if_else

September 15, 2025

1 Conditionals

2 Control Flow in Python

By default, statements in Python script are executed sequentially from **top to bottom**.
If the processing logic requires so, the sequential flow of execution can be altered in two ways:

- **Conditional execution**: a block of one or more statements will be executed if a certain expression is true.
- **Repetitive execution**: a block of one or more statements will be repetitively executed as long as a certain expression is true.

In this section, we will cover `if`, `else`, and `elif` statements.
The comparison and logical operators we learned in previous sections will be useful here.

2.1 If Condition

In Python and other programming languages, the keyword **`if`** is used to check if a condition is true and to execute the block of code.

Remember: Indentation after the colon (`:`) is required in Python.

3 syntax

if condition: this part of code runs for truthy conditions

4 Example: 1

[25]:
```
a = 3
if a > 0:
    print('A is a positive number')
# A is a positive number
```

```
A is a positive number
```

4.1 If Else

As you can see in the example above, `3` is greater than `0`.
The condition was true and the block code was executed.

However, if the condition is **false**, we do not see the result.
In order to see the result of the falsy condition, we should have another block, which is `else`.

4.1.1 If Else Statement

If the condition is true, the **first block** will be executed.
If not, the **else block** will run.

4.1.2 Syntax

"'python if condition: # this part of code runs for truthy conditions else: # this part of code runs for false conditions

[34]:
```
a = 3
if a < 0:
    print('A is a negative number')
else:
    print('A is a positive number')
```

```
A is a positive number
```

4.2 If Elif Else

The condition above proved false, therefore the `else` block was executed.
But what if our condition is **more than two**?
In that case, we could use `elif`.

4.2.1 Why Elif?

In our daily life, we make decisions on multiple conditions, not just one.
Similarly, in programming, we often need to check for **multiple conditions**.

We use `elif` when there are **more than two possible choices**.

4.2.2 Syntax

"'python if condition: # code block 1 (runs if first condition is true) elif condition: # code block 2 (runs if elif condition is true) else: # code block 3 (runs if none of the above are true)

[37]:
```
a = 0
if a > 0:
    print('A is a positive number')
elif a < 0:
    print('A is a negative number')
else:
    print('A is zero')
```

```
A is zero
```

4.3 Short Hand

4.3.1 syntax

“‘python code if condition else code

4.4 Example

[50]:
```
a = 3
print('A is positive') if a > 0 else print('A is negative') # first condition␣
↪met, 'A is positive' will be printed
```

```
A is positive
```

4.5 Nested Conditions

4.5.1 Conditions can be nested

“‘python # syntax if condition: code if condition: code

[55]:
```
a = 0
if a > 0:
    if a % 2 == 0:
        print('A is a positive and even integer')
    else:
        print('A is a positive number')
elif a == 0:
    print('A is zero')
else:
    print('A is a negative number')
```

```
A is zero
```

4.5.2 We can avoid writing nested condition by using logical operator and.

4.6 If Condition and Logical Operators

“‘python # syntax if condition and condition: code

[59]:
```
a = 0
if a > 0 and a % 2 == 0:
```

```
        print('A is an even and positive integer')
elif a > 0 and a % 2 !=  0:
     print('A is a positive integer')
elif a == 0:
    print('A is zero')
else:
    print('A is negative')
```

```
A is zero
```

4.6.1 If and Or Logical Operators

“‘python # syntax if condition or condition: code

[71]:
```
user = 'James'
access_level = 3
if user == 'admin' or access_level >= 4:
        print('Access granted!')
else:
    print('Access denied!')
```

```
Access denied!
```

1. Get user input using input(“Enter your age:”).: If user is 18 or older, give feedback: You are old enough to drive. If below 18 give feedback to wait for the missing amount of years. Output “‘python Enter your age: 30 You are old enough to learn to drive. Output: Enter your age: 15 You need 3 more years to learn to drive.

[96]:
```
# 1. Driving age check
age = int(input("Enter your age: "))

if age >= 18:
    print("You are old enough to learn to drive.")
else:
    wait_years = 18 - age
    print(f"You need {wait_years} more years to learn to drive.")
```

```
Enter your age:  4
```

```
You need 14 more years to learn to drive.
```

[98]:
```
# 1. Driving age check
age = int(input("Enter your age: "))

if age >= 18:
    print("You are old enough to learn to drive.")
else:
    wait_years = 18 - age
    print(f"You need {wait_years} more years to learn to drive.")
```

```
Enter your age:  26
```

```
You are old enough to learn to drive.
```

4.6.2 2. Compare the values of my_age and your_age using if ... else. Who is older (me or you)? Use input("Enter your age:") to get the age as input. You can use a nested condition to print 'year' for 1 year difference in age, 'years' for bigger differences, and a custom text if my_age = your_age. Output:

"'python Enter your age: 30 You are 5 years older than me.

[100]:
```python
# 2. Compare my_age and your_age
my_age = 25   # You can fix this as your age
your_age = int(input("Enter your age: "))

if your_age > my_age:
    diff = your_age - my_age
    if diff == 1:
        print("You are 1 year older than me.")
    else:
        print(f"You are {diff} years older than me.")
elif your_age < my_age:
    diff = my_age - your_age
    if diff == 1:
        print("I am 1 year older than you.")
    else:
        print(f"I am {diff} years older than you.")
else:
    print("We are the same age!")
```

```
Enter your age:  55

You are 30 years older than me.
```

4.6.3 3. Get two numbers from the user using input prompt. If a is greater than b return a is greater than b, if a is less b return a is smaller than b, else a is equal to b. Output:

"'python Enter number one: 4 Enter number two: 3 4 is greater than 3

[102]:
```python
# 3. Compare two numbers
a = int(input("Enter number one: "))
b = int(input("Enter number two: "))

if a > b:
    print(f"{a} is greater than {b}")
elif a < b:
    print(f"{a} is smaller than {b}")
else:
    print(f"{a} is equal to {b}")
```

```
Enter number one:  6
Enter number two:  9

6 is smaller than 9
```

4.6.4 3. Write a code which gives grade to students according to theirs scores:

"'python 80-100, A 70-89, B 60-69, C 50-59, D 0-49, F

[104]:
```python
# Grade calculation
score = int(input("Enter your score (0-100): "))

if 80 <= score <= 100:
    grade = "A"
elif 70 <= score <= 79:
    grade = "B"
elif 60 <= score <= 69:
    grade = "C"
elif 50 <= score <= 59:
    grade = "D"
elif 0 <= score <= 49:
    grade = "F"
else:
    grade = "Invalid score"

print(f"Your grade is: {grade}")
```

```
Enter your score (0-100):  66

Your grade is: C
```

4.6.5 2. Check if the season is Autumn, Winter, Spring or Summer. If the user input is: September, October or November, the season is Autumn. December, January or February, the season is Winter. March, April or May, the season is Spring June, July or August, the season is Summer

[106]:
```python
# Season check
month = input("Enter month: ").strip().capitalize()

if month in ["September", "October", "November"]:
    season = "Autumn"
elif month in ["December", "January", "February"]:
    season = "Winter"
elif month in ["March", "April", "May"]:
    season = "Spring"
elif month in ["June", "July", "August"]:
    season = "Summer"
else:
    season = "Invalid month"
```

```
print(f"The season is: {season}")
66
```

```
Enter month:  January

The season is: Winter
```

[106]:
```
66
```

[ ]:
```

```