

# 06-conditionals

January 25, 2022

## 1 Conditionals

### 1.1 Types:

- booleans: `True` and `False`

### 1.2 Comparison operators:

#### 1.2.1 Equality

```
[1]: x = 10  
     y = 3  
     y == x
```

```
[1]: False
```

```
[2]: 4 == 4.0
```

```
[2]: True
```

```
[3]: 3.0 == 3.5
```

```
[3]: False
```

#### 1.2.2 “Disequality” (not equal)

```
[4]: 4.0 != 4
```

```
[4]: False
```

```
[5]: 3 != 7
```

```
[5]: True
```

#### 1.2.3 Less than

```
[6]: 4 < 6
```

[6]: True

[7]: 4.0 < 3.9

[7]: False

[8]: 5 < 6 < 7

[8]: True

#### 1.2.4 Less than or equal

[9]: 32 <= 32

[9]: True

[10]: 32 <= 32.8

[10]: True

[11]: 7.5 <= 4

[11]: False

#### 1.2.5 Greater than

[12]: 5 > 6

[12]: False

[13]: 8 > 4

[13]: True

#### 1.2.6 Greater than or equal

[14]: 4.0 >= 4

[14]: True

[15]: 4.5 >= 3

[15]: True

[16]: 4.5 >= 9

[16]: False

## 1.3 Boolean operators:

### 1.3.1 And

a	b	a and b
True	True	True
True	False	False
False	True	False
False	False	False

```
[17]: True and False
```

```
[17]: False
```

```
[18]: 4 > 0 and 4 < 3
```

```
[18]: False
```

```
[19]: 4 > 0 and 5 < 10
```

```
[19]: True
```

### 1.3.2 Or

a	b	a or b
True	True	True
True	False	True
False	True	True
False	False	False

```
[20]: True or False
```

```
[20]: True
```

```
[21]: 4 < 0 or 3 > 2
```

```
[21]: True
```

```
[22]: 4 < 0 or 3 < 0
```

```
[22]: False
```

### 1.3.3 Not

a	not a
True	False
False	True

```
[23]: not 4 < 0
```

```
[23]: True
```

```
[24]: not 0 < 4
```

```
[24]: False
```

### 1.3.4 If conditions

if-else

```
[25]: def myAbs(x):
      if x < 0:
          x = -x
          # ... other commands may appear here, for example, another assignment
          ↪ or another if!
      else:
          x = x
          # else case cannot be empty
      return x

myAbs(-3)
```

```
[25]: 3
```

if (without else)

If an `else` case “does nothing”, we can get rid of it.

```
[26]: def myAbs(x):
      if x < 0:
          x = -x
      return x

myAbs(-32)
```

```
[26]: 32
```

if-elif

```
[27]: def sign(n):
      s = 0
      if n < 0:
```

```

        s = -1
    elif n > 0:
        s = 1
    #else:
    #    s = 0

    return s

sign(-43)

```

[27]: -1

## 1.4 Exercise 1: middle number

Implement the function `middle(a, b, c)` that takes three numbers as input, and returns the middle among them.

Hint: Draw a flowchart to avoid an explosion on the number of cases.

```

[28]: def middle(a, b, c):
        return 42.0

```

## 1.5 Exercise 2: card game

Tri-du is a card game inspired in the popular game of Truco. The game uses a normal deck of 52 cards, with 13 cards of each suit, but suits are ignored. What is used is the value of the cards, considered as integers between 1 to 13.

In the game, each player gets three cards. The rules are simple:

1. A Three of a Kind (three cards of the same value) wins over a Pair (two cards of the same value).
2. A Three of a Kind formed by cards of a larger value wins over a Three of a Kind formed by cards of a smaller value.
3. A Pair formed by cards of a larger value wins over a Pair formed by cards of a smaller value.

Note that the game may not have a winner in many situations; in those cases, the cards are returned to the deck, which is re-shuffled and a new game starts.

A player received already two of the three cards, and knows their values. Your task is to write a program to determine the value of the third card that maximizes the probability of that player winning the game.

Implement the function `bestCard(c1, c2)` that takes the values `c1` and `c2` of the two cards at hand, and returns the value of that card that will result on the best hand.

```

[29]: def bestCard(c1, c2):
        return 42

```

## 1.6 Exercise 3: rounding numbers

Humans like round numbers to the closest integer. That is why 99.99 becomes 100, and 2.01 becomes 2.

Implement the function `round(x)` that rounds `x` to the closest integer. That means that if `x` ends with .5 or greater, you should round the number up. Otherwise, round the number down.

```
[30]: import math

def round(x):
    return 42
```

## 1.7 Exercise 4: Libertadores

The Libertadores Cup is the main football competition among South American professional clubs, organized by the South American Confederation of Football (CONMEBOL).

During the quarter finals, two teams play two matches, one in each team's home stadium, competing to advance to the semi-final. The winning team is selected according to the following criteria, where each criteria is considered only if all previous ones resulted in a tie:

1. Highest number of goals scored;
2. Highest number of goals scored in the other teams' stadium (the "away" match)

If all criteria tie, the winner is decided with penalty shootouts.

Implement the function `winner(game1Team1, game1Team2, game2Team1, game2Team2)` that takes as input:

- the score for the match in team 1's stadium: `game1Team1` x `game1Team2` and
- the score for the match in team 2's stadium: `game2Team2` x `game2Team1`.

The function must return "Team 1" if Team 1 advances to the semi-final, "Team 2" if Team 2 advances to the semi-final, or "Penalty" if the advancing team must be decided with penalty shootouts.

```
[31]: def winner(game1Team1, game1Team2, game2Team1, game2Team2):
    return "42"
```