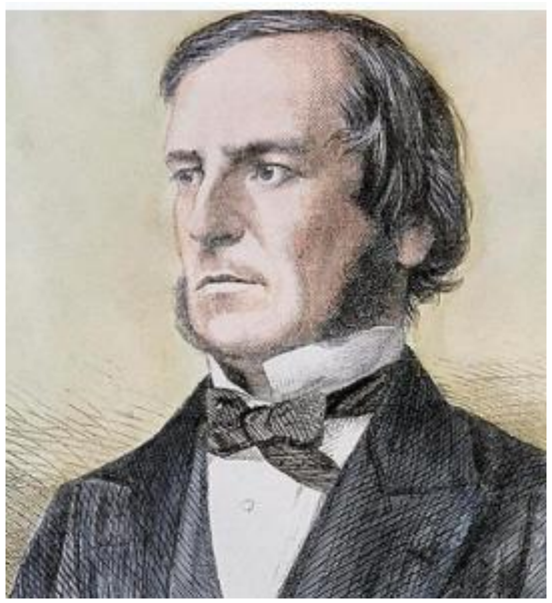
Demo: chp03Aex00BLogicalOperators

George Boole

.
https://en.wikipedia.org/wiki/George_Boole

Boolean expression

Born 2 November 1815
Died: 8 December 1864
English mathematician, philosopher and logic
Boolean algebra



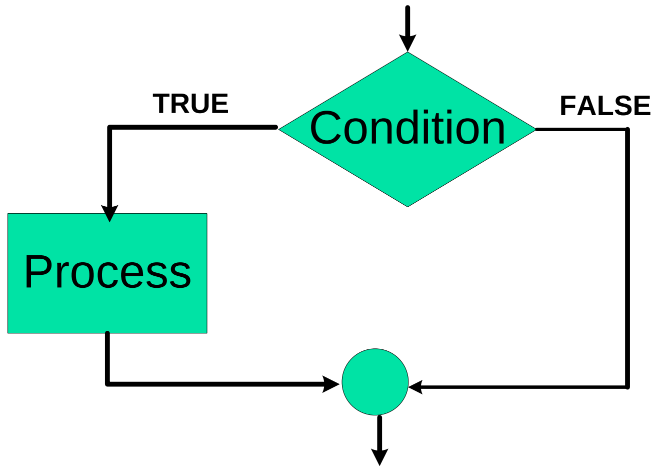
Simple if statement

Syntax:

```
if Boolen_expression:
    statement(s) to execute if BE==True
```

- Can be one or more statements (block) to execute if Boolean Expression (BE) is True.
- **There must be** a colon (:) after the Boolean expression.
- The body of the **if** must be indented (by default Python indentation is 4 spaces).
- Python relies on **indentation**, using whitespace, to define scope in the code. Other programming languages often use curly-brackets {} for this purpose.

- There is no need to put brackets to enclose the Boolean expression, this is a standard in python of **NOT** enclosing Boolean expression in parenthesis.
- If Boolean Expression evaluates to False, then the first set of code after the end of block is executed.
- Following diagram shows Flow diagram of if statement.



Example:

chp03Aex01SimpleIf01

```
marks=float(input("Enter marks: "))
if marks>=70:
    print("Grade is A")
    print("Remark is pass")
print("This is outside the if block")
```

Output if entered marks >=70

```
Grade is A
Remark is pass
This is outside the if block
```

Output if marks<70

```
This is outside the if block
```

If the suite of an **if** clause consists only of a single line, it may go on the same line as the header statement (BUT what about readability?)

```
chp03Aex02SimpleIf02
marks=float(input("Enter marks: "))
if marks>=70: print("Grade is A, Remark is pass")
print("This is outside the block")
```

Note: If statement, **without indentation** will raise an error as following code shows.

```
if marks>=70:
print("Grade is A, Remark is pass") #error
```

