Introdução à Programação
Mestrado em Multimédia

T2. The game "rock, paper, scissors"

António Coelho e Luís Reis

The main objective of this tutorial is to continue the development of simple Python programs, identifying the main features of the programming language lexicon and syntax focusing on conditional structures.
More specifically we will explore the following learning objectives:

- Identify the fundamentals of the Python lexicon and syntax;
- Apply variables and primitive data types;
- Use expressions;
- Apply basic I/O functions for data entry;
- Select conditional control structures;
- Identify strings and lists;
- Identify basic game concepts;

It is recommended to read the following supporting texts before the tutorial:
How to Think Like a Computer Scientist: Learning with Python 3ed; Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers, 2012

- Chapter 5 - Conditionals

The game " rock, paper, scissors"

"Rock, paper, scissors" is a simple game that two or more people can play to decide, for example, who starts a game. Simultaneously, each player casts Rock (closed hand), Paper (open hand) or Scissors (index finger and middle finger in V).
The rules are simple:

- Rock wins against Scissors;
- Scissors win against paper;
- Paper wins against Rock.

1 Let's start...

Like the "heads or tails" game (Tutorial T1), the first part of the program is to randomly select the computer move and read the player's move.
The algorithm of this game is as follows:

1. Randomly select the computer move.
2. Read the player's move (input) and show the move of the computer.
3. Check if the player wins, loses or draws and visualize this result (output).

Thus, the player should think of an option (rock, paper or scissors) and then run the program, checking whether or not it is a win.
In Python, the code that implements the first 2 steps of the algorithm is as follows:

```
import random

# 1. randomly select the computer's move
computer = random.choice(["rock", "paper", "scissors"])

# 2. read the player's move and show the computer's
player = input("rock, paper or scissors? ")
print ("I played " + computer)
```

The random library creates the "computer" variable to which the value of "rock", "paper" or "scissors" is randomly assigned.
Then read the player's move for the variable "player" and view the result of the choice of the computer using the print function.
Now try this first version and make sure it works well.

2 Who wins the game?

This game has many possibilities and you can get a defeat, a victory or a draw. This depends on the rules of the game, also called "game mechanics".
How many possibilities will there be? With three symbols we have 3_2 = 9 possibilities (player vs computer):

- "rock" vs "scissors": Lost!
- "scissors" vs "paper": "Lost!
- "paper" vs "rock": "Lost!"
- "scissors" vs "rock": "Won!"
- "paper" vs "scissors": "Won!"
- "rock" vs "paper": "Won!"

The remaining 3 possibilities give the "Draw ...".

We will then have to make a decision between multiple possibilities. We need to use a decision framework to make a multiple selection. In Python this structure is if...elif...else. Both the first "if" and successive "elif" have conditional expressions to evaluate each move. The first expression to evaluate to true is executed and the rest of the expressions are no longer evaluated. If none of these conditional expressions is true then the code in "else" applies if it has been defined.

In Python, the code that implements the game rules defined above is as follows:

```
# 3. Verify if player wins or loses
if computer == "rock" and player == "scissors":
    print ("You lost... ")
elif computer == "scissors" and player == "paper":
    print ("You lost...")
elif computer == "paper" and player == "rock":
    print ("You lost... ")
elif computer == "scissors" and player == "rock":
    print ("You won! ")
elif computer == "paper" and player == "scissors":
    print ("You won! ")
elif computer == "rock" and player == "paper":
    print ("You won! ")
else:
    print ("It's a draw!")
```

Note that in order to evaluate each move you must ensure that both moves have a specific value. For this, a conjunction operation, defined by the relational operator `and`, is used. Thus, only when both expressions are true is the conditional expression true. In the remaining cases it is false.

ⓘ *See the truth tables of the and, or and not operators in section 5 of chapter 5 - Conditionals.*

3 Final challenge

Can game mechanics be reused in different games with different themes and / or concepts?
Develop a variation of this game by selecting another theme to replace the words "rock", "paper" and "scissors" with three others that work with the same rules.

Bibliography

4 How to Think Like a Computer Scientist: Learning with Python 3ed; Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers, 2012
Available online at: http://www.openbookproject.net/thinkcs/python/english3e/

5 Python Programming Language – Official Website.
Available online at: http://www.python.org/