

06 - FUNCIONES

Jorge Muñoz
IIC1103 – Introducción a la Programación

¿ES QUE ESO NO PUEDE ...

validaRut



Validar

El dígito verificador es: K

12342932

Copy

12342932K

Copy

12.342.932-K

Copy

[Ver información Tributaria](#)

Rut? 12342932

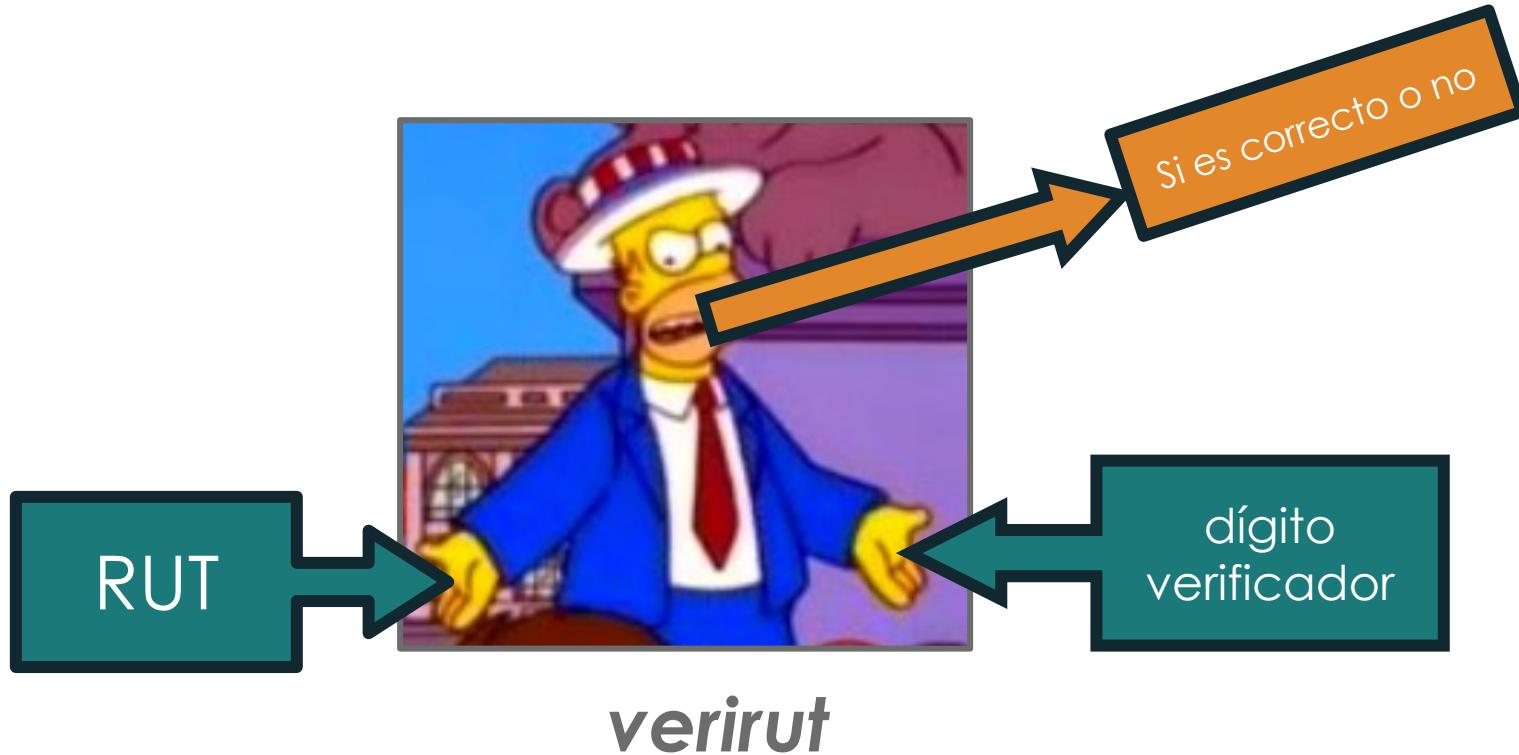
Dígito Verificador? K
RUT correcto

Rut? 12342932

Dígito Verificador? 7
RUT incorrecto

CAN'T SOMEONE ELSE DO IT?

¿ES QUE ESO NO
PUEDE HACERLO OTRO?



```
rut = input("Rut? ")
dv = input("Digito Verificador? ")

# Llamemos a mi colega 'verirut'.
# Le damos el rut, le damos el digito
# verificador, y que nos responda
# si es correcto.

# Guardemos esa respuesta (True o False)
# en una variable llamada 'correcto'
# y sigamos con el programa

if correcto:
    print("RUT correcto")
else:
    print("RUT incorrecto")
```

```
rut = input("Rut? ")
dv = input("Digito Verificador? ")

#Asi es como llamamos a mi colega verirut.
#Para verirut el orden es muy importante:
#el primero el rut, y el segundo el digito verificador

correcto = verirut(rut,dv)

if correcto:
    print("RUT correcto")
else:
    print("RUT incorrecto")
```

```
correcto = verirut(rut,dv)
```

Para mi colega (**función**) es muy importante:

- Que lo **llames** por su nombre
- Que le des el **número** de cosas (**parámetros**) que el te ha indicado
- En el **orden** que el te ha indicado
- Y del **tipo** que el te ha indicado

E.g., “verirut acepta primero un str con el rut y segundo un str con el dígito de verificación”

Y a cambio se compromete:

- **Retornarte** la respuesta correcta en el **formato** que hayais acordado
“verirut te retornará un bool indicando si el rut está correcto”

```
if correcto:  
    print("RUT correcto")  
else:  
    print("RUT incorrecto")
```

¿No debería ser ...

```
if correcto == True:  
    print("RUT correcto")  
else:  
    print("RUT incorrecto")
```

Fíjate que hace exactamente lo mismo 😊



- Calcular dígito verificador del RUT
 - <https://www.biobiochile.cl/noticias/2016/03/07/para-que-sirve-y-como-se-calcula-el-dígito-verificador-de-nuestro-rut.shtml>
- Simpsons (e22s09): Trash of the Titans
 - https://en.wikipedia.org/wiki/Trash_of_the_Titans

FUNCIONES

Built-in

No hay que instalar nada ni hacer nada
Solo llamarlas

¿Hemos utilizado ya alguna función?



```
x = str(5)
```

La **función** se llama **str**,

Recibe una sola **parametro**: un **int**

Retorna un **string** con el valor del **int** que le has dado

```
x = str(5)
y = int("5")

print(x,y)

z = input("Dame algo? ")
hr = input()

t = int(input("Dame un numero? "))
```

- **Input** es un poco especial
 - Hay una versión que recibe solo UN str
 - Pero hay otra (que usamos en HR) que no recibe NADA
- **Print** es un poco especial
 - No retorna nada
 - Puede recibir 0 cosas, 1, 2, 3,
- Puedes **encadenar** llamadas a funciones
 - Lo que te retorna una, se la das a otra, etc

FUNCIONES

Importables ya instaladas

No hay que instalar nada ...

... pero si hay que decirle a Python que
las vas a usar

Usaremos: **import**



¿La del número al azar del cachipun y las tortugas ninja?

import elmodulo

- Hacemos import del modulo donde está la función
- Se acostumbra poner los imports al **principio** del programa

x = elmodulo.lafuncion(...)

- Luego para llamar a la función usamos el nombre del modulo, punto, y nombre de la función
- Podemos llamarla tantas **veces** como queramos (con un import al principio vale)

Solo entran 2 funciones en el temario

```
x = math.sqrt(a)
```

```
x = random.randint(a,b)
```

```
import math
x = math.sqrt(a)
#Recibe solo una cosa: un numero (int o float)
#Retorna un float con la raiz cuadrada del numero
```

```
import math  
  
s4 = math.sqrt(4)  
s9 = math.sqrt(9)  
  
print("La raiz cuadrada de 4 es",s4)  
print("La raiz cuadrada de 9 es",s9)
```

La raiz cuadrada de 4 es 2.0
La raiz cuadrada de 9 es 3.0

```
import random
x = random.randint(a,b)

#Recibe dos int
#Retorna un int al azar entre 'a' y 'b' (ambos incluidos)
#cada vez que se llama
```

```
import random

ini = int(input("Entre el ...? "))
fin = int(input(" ... y el ...? "))

n1 = random.randint(ini,fin)
n2 = random.randint(ini,fin)
n3 = random.randint(ini,fin)

print("Los tres numeros al azar entre el",ini,"y el",fin,"(ambos incluidos) son", n1, n2, n3)
```

```
Entre el ...? 1
... y el ...? 100
Los tres numeros al azar entre el 1 y el 100 (ambos incluidos) son 47 66 93
```

```
Entre el ...? 1
... y el ...? 100
Los tres numeros al azar entre el 1 y el 100 (ambos incluidos) son 64 14 45
```

```
Entre el ...? 1
... y el ...? 100
Los tres numeros al azar entre el 1 y el 100 (ambos incluidos) son 53 83 90
```

Todas las otras funciones de random y
math, o todos los otros módulos ...

.... FUERA DEL TEMARIO

Curso de **Pensamiento Algorítmico**, no
de memorizar funciones



- Python Random
 - <https://docs.python.org/3/library/random.html>
- Python Math
 - <https://docs.python.org/3/library/math.html>



memesintroalaprogr • Following ...



9 likes

SEPTEMBER 23, 2019



Add a comment...

Post

CALCULADORA PROGRAMABLE



INTERESES

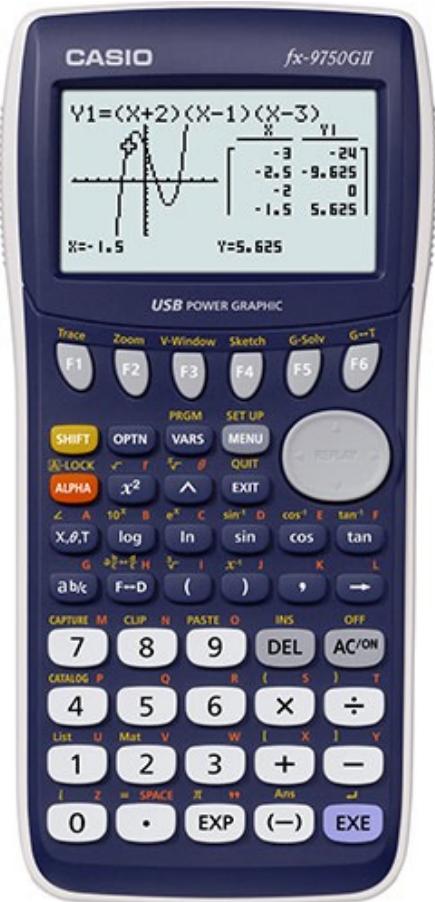
“...**ramos**, sobretodo los matemáticos”

“**Automatizar** cosas”

“ecuaciones diferenciales”



Situación
Real





■ Creando un programa

□ Creando un programa nuevo

Ejemplo: Crear un programa que determine el área de superficie de tres octaedros regulares, cada uno de los lados con un largo de 7 cm, 10 cm y 15 cm.

A? EXE

(Aparecerá un prompt para que se introduzca un valor para la variable A).

Introduzca 7 para A:

7 EXE A? Disp

7 169.7409791

(Se visualizará el resultado calculado en el área de superficie S).

□ Comandos de programa

Al seleccionar **FUNCTION** – {PROG} en la pantalla de edición de programas se visualizará el menú de los comandos de programa que se pueden usar para introducir el comando.

PRGM ▼

1:?	2:→
3:If	4:Then
5:Else	6:IfEnd
7:Lbl	8:Goto

FUNCTION – {PROG}

◀ ▶ ↑ ↓



When tu papá es ingeniero



$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$



$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Se usa \pm para indicar las dos soluciones:

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad \text{y} \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

$$x^2 - 5x + 6 = 0 \qquad x = \frac{5 \pm \sqrt{5^2 - 4 \cdot 6}}{2} = \frac{5 \pm \sqrt{25 - 24}}{2} = \frac{5 \pm 1}{2} =$$

$\nearrow x_1 = \frac{6}{2} = 3$
 $\searrow x_2 = \frac{4}{2} = 2$

>>>

a: 1
b: -5
c: 6

Las soluciones son 3.0 y 2.0

>>>





I'LL WAIT
FOR YOU HERE



$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Se usa \pm para indicar las dos soluciones:

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad \text{y} \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

$$x^2 - 5x + 6 = 0 \qquad x = \frac{5 \pm \sqrt{5^2 - 4 \cdot 6}}{2} = \frac{5 \pm \sqrt{25 - 24}}{2} = \frac{5 \pm 1}{2} =$$

$\nearrow x_1 = \frac{6}{2} = 3$
 $\searrow x_2 = \frac{4}{2} = 2$

>>>

a: 1
b: -5
c: 6

Las soluciones son 3.0 y 2.0

>>>



```
import math

a = int(input("a: "))
b = int(input("b: "))
c = int(input("c: "))

s1 = (b**2) - (4*a*c)
t1 = (-b) + math.sqrt(s1)
x1 = t1 / (2*a)

s2 = (b**2) - (4*a*c)
t2 = (-b) - math.sqrt(s2)
x2 = t2 / (2*a)

print("Las soluciones son", x1, "y", x2)
```

BARCELONA 92



INTERESES

“aviones, **atletismo**, carreras de autos, barcos, fabricas”

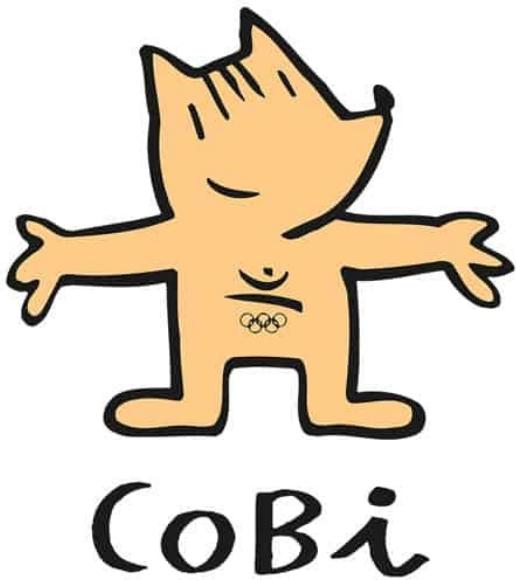
“deportes, **atletismo** y golf”

