Contenido creado por: Isabel Maniega

Question 1 (Basics)

The folder created by Python used to store pyc files is named:

A. `__pycfiles__`

B. `__cache__`

C. `__pycache__`

D. `__pyc__`

Solution 1

In [2]:
```
# Solución
# C
```

Question 2 (Data Aggregates)

What is the expected output of the following code?

```
fruits1 = ['Apple', 'Pear', 'Banana']
fruits2 = fruits1
fruits3 = fruits1[:]

fruits2[0] = 'Cherry'
fruits3[1] = 'Orange'

res = 0

for i in (fruits1, fruits2, fruits3):
    if i[0] == 'Cherry':
        res += 1
    if i[1] == 'Orange':
        res += 10

print(res)
```

A. `12`

B. `11`

C. `22`

D. `0`

Solution 2

In [4]:
```
# resuelto paso a paso..
```

In [5]:
```
fruits1 = ['Apple', 'Pear', 'Banana']
print('fruits1 inicial: ', fruits1)
print('\n')
fruits2 = fruits1
print('fruits2 = fruits1')
print('fruits1: ', fruits1)
print('fruits2: ', fruits2)
print('\n')
fruits3 = fruits1[:]
print('fruits3 = fruits1[:]')
print('fruits1: ', fruits1)
print('fruits3: ', fruits3)
print('\n')
fruits2[0] = 'Cherry'
print("fruits2[0] = 'Cherry'")
print('fruits1: ', fruits1)
print('fruits2: ', fruits2)
print('\n')
fruits3[1] = 'Orange'
print("fruits3[1] = 'Orange'")
print('fruits1: ', fruits1)
print('fruits3: ', fruits3)
print('\n')
```

```
fruits1 inicial:  ['Apple', 'Pear', 'Banana']


fruits2 = fruits1
fruits1:  ['Apple', 'Pear', 'Banana']
fruits2:  ['Apple', 'Pear', 'Banana']


fruits3 = fruits1[:]
fruits1:  ['Apple', 'Pear', 'Banana']
fruits3:  ['Apple', 'Pear', 'Banana']


fruits2[0] = 'Cherry'
fruits1:  ['Cherry', 'Pear', 'Banana']
fruits2:  ['Cherry', 'Pear', 'Banana']


fruits3[1] = 'Orange'
fruits1:  ['Cherry', 'Pear', 'Banana']
fruits3:  ['Apple', 'Orange', 'Banana']
```

In [6]:
```
print(fruits1, fruits2, fruits3)
```

```
['Cherry', 'Pear', 'Banana'] ['Cherry', 'Pear', 'Banana'] ['Apple', 'Orang
e', 'Banana']
```

In [7]:
```
for i in (fruits1, fruits2, fruits3):
    print(i)
```

```
['Cherry', 'Pear', 'Banana']
['Cherry', 'Pear', 'Banana']
['Apple', 'Orange', 'Banana']
```

In [8]:
```
# bucle final
```

In [9]:
```
res = 0

for i in (fruits1, fruits2, fruits3):     # para cada lista por separado
    if i[0] == 'Cherry':
        res += 1
    if i[1] == 'Orange':
        res += 10

    # res = 0

    # ['Cherry', 'Pear', 'Banana'] (primera lista)
    #    +1

    # ['Cherry', 'Pear', 'Banana'] (segunda lista)
    #    +1

    # ['Apple', 'Orange', 'Banana'] (tercera lista)
    #              +10

    # +1 +1 +10 ==> +12

print(res)
```

```
12
```

In [10]:
```
# TODO DE UN PASO
```

In [11]:
```
fruits1 = ['Apple', 'Pear', 'Banana']
fruits2 = fruits1
fruits3 = fruits1[:]

fruits2[0] = 'Cherry'
fruits3[1] = 'Orange'

res = 0

for i in (fruits1, fruits2, fruits3):
    if i[0] == 'Cherry':
        res += 1
    if i[1] == 'Orange':
        res += 10

print(res)
```

```
12
```

In [12]:
```
# Solución
# A
```

Question 3 (Data Aggregates)

An alternative name for a data structure called a stack is:

A. FIFO B. LIFO C. FOLO

Solution 3

In [14]:
```
# stack --> pila, apilamiento
# la estructura en la cual APILAS, en un montón,
# en ella el último que entra es el primero que sale
# LAST IN FIRST OUT (LIFO)


# LIFO
```

