mcpp_taller1.julian_ramirez

August 8, 2019

1 Taller 1

Métodos Computacionales para Políticas Públicas - URosario
Entrega: viernes 9-ago-2019 11:59 PM
Julián Santiago Ramírez julians.ramirez@urosario.edu.co

1.1 Instrucciones:

- Guarde una copia de este *Jupyter Notebook* en su computador, idealmente en una carpeta destinada al material del curso. Sugiero una estructura similar a la del repositorio del curso.
- Modifique el nombre del archivo del *notebook*, agregando al final un guión inferior y su nombre y apellido, separados estos últimos por otro guión inferior. Por ejemplo, mi *notebook* se llamaría: mcpp_taller1_santiago_matallana
- Marque el *notebook* con su nombre y e-mail en el bloque verde arriba. Reemplace el texto "[Su nombre acá]" con su nombre y apellido. Similar para su e-mail.
- Desarrolle la totalidad del taller sobre este *notebook*, insertando las celdas que sea necesario debajo de cada pregunta. Haga buen uso de las celdas para código y de las celdas tipo *markdown* según el caso.
- Recuerde salvar periódicamente sus avances.
- Cuando termine el taller:

    1. Descárguelo en PDF. Esto puede implicar instalar LaTex en su computador. Resuélvalo por su cuenta, por favor. Recuerde: Google es su amigo.
    2. Suba los dos archivos (.pdf y .ipynb) a su repositorio en GitHub antes de la fecha y hora límites. Asegúrese de que Daniel sea "colaborador" de su repositorio y de que los dos archivos queden en su repositorio, en la nube (no solo en su computador). No lo deje para última hora. Talleres subidos después de la fecha y hora límites no serán valorados, como tampoco lo serán si son remitidos vía e-mail.

(Todos los ejercicios tienen el mismo valor.)

1.2 1. Zelle, sección 1.10 (p. 17):

- "Multiple Choice", Ejercicios # 1-10.
- "Programming Exercises", Ejercicio # 1.

1.3 1. What is the fundamental question of computer science?

a) How fast can a computer compute?

b) What can be computed?

c) What is the most effective programming language?

d) How much money can a programmer make?

1.3.1 Rta:

B

1.4 2. An algorithm is like a

a) newspaper b) venus flytrap c) drum d) recipe

1.4.1 Rta:

D

1.5 3. A problem is intractable when

a) you cannot reverse its solution

b) it involves tractors

c) it has many solutions

d) it is not practical to solve

1.5.1 Rta:

D

1.6 4. Which of the following is not an example of secondary memory?

a) RAM b) hard drive c) USB flash drive d) CD-Rom

1.6.1 Rta:

A

1.7 5. Computer languages designed to be used and understood by humans are

a) natural languages

b) high-level computer languages

c) machine languages

d) fetch-execute languages

1.7.1 Rta:

B

1.8 6. A statement is

a) a translation of machine language

b) a complete computer command

c) a precise description of a problem

d) a section of an algorithm

1.8.1 Rta:

B

1.9 7. One difference between a compiler and an interpreter is

a) a compiler is a program

b) a compiler is used to translate high-level language into machine language

c) a compiler is no longer needed after a program is translated

d) a compiler processes source code

1.9.1 Rta:

C

1.10 8. By convention, the statements of a program are often placed in a function called

a) import b) main c) program d) IDLE

1.10.1 Rta:

B

1.11 9. Which of the following is not true of comments?

a) They make a program more efficient

b) They are intended for human readers

c) They are ignored by Python

d) In Python, they begin with a pound sign (#)

1.11.1 Rta:

A

1.12 10. The items listed in the parentheses of a function definition are called

a) parentheticals b) scripts c) comments d) parameters

1.12.1 Rta:

D

2 Programming exercises

1. Start up an interactive Python session and try typing in each of the following commands. Write down the results you see.