“futbol, juegos **olímpicos**, deportes”

“**Juegos** divertidos”

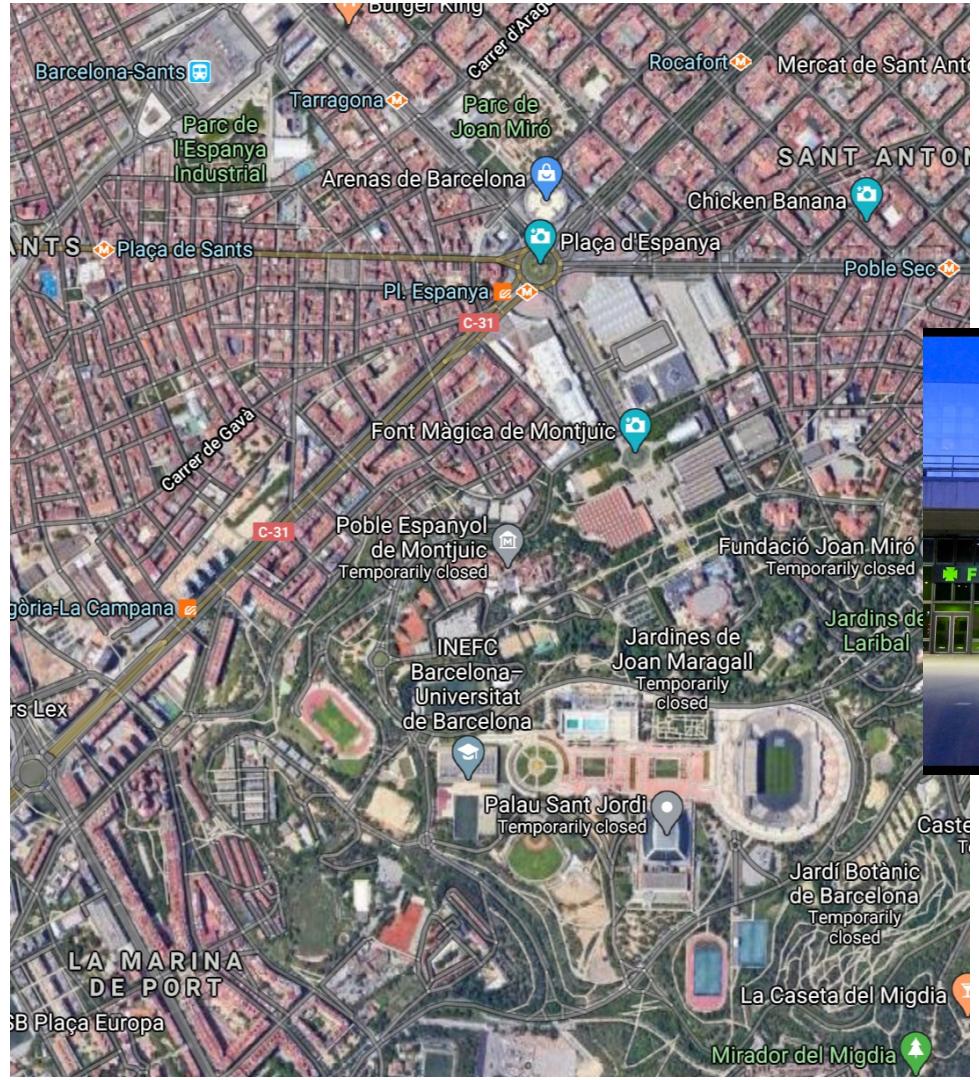


Situación
Real



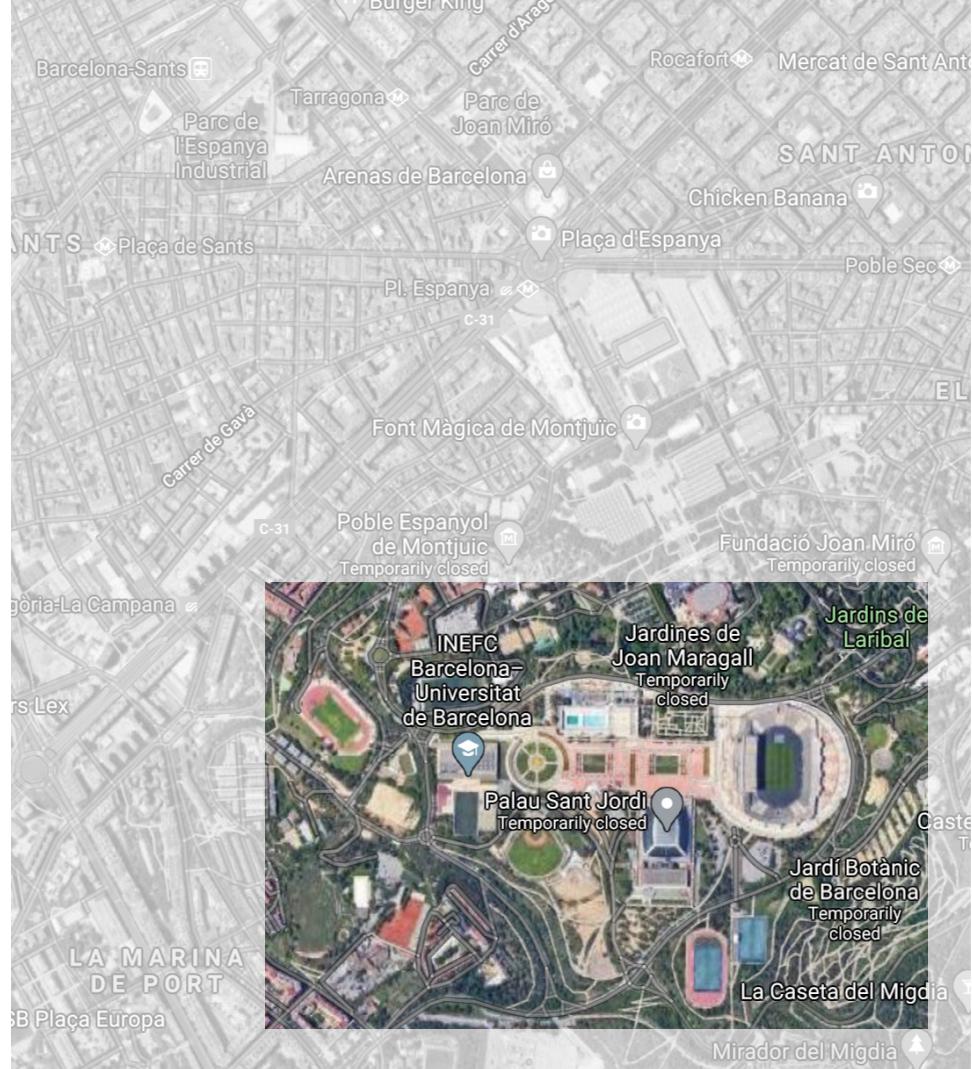


Situación
Real



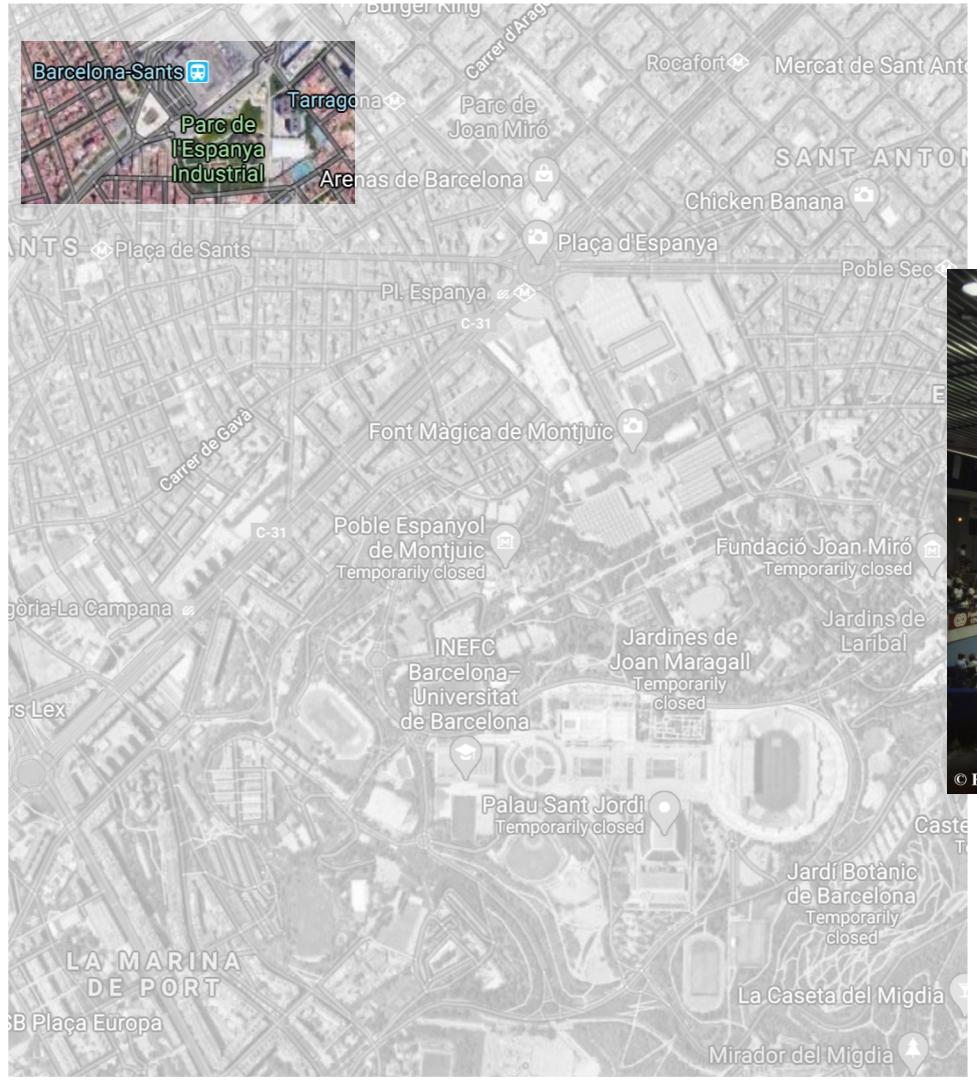


Situación
Real



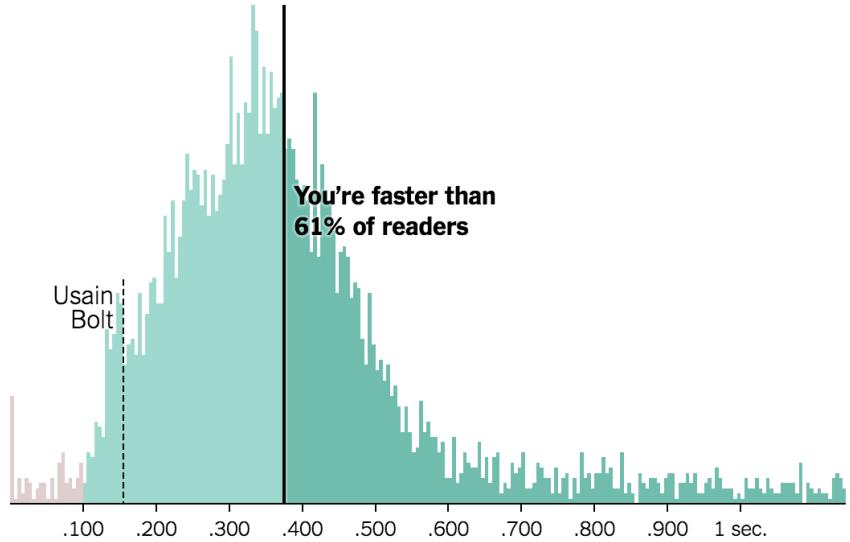


Situación
Real





Situación
Real



<http://www.nytimes.com/interactive/2016/08/13/sports/olympics/can-you-beat-usain-bolt-out-of-the-blocks.html>



preparados
listos
YA!

Tu tiempo de reaccion ha sido 0.6174020767211914

preparados
listos
YA!

Tu tiempo de reaccion ha sido 0.3508009910583496



Tiempos al azar:

- Preparados (entre 1 y 3 seg)
- Listos (entre 1 y 3 seg)
- Ya (entre 1 y 5)

preparados
listos
YA!

Tu tiempo de reaccion ha sido 0.6174020767211914

Modulos y funciones

- random
 - random.randint(x,y)
- time (**FUERA DEL TEMARIO**)
 - time.sleep(x)
 - Para el programa durante x segundos
 - time.time():
 - Retorna los seg des de 01/01/1970
- ¿cómo hacer que se quede esperando hasta que apretemos algo?
 - Se usa una de las primeras funciones del curso ☺





I'LL WAIT
FOR YOU HERE



Tiempos al azar:

- Preparados (entre 1 y 3 seg)
- Listos (entre 1 y 3 seg)
- Ya (entre 1 y 5)

preparados
listos
YA!