In [15]:
```
# Solución
# B
```

Question 4 (Basics)

UNICODE is a standard:

A. like ASCII, but much more expansive

B. used by coders from Universities

C. for coding floating-point numbers

D. honored by the whole universe

Solution 4

In [17]:
```
# Solución
# A
```

Question 5 (Operators)

What is the expected output of the following code if the user enters 2 and 4?

```
x = float(input())
y = float(input())
print(y ** (1 / x))
```

A. `1.0`

B. `2.0`

C. `0.0`

D. `4.0`

Solution 5

In [19]:
```
# 2 and 4

x = float(input())    # 2.00
y = float(input())    # 4.00
print(y ** (1 / x))
# 4 ** (1 / 2.00) = 4 ** 0.50
```

```
2
4
2.0
```

In [20]:
```
4**0.50
```

Out[20]: 2.0

In [21]:
```
4**0, 4**1, 'entonces..', 4**0.5
```

Out[21]: (1, 4, 'entonces..', 2.0)

In [22]:
```
# Solución
# B
```

Question 6 (Data Types)

What is the result of the following code?

```
x = (3, )
print(len(x))
```

A. `2`

B. `3`

C. The program will cause an error

D. `1`

Solution 6

In [5]:
```
x = (3, )
print(len(x))
```

```
1
```

In [25]:
```
# Solución
# D
```

Question 7 (Operators)

Right-sided binding means that the following expression

```
1 ** 2 ** 3
```

will be evaluated:

A. from left to right

B. from right to left

C. in random order

Solution 7

In [27]:
```
1**2**3
```

Out[27]: 1

In [28]:
```
1**(2**3)

# from right to left
# de derecha a izquierda
```

Out[28]: 1

In [29]:
```
# Solución
# B
```

Question 8 (Functions)

What is the expected output of the following code?

```
def test(x=1, y=2):
    x = x + y
    y += 1
    print(x, y)

test()
```

A. `1 1`

B. The code is erroneous

C. `1 3`

D. `3 3`

E. `3 1`

Solution 8

In [31]:
```
def test(x=1, y=2):
    x = x + y          # x = 1 + 2 => x = 3
    y += 1             # y = y + 1 => y = 2 + 1 = 3
    print(x, y)        # 3 3


test()
```

```
3 3
```

In [32]:
```
# Solución
# D
```

Question 9 (Basics)

You have the following file.

index.py:

```
from sys import argv
print(argv[0])
```

You run the file by executing the following command in the terminal.

```
python index.py Hello
```

What is the expected oputput?

A. `IndexError`

B. `Hello`

C. `index.py`

D. `ImportError`

Solution 9

In [34]:
```
# Solución
# C
```

cómo comprobarlo

In [35]:
```
"""
ir a un bloc de notas (editor de textos en Ubuntu)

escribir el siguiente código:

from sys import argv
print(argv)

entonces, guardar como todos los archivos,
de nombre: "prueba.py" mismanente en el escritorio

abrir la consola en esa ruta, (si es escritorio, ahí)

-> en ubuntu:
botón derecho en el archivo en el escritorio -> mostrar en files
en esa ruta sin hacer click en archivo, botón derecho, abrir en una termi
abre la consola en esa ruta


y ejecutar el archivo, haciendo lo siguiente:

python3 index.py Hello World   (ubuntu)
python index.py Hello World    (Windows)

(python 3 en vez de Python para Ubuntu)

nos quedará:

['prueba.py', 'Hello', 'World']

y podríamos acceder a cualquier elemento de esos.
index 0, 1 y 2
"""

# entonces...podremos acceder al string prueba.py si definimos index 0

# from sys import argv
# print(argv[0])

# y ejecutando:
# python index.py Hello
```

Out[35]:
```
'\nir a un bloc de notas (editor de textos en Ubuntu)\n\nescribir el sig
uiente código:\n    \nfrom sys import argv\nprint(argv)\n\nentonces, gua
rdar como todos los archivos,\nde nombre: "prueba.py" mismanente en el e
scritorio\n    \nabrir la consola en esa ruta, (si es escritorio, ahí)\n
\n-> en ubuntu:\nbotón derecho en el archivo en el escritorio -> mostrar
en files\nen esa ruta sin hacer click en archivo, botón derecho, abrir e
n una terminal.\nabre la consola en esa ruta\n\n\ny ejecutar el archivo,
haciendo lo siguiente:\n    \npython3 index.py Hello World   (ubuntu)\np
ython index.py Hello World    (Windows)\n\n(python 3 en vez de Python pa
ra Ubuntu)\n    \nnos quedará: \n    \n[\'prueba.py\', \'Hello\', \'Worl
d\']\n\ny podríamos acceder a cualquier elemento de esos.\nindex 0, 1 y
2\n'
```