if..else structure

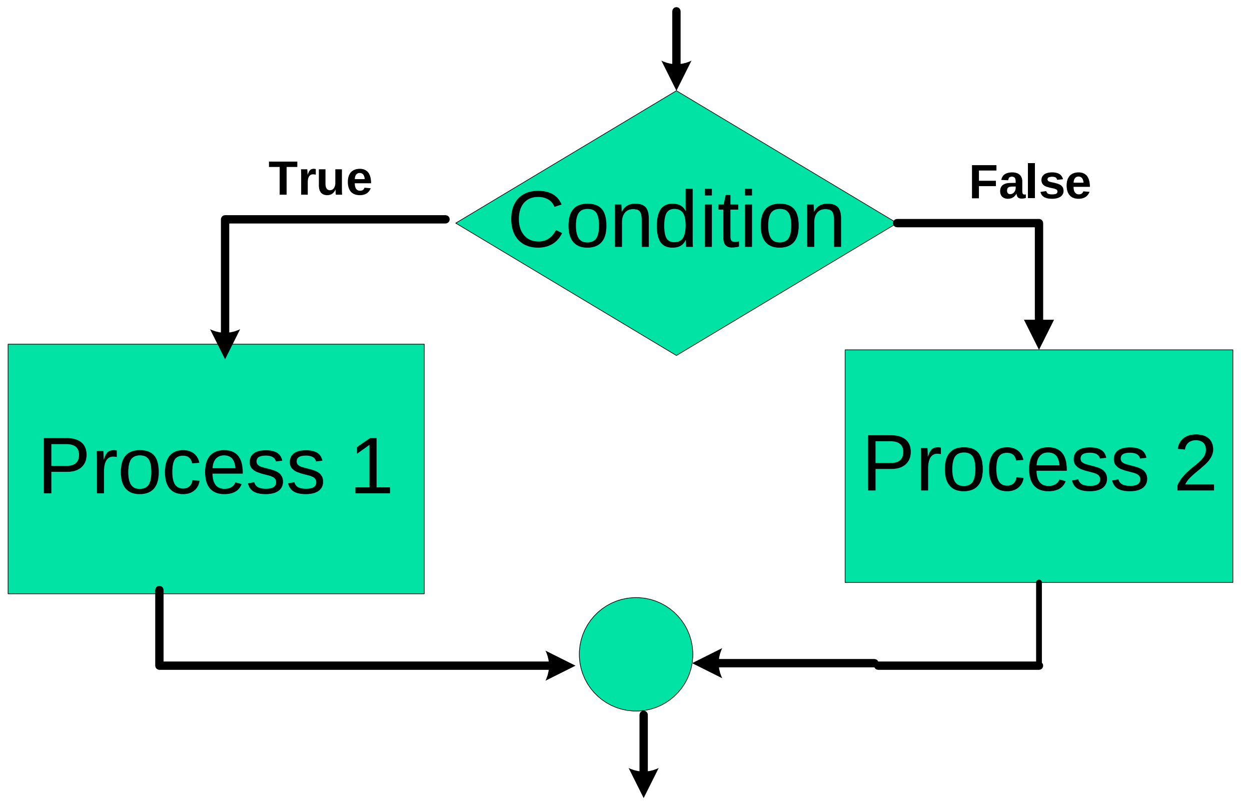
Sometimes we want to do one thing if a logical expression is True and something else if the expression is False. In such a case if..else is used.

Syntax:

```
 if Boolen_expression:
        #statement(s) if BE==True
else:
       # statement(s) if BE==False
```

The **else** statement is an optional statement and there could be at the most only one **else** statement following **if**.