Tu tiempo de reaccion ha sido 0.6174020767211914

Modulos y funciones

- random
 - random.randint(x,y)
- time (**FUERA DEL TEMARIO**)
 - time.sleep(x)
 - Para el programa durante x segundos
 - time.time():
 - Retorna los seg des de 01/01/1970
- ¿cómo hacer que se quede esperando hasta que apretemos algo?
 - Se usa una de las primeras funciones del curso ☺



```
import time
import random

preparados_espera = random.randint(1,3)
time.sleep(preparados_espera)
print("preparados")

listos_espera = random.randint(1,3)
time.sleep(listos_espera)
print("listos")

ya_espera = random.randint(1,5)
time.sleep(ya_espera)
inicio = time.time()
x = input("YA!")
fin = time.time()

reaccion = fin - inicio
print("Tu tiempo de reaccion ha sido", reaccion)
```



- Can you beat Usain Bolt
 - <http://www.nytimes.com/interactive/2016/08/13/sports/olympics/can-you-beat-usain-bolt-out-of-the-blocks.html>
- Barcelona 92
 - <https://www.youtube.com/watch?v=ck2jpaLoOMA>
- Freddie Mercury - Barcelona
 - <https://www.youtube.com/watch?v=q0wdxj8-mAU>



QUINIOLA



INTERESES

“Predicciones sobre deportes”

“futbol, **juegos de azar**, deportes, casino,
emprendimientos”

“deportes”

“futbol”



“La lotería es el impuesto de los que no saben matemáticas”

Anónimo



Situación Real

1.^a LaLiga Santander / Liga Femenina Iberdrola JORNADA: 44.^a FECHA: 11-3-18 |



LA QUINIOLA

1. Granada	- Espanyol	2-1	1
2. Real Madrid	- At. Madrid	1-0	1
3. Mallorca	- Valladolid	0-1	2
4. Valencia	- Celta	1-0	1
5. Leganés	- R. Sociedad	2-1	1
6. Eibar	- Betis	1-1	X
7. Athletic Club	- Getafe	0-2	2
8. Sevilla	- Alavés	1-1	X
9. Ponferradina	- Huesca	3-1	1
10. Elche	- Málaga	2-0	1
11. Fuenlabrada	- Girona	0-1	2
12. Cádiz	- R. Zaragoza	1-1	X
13. R. Oviedo	- Albacete	3-1	1
14. Racing	- Alcorcón	1-1	X

Estadísticamente

- '1' es 50%
- 'X' es 30%
- '2' es 20%



1 => 2
2 => 2
3 => 1
4 => 2
5 => X
6 => 1
7 => X
8 => 2
9 => 1
10 => X
11 => X
12 => 1
13 => X
14 => 1

1 => 2
2 => 1
3 => X
4 => X
5 => X
6 => 1
7 => 1
8 => 1
9 => X
10 => 1
11 => 1
12 => 2
13 => 2
14 => 2

Estadísticamente

- '1' es 50%
- 'X' es 30%
- '2' es 20%





I'LL WAIT
FOR YOU HERE

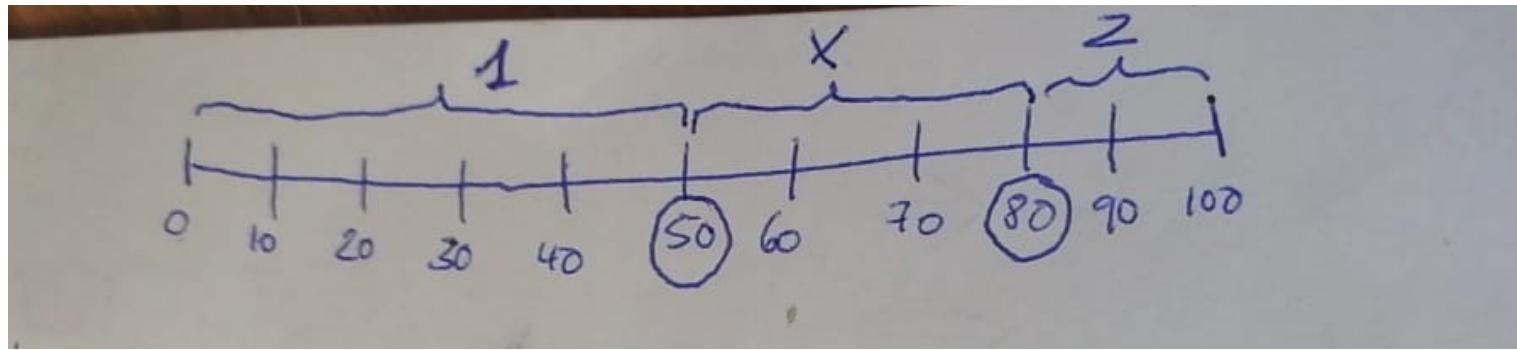


1 => 2
2 => 2
3 => 1
4 => 2
5 => X
6 => 1
7 => X
8 => 2
9 => 1
10 => X
11 => X
12 => 1
13 => X
14 => 1

1 => 2
2 => 1
3 => X
4 => X
5 => X
6 => 1
7 => 1
8 => 1
9 => X
10 => 1
11 => 1
12 => 2
13 => 2
14 => 2

Estadísticamente

- '1' es 50%
- 'X' es 30%
- '2' es 20%





```
import random

lim1 = 50
lim2 = 80

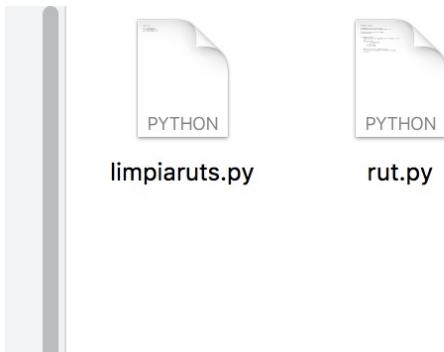
for i in range(1,15):
    n = random.randint(1,100)
    if n < lim1:
        s = "1"
    elif n > lim2:
        s = "2"
    else:
        s = "X"

    print(i,"=>",s)
```

1 =>	2
2 =>	2
3 =>	1
4 =>	2
5 =>	X
6 =>	1
7 =>	X
8 =>	2
9 =>	1
10 =>	X
11 =>	X
12 =>	1
13 =>	X
14 =>	1

FUNCIONES

Importar de archivo



limpiaruts.py -

```
import rut

rs = input("RUT? ")
rl = rut.limpiar(rs)
print("RUT LIMPIO:", rl)
```

rut.py -

```
# Nombre 'limpiar'

# Recibe un str con el rut sucio
# (con/sin puntos, guion, espacios, k/K, ...)

# Retorna un str con el rut limpio
# (xx.xxx.xxx-y)
```

RUT? 12345678k
 RUT LIMPIO: 12.345.678-K

RUT? 12.345.678.K
 RUT LIMPIO: 12.345.678-K

RUT? 12 345 678 K
 RUT LIMPIO: 12.345.678-K

RUT? 12345678-k
 RUT LIMPIO: 12.345.678-K

RUT? 12-345-678-k
 RUT LIMPIO: 12.345.678-K

- miprograma.py y milibreria.py en la **misma carpeta**
- miprograma.py puede usar las funciones definidas en milibreria.py haciendo **import milibreria** y usandolas normalmente como **milibreria.funX(...)**

WARGAMES

INTERESES

“videojuegos”

“juegos”

“películas o series”

“historia de la computación”



- Acaban llenando el tablero sin que nadie gane

EMPIEZA EL JUEGO

| |
-+--
| |
-+--
| |

Turno Jugador 1

Fila? 1
Columna? 1

| |
-+--
| □ |
-+--
| |

Turno Jugador 2

Fila? 0
Columna? 0

■ | |
-+--
| □ |
-+--
| |

Turno Jugador 1

Fila? 0
Columna? 2

■ | □ |
-+--
| □ |
-+--
| |

Turno Jugador 2

Fila? 2
Columna? 0

■ | □ | □ |
-+--
| □ |
-+--
■ | |

Turno Jugador 1

Fila? 1
Columna? 0

■ | □ | □ |
-+--
□ | □ |
-+--
■ | |

Turno Jugador 2

Fila? 1
Columna? 2

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | |

Turno Jugador 1

Fila? 0
Columna? 1

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | |

Turno Jugador 2

Fila? 2
Columna? 1

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | ■ |

Turno Jugador 1

Fila? 2
Columna? 2

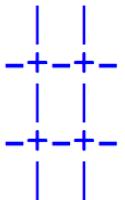
Empate

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | ■ | □ |

ACABA EL JUEGO

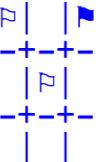


EMPIEZA EL JUEGO



Turno Jugador 1

Fila? 0
Columna? 0

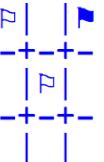
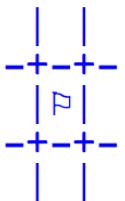


Turno Jugador 1

Fila? 1
Columna? 1

Turno Jugador 2

Fila? 1
Columna? 1
Casilla Ocupada! Persiste el turno

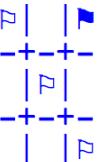
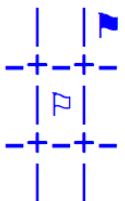


Turno Jugador 2

Fila? 0
Columna? 2

Turno Jugador 1

Fila? 2
Columna? 2
Ganaste Jugador 1



- J2 pierde un turno
- J1 gana

ACABA EL JUEGO



```
#Recibe 2 ints: fila y columna
#Retorna bool si la celda esta libre
def esta_libre(f, c):

#Recibe 3 ints: fila, columna, y jugador (1 o 2)
#Pone la ficha del jugador en la celda
def poner_ficha(f, c, j):

#No Recibe nada
#Retorna bool si todo el tablero esta lleno
def ocupado():

#No recibe nada
#Retorna int con el ganador:
# 1 (Jug.1), 2(Jug.2), 0 (Aun no hay ganador)
def ganador():

# No recibe nada
# Retorna str con el tablero listo para hacer print
def tablero_imprimible():
```



gato.py



wargames.py







I'LL WAIT
FOR YOU HERE



```
#Recibe 2 ints: fila y columna
#Retorna bool si la celda esta libre
def esta_libre(f, c):

#Recibe 3 ints: fila, columna, y jugador (1 o 2)
#Pone la ficha del jugador en la celda
def poner_ficha(f, c, j):

#No Recibe nada
#Retorna bool si todo el tablero esta lleno
def ocupado():

#No recibe nada
#Retorna int con el ganador:
# 1 (Jug.1), 2(Jug.2), 0 (Aun no hay ganador)
def ganador():

# No recibe nada
# Retorna str con el tablero listo para hacer print
def tablero_imprimible():
```



gato.py



wargames.py





- Acaban llenando el tablero sin que nadie gane

EMPIEZA EL JUEGO

| |
-+--
| |
-+--
| |

Turno Jugador 1

Fila? 1
Columna? 1

| |
-+--
| □ |
-+--
| |

Turno Jugador 2

Fila? 0
Columna? 0

■ | |
-+--
| □ |
-+--
| |

Turno Jugador 1

Fila? 0
Columna? 2

■ | □ |
-+--
| □ |
-+--
| |

Turno Jugador 2

Fila? 2
Columna? 0

■ | □ | □ |
-+--
| □ |
-+--
■ | |

Turno Jugador 1

Fila? 1
Columna? 0

■ | □ | □ |
-+--
□ | □ |
-+--
■ | |

Turno Jugador 2

Fila? 1
Columna? 2

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | |

Turno Jugador 1

Fila? 0
Columna? 1

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | |

Turno Jugador 2

Fila? 2
Columna? 1

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | ■ |

Turno Jugador 1

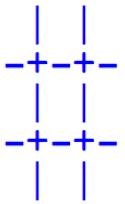
Fila? 2
Columna? 2
Empate

■ | □ | □ |
-+--
□ | □ | ■ |
-+--
■ | ■ | □ |

ACABA EL JUEGO

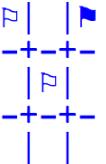


EMPIEZA EL JUEGO



Turno Jugador 1

Fila? 0
Columna? 0

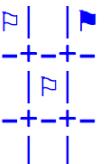
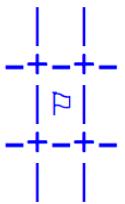


Turno Jugador 1

Fila? 1
Columna? 1

Turno Jugador 2

Fila? 1
Columna? 1
Casilla Ocupada! Persiste el turno

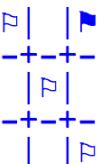
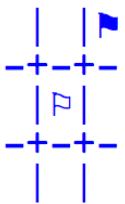


Turno Jugador 2

Fila? 0
Columna? 2

Turno Jugador 1

Fila? 2
Columna? 2
Ganaste Jugador 1



- J2 pierde un turno
- J1 gana

ACABA EL JUEGO



```
import gato
print("EMPIEZA EL JUEGO")
j = 1 #De que jugador es el turno
continuar = True

while continuar:
    #Imprimir el tablero actual
    t = gato.tablero_imprimible()
    print(t)
```

...

...

```
t = gato.tablero_imprimible()
print(t)
print("ACABA EL JUEGO")
```



```
import gato

print("EMPIEZA EL JUEGO")

j = 1 #De que jugador es el turno
continuar = True

while continuar:

    #Imprimir el tablero actual
    t = gato.tablero_imprimible()
    print(t)

    #Pedir Tirada
    print("Turno Jugador",j)
    f = int(input("Fila? "))
    c = int(input("Columna? "))

    #Hacer tirada
    libre = gato.esta_libre(f,c)
    if libre:
        gato.poner_ficha(f,c,j)
    else:
        print("Casilla Ocupada! Persiste el turno")
```

...

...

```
t = gato.tablero_imprimible()
print(t)
print("ACABA EL JUEGO")
```



```
import gato

print("EMPIEZA EL JUEGO")

j = 1 #De que jugador es el turno
continuar = True

while continuar:

    #Imprimir el tablero actual
    t = gato.tablero_imprimible()
    print(t)

    #Pedir Tirada
    print("Turno Jugador",j)
    f = int(input("Fila? "))
    c = int(input("Columna? "))

    #Hacer tirada
    libre = gato.esta_libre(f,c)
    if libre:
        gato.poner_ficha(f,c,j)
    else:
        print("Casilla Ocupada! Persiste el turno")

    #Comprobar ganador, empate, o continuar
    g = gato.ganador()

    if g == 1:
        print("Ganaste Jugador 1")
        continuar = False

    elif g == 2:
        print("Ganaste Jugador 2")
        continuar = False
```

...

```
t = gato.tablero_imprimible()
print(t)
print("ACABA EL JUEGO")
```



```
import gato

print("EMPIEZA EL JUEGO")

j = 1 #De que jugador es el turno
continuar = True

while continuar:

    #Imprimir el tablero actual
    t = gato.tablero_imprimible()
    print(t)

    #Pedir Tirada
    print("Turno Jugador",j)
    f = int(input("Fila? "))
    c = int(input("Columna? "))

    #Hacer tirada
    libre = gato.esta_libre(f,c)
    if libre:
        gato.poner_ficha(f,c,j)
    else:
        print("Casilla Ocupada! Persiste el turno")

    #Comprobar ganador, empate, o continuar
    g = gato.ganador()

    if g == 1:
        print("Ganaste Jugador 1")
        continuar = False

    elif g == 2:
        print("Ganaste Jugador 2")
        continuar = False

    else:
        ocupado = gato.ocupado()

        if ocupado:
            print("Empate")
            continuar = False

        else:
            #Cambiar el turno
            if j == 1:
                j = 2
            else:
                j = 1

    t = gato.tablero_imprimible()
    print(t)
    print("ACABA EL JUEGO")
```



¿Jorge, por qué le has llamado a este
problema WarGames?



