

## • Módulo 4: Funciones, tuplas, diccionarios, exposiciones y procesamiento de datos

### 1. Laboratorio 1

Tu tarea es escribir y probar una función que toma un argumento (un año) y devuelve True si el año es un año bisiesto, o False si no lo es..

```
1 def is_year_leap(year):
2     if year % 4 != 0:
3         return False
4     elif year % 100 != 0:
5         return True
6     elif year % 400 != 0:
7         return False
8     else:
9         return True
10
11 test_data = [1900, 2000, 2016, 1987]
12 test_results = [False, True, True, False]
13 for i in range(len(test_data)):
14     yr = test_data[i]
15     print(yr, "-> ", end="")
16     result = is_year_leap(yr)
17     if result == test_results[i]:
18         print("OK")
19     else:
20         print("Fallido")
```

Console >\_

```
1900 -> OK
2000 -> OK
2016 -> OK
1987 -> OK
```

### 2. Laboratorio 2

Tu tarea es escribir y probar una función que toma dos argumentos (un año y un mes) y devuelve el número de días del mes/año dado (mientras que solo febrero es sensible al valor year, tu función debería ser universal).

```
1 def is_year_leap(year):
2     if year % 4 != 0:
3         return False
4     elif year % 100 != 0:
5         return True
6     elif year % 400 != 0:
7         return False
8     else:
9         return True
10
11 def days_in_month(year, month):
12     if year < 1582 or month < 1 or month > 12:
13         return None
14     days = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
15     res = days[month - 1]
16     if month == 2 and is_year_leap(year):
17         res = 29
18     return res
19
20 test_years = [1900, 2000, 2016, 1987]
21 test_months = [2, 2, 1, 11]
22 test_results = [28, 29, 31, 30]
23 for i in range(len(test_years)):
24     yr = test_years[i]
25     mo = test_months[i]
26     print(yr, mo, "-> ", end="")
27     result = days_in_month(yr, mo)
28     if result == test_results[i]:
29         print("OK")
30     else:
31         print("Fallido")
```

Console >\_

```
1900 2 -> OK
2000 2 -> OK
2016 1 -> OK
1987 11 -> OK
```

### 3. Laboratorio 3

Tu tarea es escribir y probar una función que toma tres argumentos (un año, un mes y un día del mes) y devuelve el día correspondiente del año, o devuelve None si cualquiera de los argumentos no es válido. Debes utilizar las funciones previamente escritas y probadas. Agrega algunos casos de prueba al código. Esta prueba es solo el comienzo.

```
1 def is_year_leap(year):
2     if year % 4 != 0:
3         return False
4     elif year % 100 != 0:
5         return True
6     elif year % 400 != 0:
7         return False
8     else:
9         return True
10
11 def days_in_month(year, month):
12     if year < 1582 or month < 1 or month > 12:
13         return None
14     days = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
15     res = days[month - 1]
16     if month == 2 and is_year_leap(year):
17         res = 29
18     return res
19
20 def day_of_year(year, month, day):
21     days = 0
22     for m in range(1, month):
23         md = days_in_month(year, m)
24         if md == None:
25             return None
26         days += md
27     md = days_in_month(year, month)
28     if day >= 1 and day <= md:
29         return days + day
30     else:
31         return None
32
33 print(day_of_year(2000, 12, 31))
34
```

Console >\_ 366

### 4. Laboratorio 4

Tu tarea es escribir una función que verifique si un número es primo o no.

La función:

- Se llama is\_prime.
- Toma un argumento (el valor a verificar).
- Devuelve True si el argumento es un número primo, y False de lo contrario.

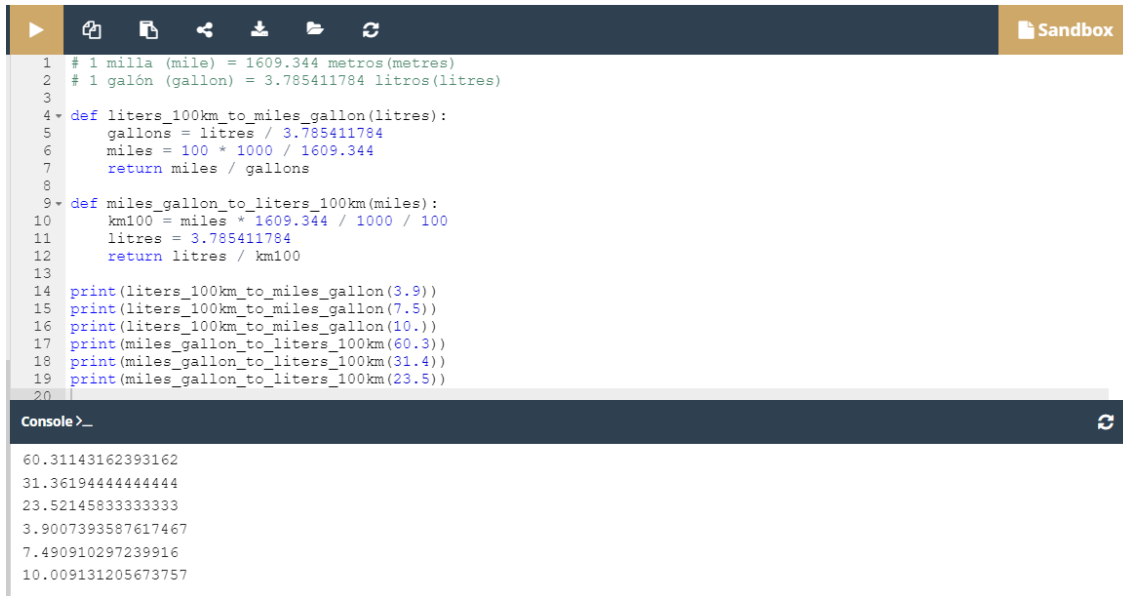
```
1 def is_prime(num):
2     for i in range(2, int(1 + num ** 0.5)):
3         if num % i == 0:
4             return False
5     return True
6
7 for i in range(1, 20):
8     if is_prime(i + 1):
9         print(i + 1, end=" ")
10 print()
11
```