(a) print("Hello, world!")

```
In [13]: print("Hello, world!")
```

```
Hello, world!
```

2.1 Rta:

Hello, world!

(b) print("Hello", "world!")

```
In [14]: print("Hello", "world!")
```

```
Hello world!
```

2.2 Rta:

Hello world!

(c) print(3)

```
In [12]: print(3)
```

```
3
```

2.3 Rta:

3

(d) print(3.0)

```
In [11]: print(3.0)
```

```
3.0
```

2.4 Rta:

3.0

(e) print(2 + 3)

```
In [10]: print(2 + 3)
```

```
5
```

2.5 Rta:

5

(f) print(2.0 + 3.0)

```
In [9]: print(2.0 + 3.0)
```

```
5.0
```

2.6 Rta:

5.0

(g) print("2" + "3")

```
In [8]: print("2" + "3")
```

```
23
```

2.7 Rta:

23

(h) print("2 + 3 =", 2 + 3)

```
In [6]: print("2 + 3 =", 2 + 3)
```

```
2 + 3 = 5
```

2.8 Rta:

2 + 3 = 5

(i) print(2 * 3)

```
In [5]: print(2 * 3)
```

```
6
```

2.9 Rta:

6

(j) print(2 ** 3)

```
In [4]: print(2 ** 3)
```

```
8
```

2.10 Rta:

8

(k) print(2 / 3)

```
In [3]: print(2 / 3)
```

```
0.6666666666666666
```

2.11 Rta:

0.6666666666666666

En *computer science* son comunes los ejercicios denominados "pensar como un computador". Con estos usted evalúa si está comprendiendo el material, siempre y cuando no utilice un computador para correr el código del enunciado. Siempre que vea un ejercicio marcado con la etiqueta "pensar como un computador", use papel y lápiz o incluso una calculadora si es necesario para descifrar la respuesta, pero nunca ejecute el código en computador.

2.12 2. [Pensar como un computador] ¿Cuál es el valor de w después de ejecutar el siguiente código?

x = 7 y = 5.0 z = 10.0 w = x

2.12.1 Rta:

El valor de w es igual a 11.75

2.13 3. [Pensar como un computador] ¿Cuál es el valor de *c* después de ejecutar el siguiente código?

c = True d = False c = c and d c = not c or d

2.13.1 Rta:

El valor de c es igual a "True"

2.14 4. Ejecute el siguiente código y responda: ¿Por qué es falsa la tercera línea, mientras que las primeras dos son verdaderas?

```
In [18]: 1 == 1

Out[18]: True

In [16]: "1" == "1"

Out[16]: True

In [17]: 1 == "1"

Out[17]: False
```

2.14.1 Rta:

Las dos primeras lineas son verdaderas porque estoy comparando dos objetos del mismo tipo. En la primera línea, tanto el elemento de la izquierda, como el de la derecha, son de tipo "entero" y pues ambos son igual a 1; en la segunda línea, ambos elementos son de tipo "texto" o "string" y ambos son iguales a "1". En la ultima, estoy comparando elementos de diferente tipo, entonces no el computador no entiende cual seria mi verdadera intencion de preguntar en ese caso, al ser distintos no existe igualdad entre ambos por eso es falso.

2.15 5. Escriba un programa que le pida al usuario ingresar su nombre y que arroje un texto saludando de vuelta al usuario, así: "Hola, <nombre>. ¡Veo que aprendes Python rápidamente! ¡Felicitaciones!".

```
In [19]: print("Ingrese su nombre:")
         nombre=input()

         # Agregamos todo en una variable llamada "texto" que es la que se va a imprimir
         texto="Hola,"+nombre+".¡Veo que aprendes Python rápidamente!" +'\n'+"¡Felicitaciones!"
         print(texto)

Ingrese su nombre:
Julian
Hola,Julian.¡Veo que aprendes Python rápidamente!
¡Felicitaciones!
```