Bibliografía &
Investigació
n



- Wargames

- Technology of Wargames: <https://www.imsai.net/movies/wargames.htm>
- <https://www.imdb.com/title/tt0086567/>

CORONAVIRUS



INTERESES

“CORONAVIRUS”

“ lo que estamos viviendo hoy en día, las protestas, el **coronavirus**, etc...”

“la realidad actual de chile, **coronavirus**”



Estimados,

Complementando el correo del decano, les transmito la información que tengo. El rector es miembro de la "Mesa Social Covid 19" en la que además participan ministros y subsecretarios sectoriales, alcaldes, colegio médico, OMS y los rectores de la UC y UCH. En reunión de este Domingo, la mesa definió 10 temas (ver más abajo) y se le pidió a la UC coordinar 4 de ellos. Para cada uno de esos temas, el VRI nombró a un coordinador y a un "equipo nuclear" que son los encargados de coordinar ese tema. Los temas y coordinadores son:

1. Mayor detección de casos y trazabilidad de contactos, colaboración de estudiantes y personal de las universidades. Coordinador: Blanca Peñaloza.
2. Implementación de mayor número de tests. Coordinador: Sandra Solari.
3. Trabajo interdisciplinario académico al interior de las universidades, lo que debe incluir UC, UCH y todo el sistema de Educación Superior.
Coordinador: Pablo Marquet.
4. Apoyo a desarrollo de vacuna Covid19. Coordinador: Alexis Kallergis.

Yo participo en el equipo nuclear del tema 3, que enfocará el trabajo principalmente en desarrollo de iniciativas complementarias, relacionada con desarrollo de apps, protocolos alternativos de monitoreo y trazabilidad (big data), telemedicina, equipos auxiliares (e.g. ventiladores, cánulas) de rápida manufactura (e.g. fablabs), modelos predictivos, logística y distribución, entre otros. En este tema está participando también la Escuela de Diseño que ya coordinó una red de fablabs.





MARCELO ARENAS

Profesor Titular

Departamento de Ciencia
de la Computación



<https://www.litoralpress.cl/deposito/videosm/2020/03/29/8932528.mp4>





```
INI SYS CORONA
1-Abrir 2-Cerrar 0-Exit? 1
Edad? 33
Caso abierto con ID 0
1-Abrir 2-Cerrar 0-Exit? 2
ID de Caso? 0
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 0-Exit? 1
Edad? 55
Caso abierto con ID 1
1-Abrir 2-Cerrar 0-Exit? 1
Edad? 88
Caso abierto con ID 2
1-Abrir 2-Cerrar 0-Exit? 2
ID de Caso? 2
Resultado 0-Neg 1-Pos? 0
1-Abrir 2-Cerrar 0-Exit? 2
ID de Caso? 1
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 0-Exit? 0
END SYS CORONA
```



```
#Abre un nuevo caso antes de haer el analisis
#Recibe un int con la edad del paciente
#Retorna un int con el identificador del caso
def abrir(edad):

#Cierra el caso registrando el resultado del analisis
#Recibe un int con el identificador del caso
#y un int con el resultado (0-Neg o 1-Pos)
#No retorna nada
def cerrar(cid,res):
```





I'LL WAIT
FOR YOU HERE



```
INI SYS CORONA
1-Abrir 2-Cerrar 0-Exit? 1
Edad? 33
Caso abierto con ID 0
1-Abrir 2-Cerrar 0-Exit? 2
ID de Caso? 0
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 0-Exit? 1
Edad? 55
Caso abierto con ID 1
1-Abrir 2-Cerrar 0-Exit? 1
Edad? 88
Caso abierto con ID 2
1-Abrir 2-Cerrar 0-Exit? 2
ID de Caso? 2
Resultado 0-Neg 1-Pos? 0
1-Abrir 2-Cerrar 0-Exit? 2
ID de Caso? 1
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 0-Exit? 0
END SYS CORONA
```



```
#Abre un nuevo caso antes de haer el analisis
#Recibe un int con la edad del paciente
#Retorna un int con el identificador del caso
def abrir(edad):

#Cierra el caso registrando el resultado del analisis
#Recibe un int con el identificador del caso
#y un int con el resultado (0-Neg o 1-Pos)
#No retorna nada
def cerrar(cid,res):
```



```
import lab

print("INI SYS CORONA")

continuar = True
while continuar:
    op = int(input("1-Abrir 2-Cerrar 0-Exit? "))

    if op == 0:
        continuar = False

    elif op == 1:
        edad = int(input("Edad? "))
        cid = lab.abrir(edad)
        print("Caso abierto con ID", cid)

    elif op == 2:
        cid = int(input("ID de Caso? "))
        res = int(input("Resultado 0-Neg 1-Pos? "))
        lab.cerrar(cid,res)

print("END SYS CORONA")
```



```
INI SYS CORONA
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 33
Caso abierto con ID 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 0
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 55
Caso abierto con ID 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 88
Caso abierto con ID 2
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 2
Resultado 0-Neg 1-Pos? 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 1
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 11
Caso abierto con ID 3
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 3
0-R.Casos? 0
ID 0 33 1
ID 1 55 1
ID 2 88 0
ID 3 11 -1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 0
END SYS CORONA
```



```
#Abre un nuevo caso antes de haer el analisis
#Recibe un int con la edad del paciente
#Retorna un int con el identificador del caso
def abrir(edad):

    #Cierra el caso registrando el resultado del analisis
    #Recibe un int con el identificador del caso
    #y un int con el resultado (0-Neg o 1-Pos)
    #No retorna nada
    def cerrar(cid,res):

        #Retorna el numero de casos abiertos
        #No recibe nada.
        #Retorna un int con el numero de casos
        #Nota: los caso se identifican con un int del 0 al ncasos-1
        def ncasos():

            #Retorna la edad del paciente del caso
            #Recibe un int con el identificador del caso
            #Retorna un int con la edad del caso
            def edad(cid):

                #Retorna el resultado del analisis del caso
                #Recibe un int con el identificador del caso
                #Retorna un int con el resultado: Neg(0), Pos(1), Sin Resultado(-1)
                def resultado(cid):
```





I'LL WAIT
FOR YOU HERE



```
INI SYS CORONA
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 33
Caso abierto con ID 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 0
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 55
Caso abierto con ID 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 88
Caso abierto con ID 2
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 2
Resultado 0-Neg 1-Pos? 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 1
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 11
Caso abierto con ID 3
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 3
0-R.Casos? 0
ID 0 33 1
ID 1 55 1
ID 2 88 0
ID 3 11 -1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 0
END SYS CORONA
```



```
#Abre un nuevo caso antes de haer el analisis
#Recibe un int con la edad del paciente
#Retorna un int con el identificador del caso
def abrir(edad):

    #Cierra el caso registrando el resultado del analisis
    #Recibe un int con el identificador del caso
    #y un int con el resultado (0-Neg o 1-Pos)
    #No retorna nada
    def cerrar(cid,res):

        #Retorna el numero de casos abiertos
        #No recibe nada.
        #Retorna un int con el numero de casos
        #Nota: los caso se identifican con un int del 0 al ncasos-1
        def ncasos():

            #Retorna la edad del paciente del caso
            #Recibe un int con el identificador del caso
            #Retorna un int con la edad del caso
            def edad(cid):

                #Retorna el resultado del analisis del caso
                #Recibe un int con el identificador del caso
                #Retorna un int con el resultado: Neg(0), Pos(1), Sin Resultado(-1)
                def resultado(cid):
```



```
import lab

print("INI SYS CORONA")

continuar = True
while continuar:
    op = int(input("1-Abrir 2-Cerrar 3-Reportes 0-Exit? "))

    if op == 0:
        continuar = False

    elif op == 1:
        edad = int(input("Edad? "))
        cid = lab.abrir(edad)
        print("Caso abierto con ID", cid)

    elif op == 2:
        cid = int(input("ID de Caso? "))
        res = int(input("Resultado 0-Neg 1-Pos? "))
        lab.cerrar(cid,res)

    elif op == 3:
        opr = int(input("0-R.Casos? "))

        if opr == 0:
            ncs = lab.ncasos()
            for i in range(0,ncs):
                edad = lab.edad(i)
                res = lab.resultado(i)
                print("ID",i,edad,res)

print("END SYS CORONA")
```



```
INI SYS CORONA
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 33
Caso abierto con ID 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 0
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 55
Caso abierto con ID 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 88
Caso abierto con ID 2
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 2
Resultado 0-Neg 1-Pos? 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 1
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 11
Caso abierto con ID 3
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 3
0-R.Casos 1-R.J/V? 1
Jovenes: 1 / 1
Viejos (+50): 1 / 2
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 0
END SYS CORONA
```



```
#Abre un nuevo caso antes de haer el analisis
#Recibe un int con la edad del paciente
#Retorna un int con el identificador del caso
def abrir(edad):

    #Cierra el caso registrando el resultado del analisis
    #Recibe un int con el identificador del caso
    #y un int con el resultado (0-Neg o 1-Pos)
    #No retorna nada
    def cerrar(cid,res):

        #Retorna el numero de casos abiertos
        #No recibe nada.
        #Retorna un int con el numero de casos
        #Nota: los caso se identifican con un int del 0 al ncasos-1
        def ncasos():

            #Retorna la edad del paciente del caso
            #Recibe un int con el identificador del caso
            #Retorna un int con la edad del caso
            def edad(cid):

                #Retorna el resultado del analisis del caso
                #Recibe un int con el identificador del caso
                #Retorna un int con el resultado: Neg(0), Pos(1), Sin Resultado(-1)
                def resultado(cid):
```





I'LL WAIT
FOR YOU HERE



```
INI SYS CORONA
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 33
Caso abierto con ID 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 0
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 55
Caso abierto con ID 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 88
Caso abierto con ID 2
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 2
Resultado 0-Neg 1-Pos? 0
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 2
ID de Caso? 1
Resultado 0-Neg 1-Pos? 1
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 1
Edad? 11
Caso abierto con ID 3
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 3
0-R.Casos 1-R.J/V? 1
Jovenes: 1 / 1
Viejos (+50): 1 / 2
1-Abrir 2-Cerrar 3-Reportes 0-Exit? 0
END SYS CORONA
```



```
#Abre un nuevo caso antes de haer el analisis
#Recibe un int con la edad del paciente
#Retorna un int con el identificador del caso
def abrir(edad):

    #Cierra el caso registrando el resultado del analisis
    #Recibe un int con el identificador del caso
    #y un int con el resultado (0-Neg o 1-Pos)
    #No retorna nada
    def cerrar(cid,res):

        #Retorna el numero de casos abiertos
        #No recibe nada.
        #Retorna un int con el numero de casos
        #Nota: los caso se identifican con un int del 0 al ncasos-1
        def ncasos():

            #Retorna la edad del paciente del caso
            #Recibe un int con el identificador del caso
            #Retorna un int con la edad del caso
            def edad(cid):

                #Retorna el resultado del analisis del caso
                #Recibe un int con el identificador del caso
                #Retorna un int con el resultado: Neg(0), Pos(1), Sin Resultado(-1)
                def resultado(cid):
```



```
import lab
print("INI SYS CORONA")

continuar = True
while continuar:
    op = int(input("1-Abrir 2-Cerrar 3-Reportes 0-Exit? "))

    if op == 0:
        continuar = False

    elif op == 1:
        edad = int(input("Edad? "))
        cid = lab.abrir(edad)
        print("Caso abierto con ID", cid)

    elif op == 2:
        cid = int(input("ID de Caso? "))
        res = int(input("Resultado 0-Neg 1-Pos? "))
        lab.cerrar(cid,res)

    elif op == 3:
        opr = int(input("0-R.Casos 1-R.J/V? "))

        if opr == 0:
            ncs = lab.ncasos()
            for i in range(0,ncs):
                edad = lab.edad(i)
                res = lab.resultado(i)
                print("ID",i,edad,res)

        elif opr == 1:
            ncj = 0
            ncv = 0
            npj = 0
            npv = 0

            ncs = lab.ncasos()
            for i in range(0,ncs):
                edad = lab.edad(i)
                res = lab.resultado(i)

                if edad <= 50: #Jovenes
                    if res != -1: #Hay resultado
                        ncj += 1
                        if res == 1: #Es positivo
                            npj += 1
                else: #Viejos
                    if res != -1: #Hay resultado
                        ncv += 1
                        if res == 1: #Es positivo
                            npv += 1

            print("Jovenes:",npj,"/",ncj)
            print("Viejos (+50):",npv,"/",ncv)

print("END SYS CORONA")
```



Estudiantes de Ingeniería UC crean plataforma online para ir en ayuda de pymes afectadas por la pandemia

¿QUÉ ES CCARD?

CCard tiene como misión ayudar a PYMES, creando una red de difusión y facilitando el acceso a efectivo para aquellas que han visto disminuidas sus ventas y/o han tenido que cerrar temporalmente



¿CÓMO FUNCIONA?



Creamos una plataforma mediante la cual los consumidores financien a las empresas en el corto plazo a cambio de una giftcard para utilizar en el negocio a medida que la PYME pueda estar operativa.