Console >\_ 2 3 5 7 11 13 17 19

## 5. Laboratorio 5

Tu tarea es escribir un par de funciones que conviertan l/100km a mpg (milas por galón), y viceversa. Las funciones:

- Se llaman `liters_100km_to_miles_gallon` y `miles_gallon_to_liters_100km` respectivamente.
- Toman un argumento (el valor correspondiente a sus nombres).



```
1 # 1 milla (mile) = 1609.344 metros(metros)
2 # 1 galón (gallon) = 3.785411784 litros(litres)
3
4 def liters_100km_to_miles_gallon(litres):
5     gallons = litres / 3.785411784
6     miles = 100 * 1000 / 1609.344
7     return miles / gallons
8
9 def miles_gallon_to_liters_100km(miles):
10    km100 = miles * 1609.344 / 1000 / 100
11    litres = 3.785411784
12    return litres / km100
13
14 print(liters_100km_to_miles_gallon(3.9))
15 print(liters_100km_to_miles_gallon(7.5))
16 print(liters_100km_to_miles_gallon(10.))
17 print(miles_gallon_to_liters_100km(60.3))
18 print(miles_gallon_to_liters_100km(31.4))
19 print(miles_gallon_to_liters_100km(23.5))
20
```

Console >\_

```
60.31143162393162
31.361944444444444
23.521458333333333
3.9007393587617467
7.490910297239916
10.009131205673757
```

## 6. Laboratorio 6

Tu tarea es escribir un simple programa que simule jugar a tic-tac-toe (nombre en inglés) con el usuario. Para hacerlo más fácil, hemos decidido simplificar el juego. Aquí están nuestras reglas:

- La máquina (por ejemplo, el programa) jugará utilizando las 'X's.
- El usuario (por ejemplo, tu) jugarás utilizando las 'O's.
- El primer movimiento es de la maquina: siempre coloca una 'X' en el centro del tablero.
- Todos los cuadros están numerados comenzando con el 1 (observa el ejemplo para que tengas una referencia).
- El usuario ingresa su movimiento introduciendo el número de cuadro elegido. El número debe de ser valido, por ejemplo, un valor entero mayor que 0 y menor que 10, y no puede ser un cuadro que ya esté ocupado.
- El programa verifica si el juego ha terminado. Existen cuatro posibles veredictos: el juego continuo, el juego termina en empate, tus ganas, o la maquina gana.
- La máquina responde con su movimiento y se verifica el estado del juego.
- No se debe implementar algún tipo de inteligencia artificial, la maquina elegirá un cuadro de manera aleatoria, eso es suficiente para este juego.

Course: Curso Python Estadías
Edube Sandbox

edube.org/sandbox

Google
Multimedia
Escuela
Investigación Labs
Estadías
Cisco Networking A...