Flow diagram of if..else



chp03Aex03IfElse01

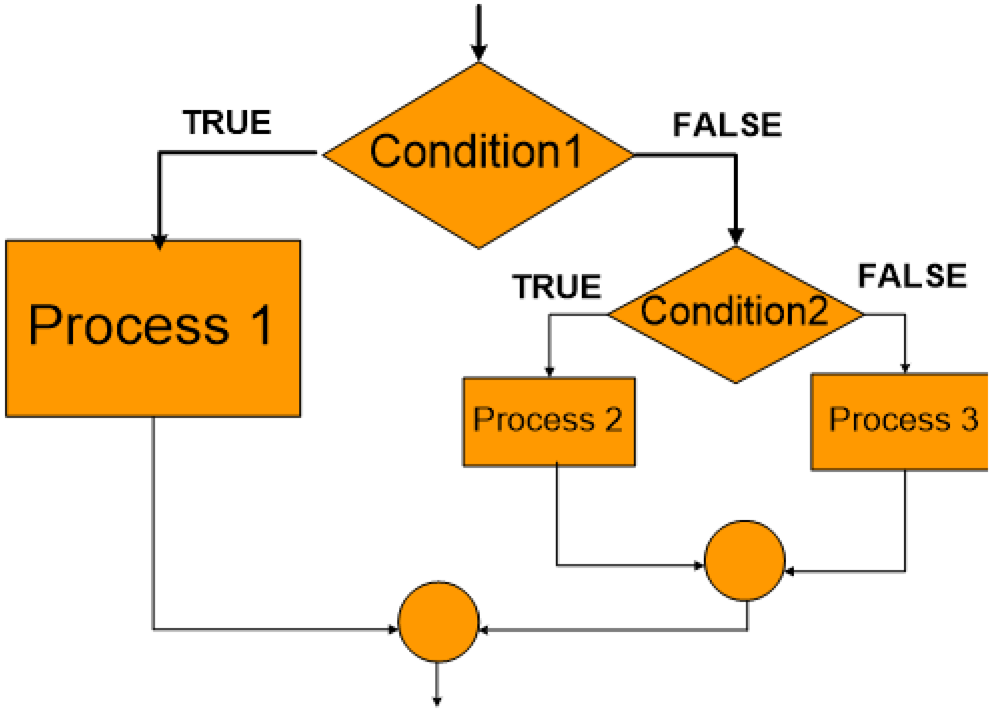
```python
marks = float(input("Enter marks: "))
if marks>=50:
    print("Grade is B or above")
    print("Remark is pass")
else:
    print("Grade is below B")
    print("Remark is Fail")

print("This is outside the block")
```

(a) Run the program for marks >=50
(b) Run the program for marks <50

if..elif..else structure - I

Syntax:

```
if expression1:
   statement(s)
elif expression2:
   statement(s)
elif expression3:
   statement(s)
………
else:
   statement(s)
```



if..elif..else structure - II

chp03Aex04ifelifelse01

```
marks=float(input("Enter marks 0-100 : "))
if marks>100: grade= "invalid"
elif marks>=70: grade="A"
elif marks>=60: grade="B+"
elif marks>=50: grade="B"
elif marks>=40: grade="C"
elif marks>=35: grade="D"
elif marks>=0: grade="E"
elif marks<0: grade= "invalid"
print(f"marks is {marks:.2f} and Grade is 
{grade}")
```

H/W: Test the program for all ranges of grades

if..elif..else structure - III

Can also use logical operators (**and**, **or**, **not**) with if structure

chp03Aex05ifelifelse02

```
marks=float(input("Enter marks 0-100 : "))
if marks<0 or marks>100:
    grade="invalid"
elif marks>=70: grade="A"
……
```

H/W: Test the program with valid and invalid marks

if..elif..else structure - IV

```
chp03Aex06ifelifelse03 #uses "and" logical operator
marks=float(input("Enter marks 0-100 : "))
if marks>=70 and marks<=100:
   grade="A"
……………
```

H/W: Test the program for all grades including invalid marks (marks<0 or marks>100)

Nested decision - I

Syntax:

```
if expression1:
    statement(s)
    if expression2:
        statement(s)
    elif expression3:
        statement(s)
    else:
        statement(s)
elif expression4:
    statement(s)
else:
    statement(s)
```

Nested decision - II

Example: chp03Aex07nestedIf01

```
x = int(input("Enter value of x: "))
if x > 1 :
    if x < 100 :
        print("x is greater than 1 and 
Less than 100")
    else: print("x is greater than 1 and 
greater than 100")
else: print("x is less or equal to 1")
print('All done')
```

H/W: Test for x<1 (b) x> 1 but x<100 (c) x>100

Using and logical operator with if

Alternative of using 'and' operator

Consider following expression:

```
marks=float(input("Enter marks 0-100 : "))
if marks>=70 and marks <=100:
…….
```

In python, above statements can be written as:

```
marks=float(input("Enter marks 0-100 : "))
if 70<=marks<=100:
…….
```

Can use: `if not(70<=marks<=100): # outside`

Conditional Expression

Consider one statement in the if and one statement in the else as follows:

chp03Aex08ifelseonestatement

```
marks=float(input("Enter marks: "))
if marks>=50: print("Grade is B or above")
else: print("Grade is below B")
# alternative I:
print("Grade is B or above" if marks>=50 else
"Grade is below B")
```

Note: In other languages is called **Ternary** operator

Switch structure in Python

Python **does not** provide switch or case statements as in other languages such as C, javam but `if..elif...else` statements can be used to simulate switch case

End of chapter 3A

Next: Chapter 3B: Loops