<https://www.instagram.com/ccardcl/>

<https://pyme.emol.com/17523/plataforma-social-liquidez-pymes-covid-19/>



The screenshot shows a LinkedIn post from the page 'cpu_uc'. The post features a blue header with the CPU logo and the word 'ATENCIÓN' in large white letters. Below the title, it says 'Voluntari@s, ex-voluntari@s e interesados de CPU'. A text box contains the message: 'Como coordinación CPU hemos comenzado a evaluar la posibilidad de trabajar de forma remota en la reparación de notebooks.' On the right side of the post, there is a sidebar with the page's bio, which includes a message about evaluating remote work for notebook repair, and other standard LinkedIn post details like likes, comments, and shares.

https://www.instagram.com/cpu_uc/



Craich
@ACraich

Siguiendo

¡QUE OS METÁIS EN CASA, JODER!



10:12 - 7 abr. 2020

115 Retweets 371 Me gusta



115



371



FUNCIONES

Instalables



INTERESES

“crear mis propios **videojuegos**”

“ **redes sociales**”

“**inteligencia artificial**”



FUERA DEL TEMARIO



Follow

Big Ben

@big_ben_clock

The first, established November 2009 & entirely unofficial. Could be found at tinyurl.com/ylqbey3, but still apparently imitated everywhere.

 Up a tower.  Joined October 2009

0 Following **441K** Followers

Not followed by anyone you're following

Tweets

Tweets & replies

Media

Likes



Big Ben @big_ben_clock · 4m

BONG BONG



Big Ben @big_ben_clock · 1h

BONG BONG BONG BONG BONG BONG BONG BONG BONG



Bibliografía & Investigación

Tweepy

An easy-to-use Python library for accessing the Twitter API.



IMDb

All

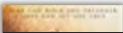
Search IMDb

Bibliografía &
Investigación

IMDb "Top 250" (Sorted by IMDb Rating Descending)

1-50 of 250 titles. | [Next »](#)View Mode: [Compact](#) | [Detailed](#)

Sort by: [Popularity](#) | [A-Z](#) | [User Rating▼](#) | [Number of Votes](#) | [US Box Office](#) | [Runtime](#) | [Year](#)
[Release Date](#) | [Date of Your Rating](#) | [Your Rating](#)

1. [The Shawshank Redemption](#) (1994) +**IMDbPY**

DOWNLOADS

SUPPORT

DEVELOPMENT

ECOSYSTEM

IMDbPY is a Python package for retrieving and managing the data of the IMDb movie database about movies and people.

Votes: 1,523,342 | Gross: \$134.97M



Ingeniería

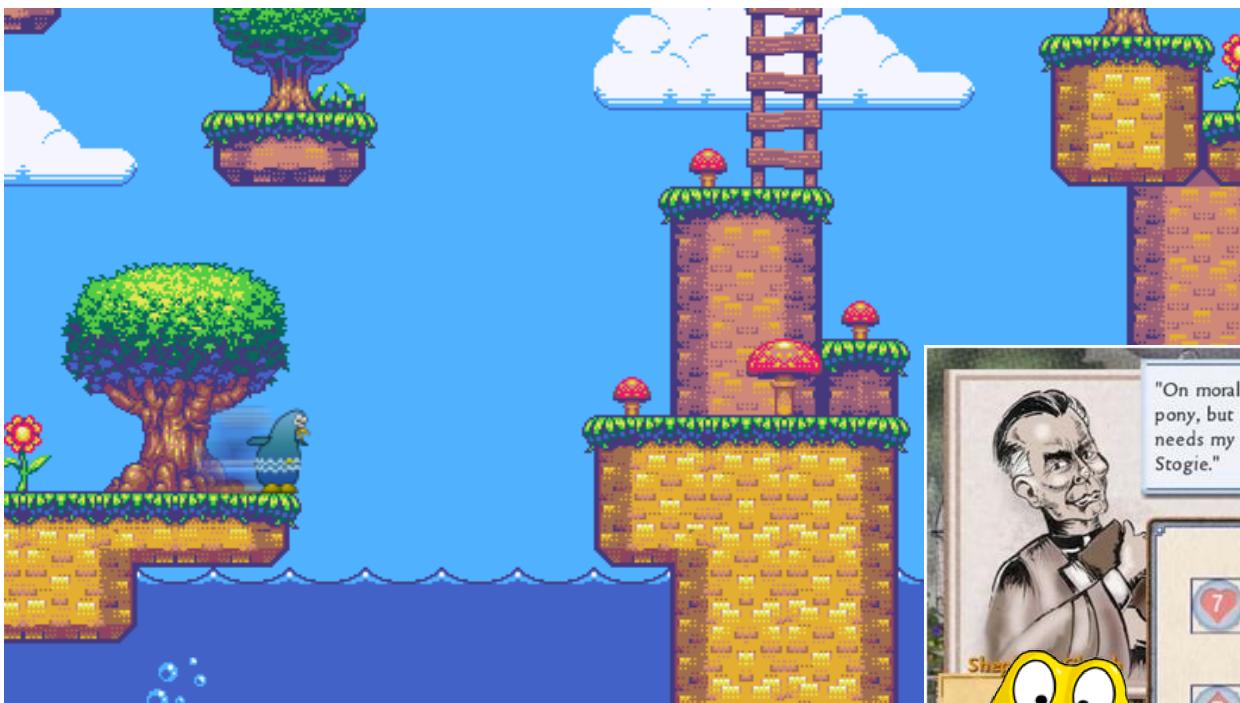
NRC	Sigla	Permite Retiro	¿Se dicta en inglés?	Sec.	¿Requiere Aprob. Especial?	Categoría	Nombre	Profesor	Campus	Créd.	Vacantes			Horario		
											Total	Disponibles	Reservadas			
10868	IIC3103	SI	NO	1	NO		Taller de Integración	Tagle Arturo	San Joaquín	10	80	1		L:5,6 J:4	CLAS N6 AYU K203	
10876	IIC3113	SI	NO	1	NO		Gestión de Proyectos de Tecnologías de Información	Droguett Mario	San Joaquín	10	85	4		M-J:1	CLAS CS405	
10881	IIC3143	SI	NO	1	NO		Desarrollo de Software	Sandoval Rodrigo	San Joaquín	10	76	0		L-W:1	CLAS BC21	
18724	IIC3182	SI	NO	1	NO		Interfaces Humano Computador	Herskovic Valeria	San Joaquín	10	30	12		J:4,5	CLAS H2	
15162	IIC3242	SI	Si	1	NO											
10892	IIC3253	SI	NO	1	NO											
21977	IIC3272	SI	NO	1	NO											
22968	IIC3413	SI	NO	1	NO											
23685	IIC3585	SI	NO	1	NO											
18871	IIC3697	SI	NO	1	NO											
10923	IIC3703	SI	NO	1	NO											
10930	IIC3724	SI	NO	1	NO											
10947	IIC3757	SI	NO	1	NO											



An open source and collaborative framework
for extracting the data you need from websites.
In a fast, simple, yet extensible way.



Bibliografía &
Investigación



Pygame

"On moral grounds I would favor the pony, but half the chantry's flock needs my help, because of Mayor Stogie."

Drag your girl's tokens to separate styles: BRAZEN, SMOOTH, or DEVIOUS. Then press GO! Click the 'Y' button for details.

GO!

BRAZEN
Score your talent and block opponent's Smooth.

SMOOTH
Score your talent and block Devious.

DEVIOUS
Steal opponent's brazen score.

Alma

SCORE TALLY

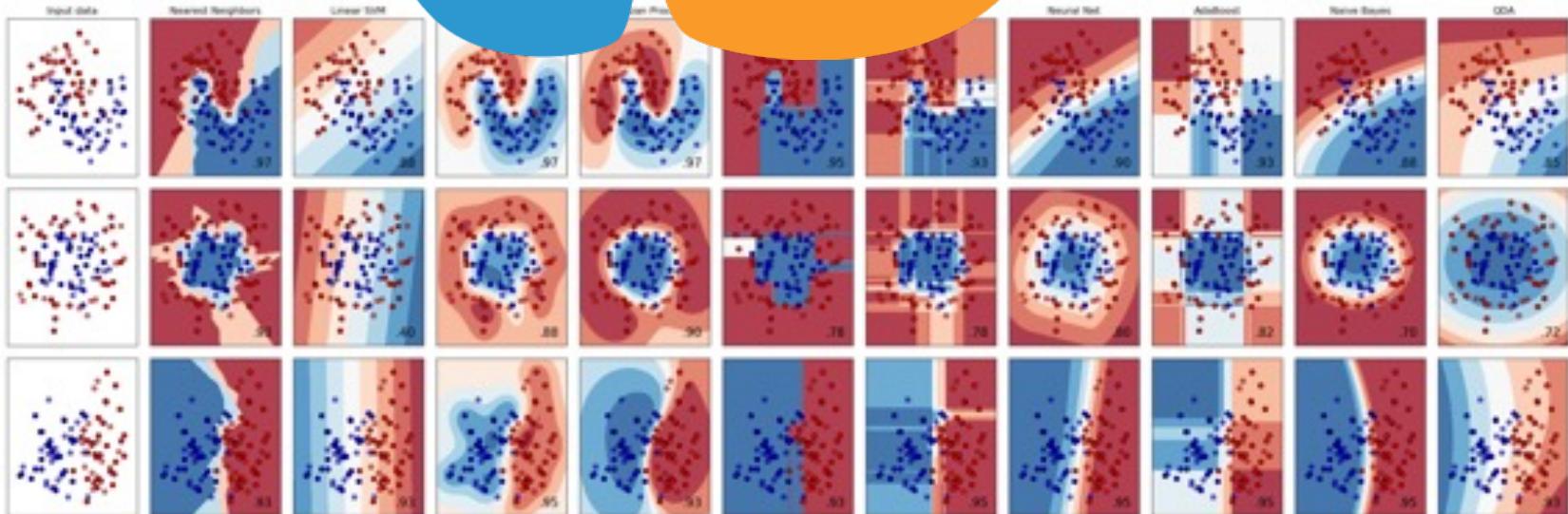
This stone temple was months of spiritual isolation, to swap epiphanies. It predates the town.



<https://realpython.com/pygame-a-primer/>
(Tutorial simple paso a paso)



scikit learn





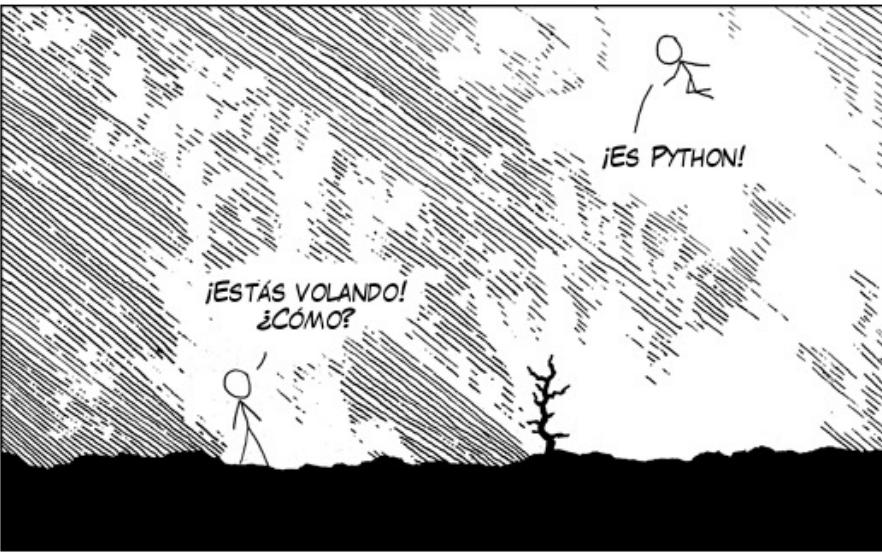
Bibliografía &
Investigación



ANACONDA®

Sistema de gestión de librerías de Data
Science, Matemáticas, Machine Learning, ...

www.anaconda.com



¡LO APRENDÍ ANOCHE!
¡TODO ES TAN
SENCILLO!

EL HOLA MUNDO ES
SIMPLEMENTE
print "¡Hola mundo!"

NO SÉ...
¿TIPADO
DINÁMICO?
¿INDENTACIÓN?
JÚNETE A
NOSOTROS!
¡PROGRAMAR
VUELVE A SER
DIVERTIDO!
¡ES UN MUNDO
NUEVO!
PERO...
¿CÓMO ESTÁS
VOLANDO?

SIMPLEMENTE ESCRIBÍ
import antigravity
¿Y YA ESTÁ?
... TAMBién PROBÉ
TODO LO QUE HAY
EN EL BOTIQUÍN,
PARA COMPARAR.
PERO CREO
QUE ESTO ES
POR PYTHON.

FUNCIONES PROPIAS

Motivación

¿Usar funciones de otros lo entiendo, pero por qué debería definir **funciones propias** en mi propio código?



INTERESES

“emprendimientos”

“ transportes”



Embárcate en un viaje
de exploración y redes
para acelerar tu
emprendimiento en
los ecosistemas más
vibrantes del mundo.