OPENEDG
Run
Share Code
Back to Course
Python 3.7

```

1 from random import randrange
2 def display_board(board):
3     print("+-----+ 3, "+"", sep="")
4     for row in range(3):
5         print("|" + " 3,|" + " ", end="")
6         for col in range(3):
7             print("|" + str(board[row][col]) + " ", end="")
8         print("|" + " 3,|" + " ", end="")
9         print("+-----+ 3, "+"", sep="")
10    print(" ")
11 def enter_move(board):
12    ok = False # suposición falsa - la necesitamos para entrar en el bucle
13    while not ok:
14        move = input("Ingresa tu movimiento: ")
15        ok = len(move) == 1 and move != '1' and move <= '9' # ¿es válido lo que ingresó el usuario?
16        if not ok:
17            print("Movimiento erróneo, ingresalo nuevamente") # no, no lo es. Ingresalo nuevamente
18            continue
19        move = int(move) - 1 # número de la celda, del 0 al 8
20        row = move // 3 # fila de la celda
21        col = move % 3 # columna de la celda
22        sign = board[row][col] # marca el cuadro elegido
23        ok = sign not in ['0', 'X']
24        if not ok: # esta ocupado, ingresa una posición nuevamente
25            print("Cuadro ocupado, ingresa nuevamente!")
26            continue
27        board[row][col] = 'O' # colocar 'O' al cuadro seleccionado
28 def make_list_of_free_fields(board):
29    free = [] # la lista está vacía inicialmente
30    for row in range(3): # itera a través de las filas
31        for col in range(3): # itera a través de las columnas
32            if board[row][col] not in ['0', 'X']: # ¿Está la celda libre?
33                free.append((row,col)) # si, agrega una nueva tupla a la lista
34    return free
35 def victory_for(board,sgn):
36    if sgn == "X": # ¿Estamos buscando X?
37        who = "me" # Si, es la maquina
38    elif sgn == "O": # ¿Estamos buscando O?
39        who = "you" # Si, es el usuario
40    else:
41        who = None # No debemos de caer aquí!
42    cross1 = cross2 = True # para las diagonales
43    for rc in range(3):
44        if board[rc][0] == sgn and board[rc][1] == sgn and board[rc][2] == sgn: # check row rc
45            return who
46        if board[0][rc] == sgn and board[1][rc] == sgn and board[2][rc] == sgn: # check column rc
47            return who
48        if board[rc][rc] != sgn: # revisar la primera diagonal
49            cross1 = False
50        if board[2 - rc][2 - rc] != sgn: # revisar la segunda diagonal
51            cross2 = False
52    if cross1 or cross2:
53        return who
54    return None
55
56 def draw_move(board):
57    free = make_list_of_free_fields(board) # crea una lista de los cuadros vacios o libres
58    cnt = len(free)
59    if cnt > 0: # si la lista no está vacía, elegir un lugar para "X" y colocarla
60        this = randrange(cnt)
61        row, col = free[this]
62        board[row][col] = 'X'
63
64 board = [
65     [3 * j + i + 1 for i in range(3)] for j in range(3) ] # crear un tablero vacío
66 board[1][1] = 'X' # colocar la primera "X" en el centro
67 free = make_list_of_free_fields(board)
68 human_turn = True # ¿De quien es turno ahora?
69 while len(free):
70     display_board(board)
71     if human_turn:
72         enter_move(board)
73         victor = victory_for(board,'O')
74     else:
75         draw_move(board)
76         victor = victory_for(board,'X')
77     if victor != None:
78         break
79     human_turn = not human_turn
80     free = make_list_of_free_fields(board)
81
82 display_board(board)
83 if victor == "you":
84     print("¡Has ganado!")
85 elif victor == "me":
86     print("¡He ganado!")
87 else:
88     print("Empate!")

```

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 4 | X | 6 |
| 7 | 8 | 9 |

Ingresa tu movimiento:

Buscar
10:13 p. m. 02/08/2023

Course: Curso Python Estadías
Edube Sandbox

edube.org/sandbox

Google
Multimedia
Escuela
Investigación Labs
Estadías
Cisco Networking A...

OPENEDG
Run
Share Code
Back to Course
Python 3.7

```

40 else:
41     who = None # No debemos de caer aquí!
42     cross1 = cross2 = True # para las diagonales
43     for rc in range(3):
44         if board[rc][0] == sgn and board[rc][1] == sgn and board[rc][2] == sgn: # check row rc
45             return who
46         if board[0][rc] == sgn and board[1][rc] == sgn and board[2][rc] == sgn: # check column rc
47             return who
48         if board[rc][rc] != sgn: # revisar la primera diagonal
49             cross1 = False
50         if board[2 - rc][2 - rc] != sgn: # revisar la segunda diagonal
51             cross2 = False
52     if cross1 or cross2:
53         return who
54     return None
55
56 def draw_move(board):
57     free = make_list_of_free_fields(board) # crea una lista de los cuadros vacios o libres
58     cnt = len(free)
59     if cnt > 0: # si la lista no está vacía, elegir un lugar para "X" y colocarla
60         this = randrange(cnt)
61         row, col = free[this]
62         board[row][col] = 'X'
63
64 board = [
65     [3 * j + i + 1 for i in range(3)] for j in range(3) ] # crear un tablero vacío
66 board[1][1] = 'X' # colocar la primera "X" en el centro
67 free = make_list_of_free_fields(board)
68 human_turn = True # ¿De quien es turno ahora?
69 while len(free):
70     display_board(board)
71     if human_turn:
72         enter_move(board)
73         victor = victory_for(board,'O')
74     else:
75         draw_move(board)
76         victor = victory_for(board,'X')
77     if victor != None:
78         break
79     human_turn = not human_turn
80     free = make_list_of_free_fields(board)
81
82 display_board(board)
83 if victor == "you":
84     print("¡Has ganado!")
85 elif victor == "me":
86     print("¡He ganado!")
87 else:
88     print("Empate!")

```

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 4 | X | 6 |
| 7 | 8 | 9 |

Ingresa tu movimiento:

Buscar
10:14 p. m. 02/08/2023