POSTULA CON TU
PROYECTO HASTA EL
11 de septiembre

www.ing.puc.cl/innovacion-y-emprendimiento

BRIDGE
explorer

THE CLOVER

360° ENGINEERING STRATEGY

Projecto apoyado por
CORFO
QUIMICOS INTELIGENTES

Proyecto apoyado por
MECESUP



>>>

IDA Horas: 1

IDA Minutos: 1

IDA Segundos: 1

IDA: 3661 segundos

UNI Horas: 0

UNI Minutos: 0

UNI Segundos: 1

UNI: 1 segundos

VUELTA Horas: 0

VUELTA Minutos: 2

VUELTA Segundos: 2

VUELTA: 122 segundos

>>>





```
#IDA
ida_h = int(input("IDA Horas: "))
ida_m = int(input("IDA Minutos: "))
ida_s = int(input("IDA Segundos: "))
ida = ida_h * 3600
ida = ida + (ida_m * 60)
ida = ida + ida_s
print("IDA:", ida_s, "segundos")
```

```
#UNI
uni_h = int(input("UNI Horas: "))
uni_m = int(input("UNI Minutos: "))
uni_s = int(input("UNI Segundos: "))
uni = uni_h * 3600
uni = uni + (uni_m * 60)
uni = uni + ida_s
print("UNI:", ida_s, "segundos")
```

```
#VUELTA
vue_h = int(input("VUELTA Horas: "))
vue_m = int(input("VUELTA Minutos: "))
vue_s = int(input("VUELTA Segundos: "))
vue = vue_h * 3600
vue = vue + (vue_m * 60)
vue = vue + vue_s
print("VUELTA:", vue, "segundos")
```

Encuentra los 7 errores:

- 360 en Vuelta
- UNI en print de Vuelta
- ida_s en print de IDA
- Parentesis en ida_s
- ida_s en último UNI
- ida en print de UNI
- ... solo hay 6 :P



#IDA

```
ida = tiemppear("IDA")
print("IDA:", ida, "segundos")
```

#UNI

```
uni = tiemppear("UNI")
print("UNI:", uni, "segundos")
```

#VUELTA

```
vuelta = tiemppear("VUELTA")
print("VUELTA:", vuelta, "segundos")
```



```
def tiemppear(etapa):
    h = int(input(etapa + " Horas: "))
    m = int(input(etapa + " Minutos: "))
    s = int(input(etapa + " Segundos: "))
    r = h * 3600
    r = r + (m * 60)
    r = r + s
    return r
```

```
#IDA
ida = tiemppear("IDA")
print("IDA:", ida, "segundos")
#UNI
uni = tiemppear("UNI")
print("UNI:", uni, "segundos")
#VUELTA
vuelta = tiemppear("VUELTA")
print("VUELTA:", vuelta, "segundos")
```

SIN funciones

Largo y tedioso

Fácil cometer errores y difícil de encontrarlos

Difícil “leer” y estructurar códigos largos

No se puede reusar

CON funciones

Más corto

Solo hay que revisar el código en un único sitio

Estructurar partes y darles nombres fáciles de entender

Se puede reusar



FUNCIONES PROPIAS

Definición

Definición de la función 'resta' que recibe 2 parámetros.
Se usa 'def'

```
def resta(x,y):  
    res = x - y  
    return res
```

Bloque de la función 'resta'

```
def suma(x,y):  
    res = x + y  
    return res
```

Retorna el resultado de realizar la función 'suma'. Se usa 'return'

#CODIGO PRINCIPAL

```
a = 2  
b = 3  
rs = suma(a,b)  
rr = resta(a,b)  
print("Resultado suma", rs)  
print("Resultado resta", rr)
```

Llama a la función 'suma'

Resultado suma 5
Resultado resta -1

La Regla de la “**Burbuja de Las Vegas**”



1- La única información que la función puede usar del exterior es la que le pasan como **parámetros**

```
def resta(x,y):  
    res = x - y  
    return res  
  
def suma(x,y):  
    res = x + y  
    return res  
  
#CODIGO PRINCIPAL  
a = 2  
b = 3  
rs = suma(a,b)  
rr = resta(a,b)  
print("Resultado suma", rs)  
print("Resultado resta", rr)
```



2- No importa como se llamen fuera, la función le pondrá sus **propios nombres** a los parámetros

```
def resta(x,y):
    res = x - y
    return res

def suma(x,y):
    res = x + y
    return res

#CÓDIGO PRINCIPAL
a = 2
b = 3
rs = suma(a,b)
rr = resta(a,b)
print("Resultado suma", rs)
print("Resultado resta", rr)
```



3- La función sabe cual parámetro es cual por el orden

```
def resta(x,y):
    res = x - y
    return res

def suma(x,y):
    res = x + y
    return res

#CÓDIGO PRINCIPAL
a = 2
b = 3
rs = suma(a,b)
rr = resta(a,b)
print("Resultado suma", rs)
print("Resultado resta", rr)
```



4- La función solo comunica cosas al exterior con lo que **retorne**

```
def resta(x,y):
    res = x - y
    return res

def suma(x,y):
    res = x + y
    return res

#CÓDIGO PRINCIPAL
a = 2
b = 3
rs = suma(a,b)
rr = resta(a,b)
print("Resultado suma", rs)
print("Resultado resta", rr)
```



5- No importa como se llame el valor a retornar dentro de la función, el exterior lo **guardará en la variable** que el quiera

```
def resta(x,y):
    res = x - y
    return res

def suma(x,y):
    res = x + y
    return res

#CODIGO PRINCIPAL
a = 2
b = 3
rs = suma(a,b)
rr = resta(a,b)
print("Resultado suma", rs)
print("Resultado resta", rr)
```



Retorno

6- Todo lo que **pasa dentro se queda dentro**: nadie fuera puede usar variables creadas dentro de la función)

```
def resta(x,y):
    res = x - y
    return res

def suma(x,y):
    res = x + y
    return res

#CODIGO PRINCIPAL
a = 2
b = 3
rs = suma(a,b)
rr = resta(a,b)
print("Resultado suma", rs)
print("Resultado resta", rr)
```



FUNCIONES PROPIAS

¿Entendí bien como funcionan las funciones?



Situación
Real

FINAL SPACE



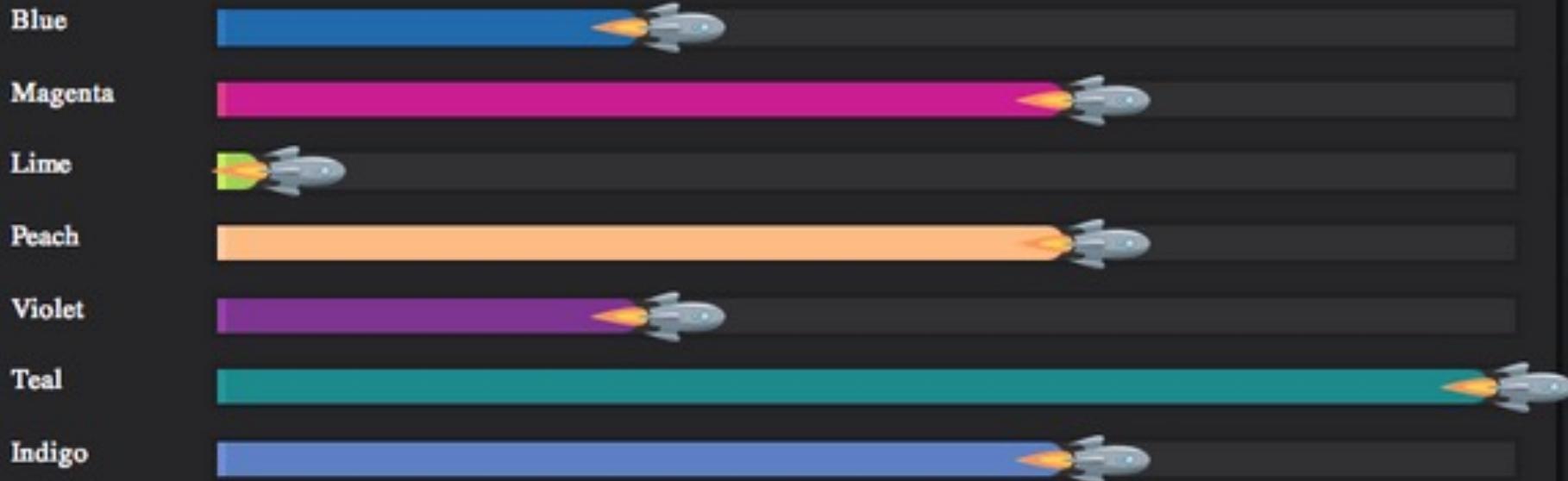
ROOM: socrative1
Space Race: State Facts

FINIS

Situación
Real

Dashboard

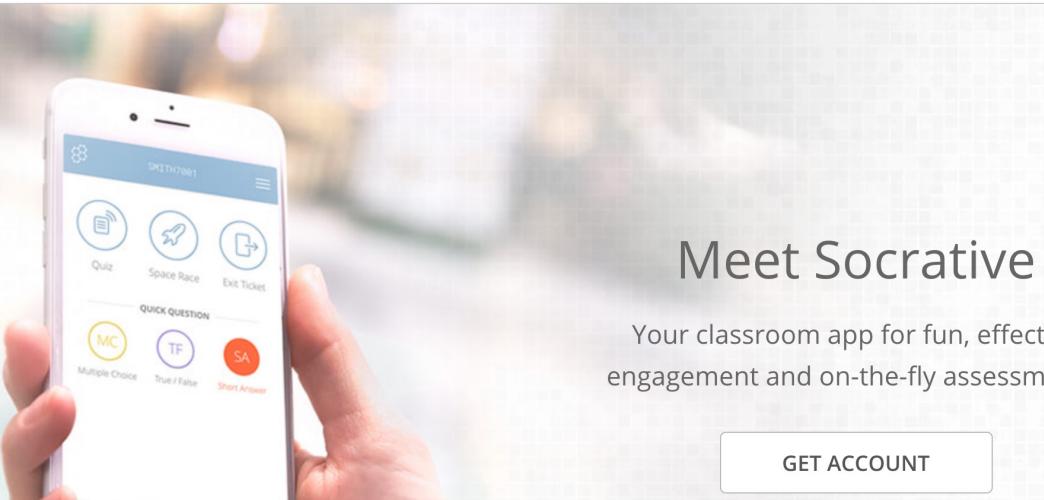
Space Race in Progress...





Plans Apps Get Help Blog

STUDENT LOGIN



JORGEINTRO

Meet Socrative

Your classroom app for fun, effect engagement and on-the-fly assessm

GET ACCOUNT

socrative.com
Student Login
JORGEINTRO





'a', 'b', y 'r' se
llaman igual dentro
y fuera

Oh, que
casualidad, ... pero
que más da!

```
#IN OUT IGUAL
def restar(a,b):
    r = a - b
    return r

a = 10
b = 2
r = restar(a,b)
print(r)
```

```
>>>
8
>>>
```



‘r’ se llaman igual dentro y fuera, pero ‘a,b’ y ‘x,y’ no.

Oh, que casualidad, ... pero que más da!

```
#IN DIF OUT IGUAL
def restar(a,b):
    r = a - b
    return r
```

```
x = 10
y = 2
r = restar(x,y)
print(r)
```

```
>>>
8
>>>
```



'a,b' se llaman igual dentro y fuera, pero 'r' y 'res' no.

Oh, que casualidad, ... pero que más da!

```
#OUT DIF IN IGUAL
def restar(a,b):
    r = a - b
    return r
```

```
a = 10
b = 2
res = restar(a,b)
print(res)
```

```
>>>
8
>>>
```



Estas intentando usar 'a,b' en fuera de la función

¿!Pero no te he dicho, que lo que pasa en la función se queda en la función?!

```
#SCOPE IN EN PRINCIPAL
def restar(a,b):
    r = a - b
    return r

x = 10
y = 2
r = restar(x,y)
print(a,"menos",b,"es",r)
```

```
NameError: name 'a' is not defined
>>>
```



Estas intentando
usar 'r' en fuera de
la función

**¿!Pero no te he
dicho, que lo que
pasa en la función
se queda en la
función?!**

```
#SCOPE OUT
def restar(a,b):
    r = a - b
    return r

a = 10
b = 2
res = restar(a,b)
print(r)
```

```
NameError: name 'r' is not defined
>>>
```



Lo llamas con *b* y *a*,
pero dentro se
llaman *a* y *b*

*No le importa los
nombres, solo el
ORDEN*

```
#IN EN DESORDEN
def restar(a,b):
    r = a - b
    return r
```

```
a = 10
b = 2
r = restar(b,a)
print(r)
```

```
>>>
-8
>>>
```



Aunque parezca que la función esta pensada para sumar dos números, le llegan 2 str como parámetros

En Python nadie fuerza a que algo sea de un tipo concreto. Tu mismo has creado la función como has querido, simplemente llámala como pensaste

```
#SUMA STR
def sumar(num1,num2):
    r = num1 + num2
    return r

num1 = "6"
num2 = "7"
r = sumar(num1,num2)
print(r)
```

```
>>>
67
>>>
```

OK BOOMER



INTERESES

“... creación de discursos y **diálogos**.”



FG

Oh sí. Ayer fui al registro civil. Tiene RUN 27 millones.

Mayor que el tuyo

Jajaja



estoy mas cerca de tu hijo que de ti :DDD

yo soy 24

Tu tienes 16?

FG

El mío es 13 millones. La mama es 15



Nombre Personaje 1? Alice

RUT de Alice? 11111111

Nombre Personaje 2? Bob

RUT de Bob? 22222222

- Maldito milenial! - dijo Alice gritando.

- Lo que tu digas boomer - respondio Bob irrespetuosamente.

Nombre Personaje 1? Charlie

RUT de Charlie? 27000111

Nombre Personaje 2? David

RUT de David? 9000111

- Maldito milenial! - dijo David gritando.

- Lo que tu digas boomer - respondio Charlie irrespetuosamente.



¿ ?

```
n1 = input("Nombre Personaje 1? ")
r1 = int(input("RUT de "+n1+"? "))

n2 = input("Nombre Personaje 2? ")
r2 = int(input("RUT de "+n2+"? "))

d1 = dosdig(r1)
d2 = dosdig(r2)

if d1 < d2:
    v = n1
    j = n2
else:
    v = n2
    j = n1

print("- Maldito milenial! - dijo",v,"gritando.")
print("- Lo que tu digas boomer - respondio",j,"irrespetuosamente.")
```





I'LL WAIT
FOR YOU HERE



¿ ?

```
n1 = input("Nombre Personaje 1? ")
r1 = int(input("RUT de "+n1+"? "))

n2 = input("Nombre Personaje 2? ")
r2 = int(input("RUT de "+n2+"? "))

d1 = dosdig(r1)
d2 = dosdig(r2)

if d1 < d2:
    v = n1
    j = n2
else:
    v = n2
    j = n1

print("- Maldito milenial! - dijo",v,"gritando.")
print("- Lo que tu digas boomer - respondio",j,"irrespetuosamente.")
```



```
def dosdig(rut):
    d = rut // 1000000
    return d

n1 = input("Nombre Personaje 1? ")
r1 = int(input("RUT de "+n1+"? "))

n2 = input("Nombre Personaje 2? ")
r2 = int(input("RUT de "+n2+"? "))

d1 = dosdig(r1)
d2 = dosdig(r2)

if d1 < d2:
    v = n1
    j = n2
else:
    v = n2
    j = n1

print("- Maldito milenial! - dijo",v,"gritando.")
print("- Lo que tu digas boomer - respondio",j,"irrespetuosamente.")
```

2 CABRAS Y 1 FERRARI



INTERESES

“probabilidades en juegos, tales como, black jack 21”

“juegos pequeños de probabilidad o sorteos”

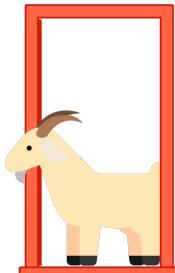
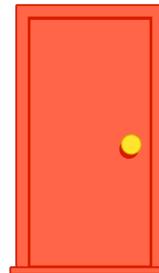
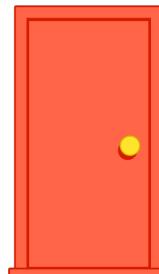


Estás en un concurso de la tele. Hay 3 puertas **A**, **B**, y **C**. Detrás de una hay el **Ferrari** que siempre quisiste. En las otras dos hay una **cabra**. Odiás las cabras.

Eliges la puerta **A**. El presentador abre la puerta **C** que descartaste y te muestra una cabra (**siempre abrirá una puerta con cabra** en el primer descarte).

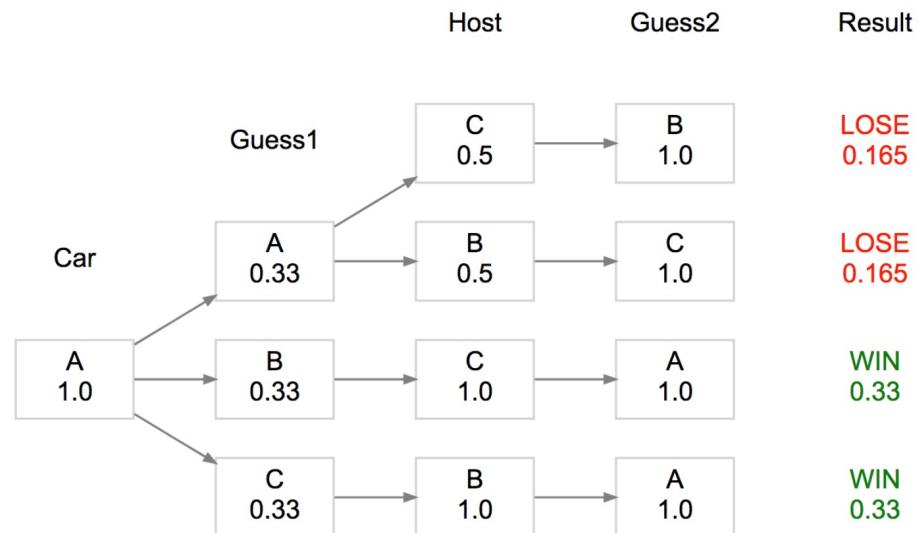
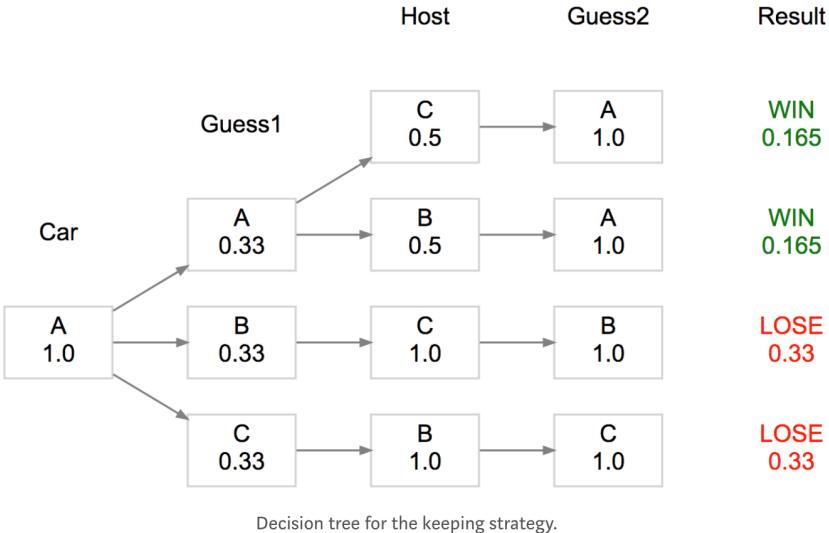
Ahora te da a **elegir**: ¿Quieres **seguir** con la puerta A o la **cambias** por la B?

¿Qué harías?





Situación
Real





Vos Savant wrote in her first column on the Monty Hall problem that the player should switch ([vos Savant 1990a](#)). She received thousands of letters from her readers—the vast majority of which, including many from readers with PhDs, disagreed with her answer.



```
>>>  
Victorias SIN CAMBIAR: 33452 / 100000  
Victorias CAMBIANDO: 66440 / 100000  
>>>
```



```
import random

'''Funcion que juega una partida.
Parametro: un bool que dice cambiara de puerta.
Retorno: un bool diciendo si gana.'''
def jugar(cambia):
    #Colocar el auto
    auto = random.randint(0,2)

    #Colocar las cabras en las otras puertas
    if auto == 0:
        cabral = 1
        cabra2 = 2
    elif auto == 1:
        cabral = 0
        cabra2 = 2
    else:
        cabral = 0
        cabra2 = 1

    #El participante elige
    eleccion = random.randint(0,2)

    #Calcular cual es la puerta alternativa que queda despues
    #de abrir una cabra
    if (eleccion == cabral) or (eleccion == cabra2):
        #Abres la otra cabra, y por tanto la alternativa es la puerta del auto
        alternativa = auto
    else:
        #Si elije el auto, abrimos la cabral por ejemplo, y solo queda cabra2
        alternativa = cabra2

    #Cambia su eleccion o no dependiendo del parametro
    if cambia:
        eleccion = alternativa

    #Miramos si ha ganado
    ganado = (eleccion == auto)

    #Devolvemos si ha ganado o no
    return ganado

#PROGRAMA PRINCIPAL
num_partidas = 100000

#Jugar sin cambiar
vsincambiar = 0
for i in range(0,num_partidas):
    ganado = jugar(False)
    if ganado:
        vsincambiar += 1

#Jugar cambiando
vcambiar = 0
for i in range(0, num_partidas):
    ganado = jugar(True)
    if ganado:
        vcambiar += 1

#Imprimir resultados
print("Victorias SIN CAMBIAR:",vsincambiar,"/",num_partidas)
print("Victorias CAMBIANDO:",vcambiar,"/", num_partidas)
```



```
import random

'''Funcion que juega una partida.
Parametro: un bool que dice cambiara de puerta.
Retorno: un bool diciendo si gana.'''
def jugar(cambia):
```

¿ ?

```
#PROGRAMA PRINCIPAL
num_partidas = 100000

#Jugar sin cambiar
vsincambiar = 0
for i in range(0,num_partidas):
    ganado = jugar(False)
    if ganado:
        vsincambiar += 1

#Jugar cambiando
vcambiar = 0
for i in range(0, num_partidas):
    ganado = jugar(True)
    if ganado:
        vcambiar += 1

#Imprimir resultados
print("Victorias SIN CAMBIAR:",vsincambiar,"/",num_partidas)
print("Victorias CAMBIANDO:",vcambiar,"/", num_partidas)
```





I'LL WAIT
FOR YOU HERE



```
import random

'''Funcion que juega una partida.
Parametro: un bool que dice cambiara de puerta.
Retorno: un bool diciendo si gana.'''
def jugar(cambia):
```

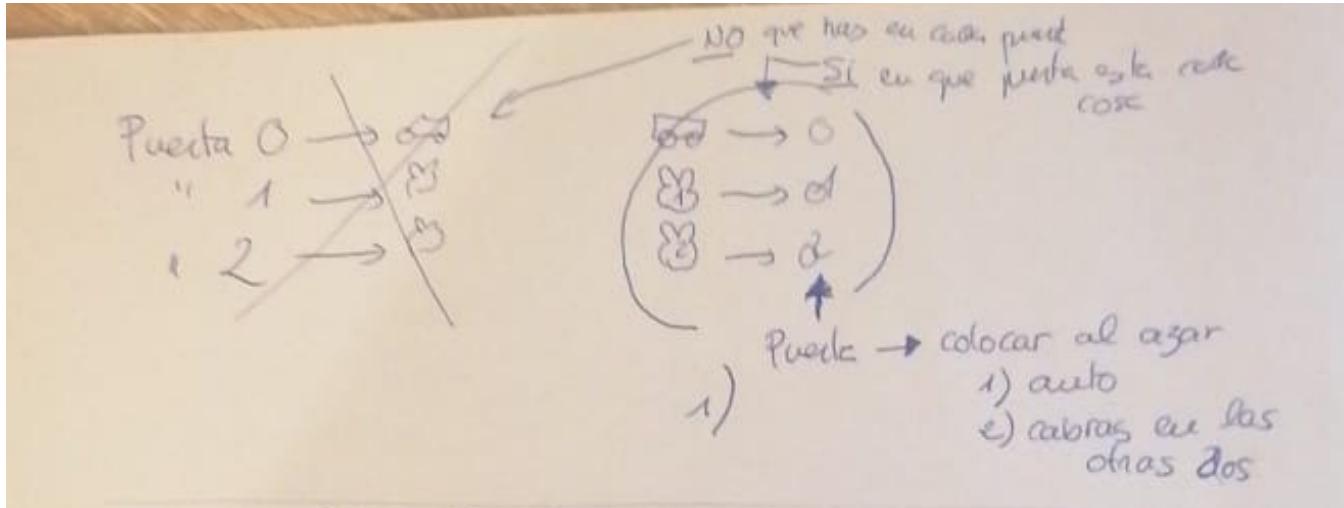
¿ ?

```
#PROGRAMA PRINCIPAL
num_partidas = 100000

#Jugar sin cambiar
vsincambiar = 0
for i in range(0,num_partidas):
    ganado = jugar(False)
    if ganado:
        vsincambiar += 1

#Jugar cambiando
vcambiar = 0
for i in range(0, num_partidas):
    ganado = jugar(True)
    if ganado:
        vcambiar += 1

#Imprimir resultados
print("Victorias SIN CAMBIAR:",vsincambiar,"/",num_partidas)
print("Victorias CAMBIANDO:",vcambiar,"/", num_partidas)
```





```
import random

'''Funcion que juega una partida.
Parametro: un bool que dice cambiara de puerta.
Retorno: un bool diciendo si gana.'''
def jugar(cambia):
    #Colocar el auto
    auto = random.randint(0,2)

    #Colocar las cabras en las otras puertas
    if auto == 0:
        cabra1 = 1
        cabra2 = 2
    elif auto == 1:
        cabra1 = 0
        cabra2 = 2
    else:
        cabra1 = 0
        cabra2 = 1
```

... .



```
import random

'''Funcion que juega una partida.
Parametro: un bool que dice cambiara de puerta.
Retorno: un bool diciendo si gana.'''
def jugar(cambia):
    #Colocar el auto
    auto = random.randint(0,2)

    #Colocar las cabras en las otras puertas
    if auto == 0:
        cabra1 = 1
        cabra2 = 2
    elif auto == 1:
        cabra1 = 0
        cabra2 = 2
    else:
        cabra1 = 0
        cabra2 = 1

    #El participante elige
    eleccion = random.randint(0,2)

    #Calcular cual es la puerta alternativa que queda despues
    #de abrir una cabra
    if (eleccion == cabra1) or (eleccion == cabra2):
        #Abres la otra cabra, y por tanto la alternativa es la puerta del auto
        alternativa = auto
    else:
        #Si elije el auto, abrimos la cabral por ejemplo, y solo queda cabra2
        alternativa = cabra2
```

... .



```
import random

'''Funcion que juega una partida.
Parametro: un bool que dice cambiara de puerta.
Retorno: un bool diciendo si gana.'''
def jugar(cambia):
    #Colocar el auto
    auto = random.randint(0,2)

    #Colocar las cabras en las otras puertas
    if auto == 0:
        cabra1 = 1
        cabra2 = 2
    elif auto == 1:
        cabra1 = 0
        cabra2 = 2
    else:
        cabra1 = 0
        cabra2 = 1

    #El participante elige
    eleccion = random.randint(0,2)

    #Calcular cual es la puerta alternativa que queda despues
    #de abrir una cabra
    if (eleccion == cabra1) or (eleccion == cabra2):
        #Abres la otra cabra, y por tanto la alternativa es la puerta del auto
        alternativa = auto
    else:
        #Si elije el auto, abrimos la cabral por ejemplo, y solo queda cabra2
        alternativa = cabra2

    #Cambia su eleccion o no dependiendo del parametro
    if cambia:
        eleccion = alternativa

    #Miramos si ha ganado
    ganado = (eleccion == auto)

    #Devolvemos si ha ganado o no
    return ganado
```

BECHDEL TEST



¿Pueden funciones llamar a otras funciones?

Sí, claro.

Que bueno que me lo preguntes, pues justo tengo un ejemplo aquí ...



INTERESES

“series y **películas**”

“cosas **curiosas** e interesantes”

“**igualdad**”

“**comics**, super-heroes”



Bechdel Test

“Es una medida de la representación de la mujer en la ficción (película, comic, etc)”



A character in *Dykes to Watch Out For* explains the rules that later came to be known as the Bechdel test (1985). □

1. Aparecen al menos **dos** mujeres [*con nombre]
2. **Hablan** entre si
3. Hablan de otra cosa que no sea un **hombre**



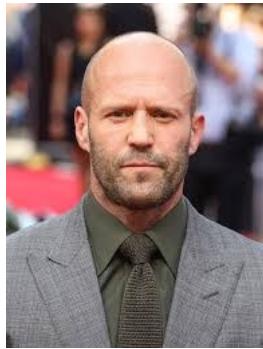
- La trilogía original de *Star Wars*
- *The Dark Knight*
- *Cuando Harry conoció a Sally*
- *Terminator Salvation*
- *Avatar*
- *Trainspotting*
- *Misión: Imposible*
- *Toy Story*
- *Gladiador*
- *Back to the Future*

- *Lara Croft: Tomb Raider*
- *X-Men*
- La saga de *Austin Powers*
- *District 9*
- *Los cazafantasmas*
- *The Bourne Identity*
- *El gran Lebowski*
- *El quinto elemento*
- *Reservoir Dogs*
- *The Truman Show*

- *Up*
- Las películas de James Bond
- La saga de *Piratas del Caribe*
- La serie *Men in Black*
- *Top Gun*
- *Breakfast at Tiffany's*
- *Stalker*, de Andréi Tarkovski
- La saga de *El Señor de los Anillos*
- 10 de 15 películas de Pixar Animation Studios (excepto *A Bug's Life*, *Toy Story 2*, *Los Increíbles*, *Brave* e *Inside Out*)



Situación
Real





Pelicula? Harry Potter 7 Part I

En la pelicula Harry Potter 7 Part I ...

... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1

... hablan entre si? (1-Si 2-No) 1

... hablan de otra cosa que no sea un hombre? (1-Si 2-No) 1

Harry Potter 7 Part I pasa el Bechdel Test

Pelicula? Harry Potter 7 Part II

En la pelicula Harry Potter 7 Part II ...

... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1

... hablan entre si? (1-Si 2-No) 2

Harry Potter 7 Part II NO pasa el Bechdel Test



```
def bechdel(m):
```

¿ ?

```
def regla1():
    op = int(input("... aparecen al menos dos mujeres con nombre? (1-Si 2-No) "))
    res = op == 1
    return res

def regla2():
    op = int(input("... hablan entre si? (1-Si 2-No) "))
    res = op == 1
    return res

def regla3():
    op = int(input("... hablan de otra cosa que no sea un hombre? (1-Si 2-No) "))
    res = op == 1
    return res

# CODIGO PRINCIPAL
p = input("Película? ")
r = bechdel(p)
if r:
    print(p, "pasa el Bechdel Test")
else:
    print(p, "NO pasa el Bechdel Test")
```

```
Película? Harry Potter 7 Part I
En la película Harry Potter 7 Part I ...
... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1
... hablan entre si? (1-Si 2-No) 1
... hablan de otra cosa que no sea un hombre? (1-Si 2-No) 1
Harry Potter 7 Part I pasa el Bechdel Test
```

```
Película? Harry Potter 7 Part II
En la película Harry Potter 7 Part II ...
... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1
... hablan entre si? (1-Si 2-No) 2
Harry Potter 7 Part II NO pasa el Bechdel Test
```





I'LL WAIT
FOR YOU HERE



```
def bechdel(m):
```

¿ ?

```
def regla1():
    op = int(input("... aparecen al menos dos mujeres con nombre? (1-Si 2-No) "))
    res = op == 1
    return res

def regla2():
    op = int(input("... hablan entre si? (1-Si 2-No) "))
    res = op == 1
    return res

def regla3():
    op = int(input("... hablan de otra cosa que no sea un hombre? (1-Si 2-No) "))
    res = op == 1
    return res

# CODIGO PRINCIPAL
p = input("Película? ")
r = bechdel(p)
if r:
    print(p, "pasa el Bechdel Test")
else:
    print(p, "NO pasa el Bechdel Test")
```

```
Película? Harry Potter 7 Part I
En la película Harry Potter 7 Part I ...
... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1
... hablan entre si? (1-Si 2-No) 1
... hablan de otra cosa que no sea un hombre? (1-Si 2-No) 1
Harry Potter 7 Part I pasa el Bechdel Test
```

```
Película? Harry Potter 7 Part II
En la película Harry Potter 7 Part II ...
... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1
... hablan entre si? (1-Si 2-No) 2
Harry Potter 7 Part II NO pasa el Bechdel Test
```



```
def bechdel(m):
    print("En la pelicula",m," ...")
    pasa = regla1()
    if pasa:
        pasa = regla2()
    if pasa:
        pasa = regla3()
    return pasa

def regla1():
    op = int(input("... aparecen al menos dos mujeres con nombre? (1-Si 2-No) "))
    res = op == 1
    return res

def regla2():
    op = int(input("... hablan entre si? (1-Si 2-No) "))
    res = op == 1
    return res

def regla3():
    op = int(input("... hablan de otra cosa que no sea un hombre? (1-Si 2-No) "))
    res = op == 1
    return res

# CODIGO PRINCIPAL
p = input("Pelicula? ")
r = bechdel(p)
if r:
    print(p,"pasa el Bechdel Test")
else:
    print(p,"NO pasa el Bechdel Test")
```

```
Pelicula? Harry Potter 7 Part I
En la pelicula Harry Potter 7 Part I ...
... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1
... hablan entre si? (1-Si 2-No) 1
... hablan de otra cosa que no sea un hombre? (1-Si 2-No) 1
Harry Potter 7 Part I pasa el Bechdel Test
```

```
Pelicula? Harry Potter 7 Part II
En la pelicula Harry Potter 7 Part II ...
... aparecen al menos dos mujeres con nombre? (1-Si 2-No) 1
... hablan entre si? (1-Si 2-No) 2
Harry Potter 7 Part II NO pasa el Bechdel Test
```



- Bechdel Test
 - https://en.wikipedia.org/wiki/Bechdel_test
- Bechdel Test Movie List
 - <https://bechdeltest.com/>
- Jason Statham: Saltador de Trampolin
 - <https://www.youtube.com/watch?v=vWrINMm1aCl>

PRESUPUESTO PERSONAL

I1 2018-2



INTERESES

“tipo prueba”

(Mis problemas son tipo prueba y menos fomes)



Luego de las celebraciones de las fiestas patrias, notas que gastaste más dinero que el que tenías presupuestado gastar. Para que esto no te vuelva a ocurrir, decides partir registrando todos tus gastos con el fin de llevar una contabilidad y tener estadísticas de ellos. Por cada compra que hagas guardarás un registro de ella, el cual contiene una categoría (lo que compraste), el monto que gastaste, y el mes en que lo compraste. Como ya conoces las maravillas de Python, decides programar algunas funciones que te ayuden en esta tarea.

Para esto, puedes **importar** la librería **cuentas** que tiene las siguientes funciones **ya implementadas**:

- **obtener_cantidad(categoría)**: recibe un **string** con el nombre de la categoría y retorna un **int** con la cantidad de registros existentes para esa categoría. Puedes suponer que una categoría siempre tendrá al menos un registro. Por ejemplo, si **obtener_cantidad("bebida")** retorna 10, significa que hubo 10 registros de compras de bebidas.
- **obtener_mes(categoría, i)**: recibe un **string** con el nombre de la categoría y un **int** que representa el índice del registro (los cuales van desde 0 hasta la cantidad de registros existentes para esa categoría - 1). Retorna un **int** con el número del mes correspondiente al registro y categoría entregada. Por ejemplo, si **obtener_mes("bebida", 0)** entrega 9, significa que la primera bebida (índice 0) fue comprada en el mes de septiembre (9).
- **obtener_monto(categoría, i)**: recibe un **string** con el nombre de la categoría y un **int** que representa el índice del registro (los cuales van desde 0 hasta la cantidad de registros existentes para esa categoría - 1). Retorna un **int** con el monto correspondiente al registro y categoría entregada. Por ejemplo, si **obtener_monto("bebida", 1)** entrega 15, significa que la segunda bebida (índice 1) comprada costó 15.

Con el objetivo de generar estadísticas de tus gastos, **debes definir las siguientes funciones**:

- a) (20 pts) **promedio_categoria(categoría)**: recibe un **string** con el nombre de la categoría y retorna un **float** con el promedio de gasto de los registros de esa categoría.
- b) (20 pts) **total_mensual_categoria(mes, categoría)**: recibe un **int** que puede ir desde el 1 al 12 y representa el mes (1 equivale a enero y 12 a diciembre) y un **string** con el nombre de la categoría. Retorna un **int** con la cantidad total de dinero gastado en la categoría y mes dados.
- c) (20 pts) **mensual_promedio_categoria(categoría)**: recibe un **string** con el nombre de la categoría y retorna un **float** con el gasto mensual promedio para la categoría dada.

Los gastos ya vienen cargados en la librería **cuentas**. Tu misión es programar las funciones mencionadas para que puedan ser utilizadas desde el siguiente código:

Luego de las celebraciones de las fiestas patrias, notas que gastaste más dinero que el que tenías presupuestado gastar. Para que esto no te vuelva a ocurrir, decides partir registrando todos tus gastos con el fin de llevar una contabilidad y tener estadísticas de ellos. Por cada compra que hagas guardarás un registro de ella, el cual contiene una categoría (lo que compraste), el monto que gastaste, y el mes en que lo compraste. Como ya conoces las maravillas de Python, decides programar algunas funciones que te ayuden en esta tarea.

Para esto, puedes **importar** la librería **cuentas** que tiene las siguientes funciones **ya implementadas**:

- **obtener_cantidad(categoría)**: recibe un **string** con el nombre de la categoría y retorna un **int** con la cantidad de registros existentes para esa categoría. Puedes suponer que una categoría siempre tendrá al menos un registro. Por ejemplo, si **obtener_cantidad("bebida")** retorna 10, significa que hubo 10 registros de compras de bebidas.
- **obtener_mes(categoría, i)**: recibe un **string** con el nombre de la categoría y un **int** que representa el índice del registro (los cuales van desde 0 hasta la cantidad de registros existentes para esa categoría - 1). Retorna un **int** con el número del mes correspondiente al registro y categoría entregada. Por ejemplo, si **obtener_mes("bebida", 0)** entrega 9, significa que la primera bebida (índice 0) fue comprada en el mes de septiembre (9).
- **obtener_monto(categoría, i)**: recibe un **string** con el nombre de la categoría y un **int** que representa el índice del registro (los cuales van desde 0 hasta la cantidad de registros existentes para esa categoría - 1). Retorna un **int** con el monto correspondiente al registro y categoría entregada. Por ejemplo, si **obtener_monto("bebida", 1)** entrega 15, significa que la segunda bebida (índice 1) comprada costó 15.



Con el objetivo de generar estadísticas de tus gastos, **debes definir las siguientes funciones:**

- a) (20 pts) `promedio_categoria(categoría)`: recibe un `string` con el nombre de la categoría y retorna un `float` con el promedio de gasto de los registros de esa categoría.
- b) (20 pts) `total_mensual_categoria(mes, categoría)`: recibe un `int` que puede ir desde el 1 al 12 y representa el mes (1 equivale a enero y 12 a diciembre) y un `string` con el nombre de la categoría. Retorna un `int` con la cantidad total de dinero gastado en la categoría y mes dados.
- c) (20 pts) `mensual_promedio_categoria(categoría)`: recibe un `string` con el nombre de la categoría y retorna un `float` con el gasto mensual promedio para la categoría dada.

Los gastos ya vienen cargados en la librería `cuentas`. Tu misión es programar las funciones mencionadas para que puedan ser utilizadas desde el siguiente código:



```
terminar = False
while not terminar:

    opcion = input(''': ¿Qué deseas hacer?
                    1) Obtener promedio de gastos de una categoría.
                    2) Obtener total de gastos de una categoría para un mes dado.
                    3) Obtener el gasto mensual promedio de una categoría.
                    4) Salir.''')

    if opcion == 1:
        categoria = input('Ingrese categoría: ')
        print(promedio_categoria(categoría))
    elif opcion == 2:
        categoria = input('Ingrese categoría: ')
        mes = int(input('Ingrese mes: '))
        print(total_mensual_categoria(mes, categoria))
    elif opcion == 3:
        categoria = input('Ingrese categoría: ')
        print(mensual_promedio_categoria(categoría))
    elif opcion == 4:
        terminar = True
```



Resolución
Problema

```
import cuentas

def promedio_categoria(categoría):
```

```
def total_mensual_categoria(mes,categoría):
```

```
def mensual_promedio_categoria(categoría):
```





I'LL WAIT
FOR YOU HERE



- a) (20 pts) `promedio_categoria(categoría)`: recibe un `string` con el nombre de la categoría y retorna un `float` con el promedio de gasto de los registros de esa categoría.

```
def promedio_categoria(categoría):
    ncat = cuentas.obtener_cantidad(categoría)
    suma = 0
    for i in range(0,ncat):
        suma += cuentas.obtener_monto(categoría,i)
    pro = suma / ncat
    return pro
```

- b) (20 pts) `total_mensual_categoria(mes, categoria)`: recibe un `int` que puede ir desde el 1 al 12 y representa el mes (1 equivale a enero y 12 a diciembre) y un `string` con el nombre de la categoría. Retorna un `int` con la cantidad total de dinero gastado en la categoría y mes dados.

```
def total_mensual_categoria(mes,categoría):
    ncat = cuentas.obtener_cantidad(categoría)
    suma = 0
    for i in range(0,ncat):
        m = cuentas.obtener_mes(categoría,i)
        if m == mes:
            suma += cuentas.obtener_monto(categoría,i)
    return suma
```



```
def mensual_promedio_categoria(categoría):  
    #Cuantos meses se ha gastado algo
```

```
#Cuanto se ha gastado en total
```

```
#Promedio
```



```
def mensual_promedio_categoria(categoría):
    #Cuantos meses se ha gastado algo
    nmeses = 0
    for m in range(1,13):
        ncat = cuentas.obtener_cantidad(categoría)
        se_gasto = False
        for i in range(0,ncat):
            if cuentas.obtener_mes(categoría,i) == m:
                se_gasto = True
        if se_gasto:
            nmeses += 1

    #Cuanto se ha gastado en total
    ncat = cuentas.obtener_cantidad(categoría)
    suma = 0
    for i in range(0,ncat):
        suma += cuentas.obtener_monto(categoría,i)

    #Promedio
    pro = suma / nmeses
    return pro
```