Question 10 (Data Aggregates)

What is the expected output of the following code?

```
t1 = (1, 4, 9)
t2 = ('A', 'D', 'Z')
t3 = (True, False, None)
t4 = (5.0, 7.5, 9.9)

t1, t3 = t2, t4
print(t1)
```

A. `(5.0, 7.5, 9.9)`

B. `(1, 4, 9)`

C. The program will cause an error

D. `('A', 'D', 'Z')`

Solution 10

In [37]:
```
t1 = (1, 4, 9)
t2 = ('A', 'D', 'Z')
t3 = (True, False, None)
t4 = (5.0, 7.5, 9.9)

t1, t3 = t2, t4
# t1 = ('A', 'D', 'Z')
# t3 = (5.0, 7.5, 9.9)

print(t1)  # ('A', 'D', 'Z')
```

```
('A', 'D', 'Z')
```

In [38]:
```
# Solución
# D
```

Question 11 (Data Agregates)

What is the expected output of the following code?

```
data = {'a': 1, 'b': 2, 'c': 3}
print(data['a', 'b'])
```

A. The code is erroneous

B. `(1,2)`

C. `[1,2]`

D. `{'a':1, 'b':2}`

Solution 11

In [40]:
```
"""data = {'a': 1,
        'b': 2,
        'c': 3}

print(data['a', 'b'])"""

# KeyError: ('a', 'b')
```

Out[40]:
```
"data = {'a': 1, \n        'b': 2, \n        'c': 3}\n\nprint(data['a', 'b'])"
```

In [41]:
```
# Solución
# A
```

explicación

In [42]:
```
# no se puede pasar varias keys, para obtener varios valores

# Otra posible opción, esta si es posible
data = {'a': 1, 'b': 2, 'c': 3}
print(data['a'], data['b'])
```

```
1 2
```

Question 12

Take a look at the snippet

and choose one of the following statements which is true:

```
nums = []
vals = nums
vals.append(1)
```

A. vals is longer than nums

B. nums is longer than vals

C. nums anb vals are of the same length

Solution 12

In [44]:
```
nums = []
vals = nums
vals.append(1)
```

In [45]:
```
print(vals)
print(nums)
```

```
[1]
[1]
```

In [46]:
```
# Solución
# C
```

Explicación con ejemplo similar

In [47]:
```
# otro ejemplo, similar,
# en vez de modificar una de las listas
# modifico la otra

nums = []
vals = nums
nums.append(1)
```

In [48]:
```
print(vals)
print(nums)
```

```
[1]
[1]
```

Question 13 (Data Agregates)

What is the expected output of the following code?

```
data = (1, 2, 4, 8)
data = data[1:-1]
data = data[0]
print(data)
```

A. `(2)`

B. The code is erroneous

C. `2`

D. `(2,)`

Solution 13

In [50]:
```
data = (1, 2, 4, 8)
print('data inicial', data)
data = data[1:-1]    # data = (2,4)
print('data tras [1,-1], osea DESDE index 1 hasta el último elemento no i
data = data[0]       # data = 2
print('data, su elemento de index 0: ', data)
print('SOLUCIÓN:')
print(data)
```

```
data inicial (1, 2, 4, 8)
data tras [1,-1], osea DESDE index 1 hasta el último elemento no incluído:
(2, 4)
data, su elemento de index 0:  2
SOLUCIÓN:
2
```

In [51]:
```
# Solución
# C
```

Question 14

How many stars will the following snippet print to the monitor?

```
x = 16
while x > 0:
    print('*')
x //= 2
```

A. five

B. The code will enter an infinite loop

C. one

D. three

Solution 14

In [53]:
```
"""
x = 16
```

```
while x > 0:
    print('*')
x //= 2

BUCLE INFINITO
"""
```

Out[53]: "\nx = 16\n\nwhile x > 0:\n   print('*')\nx //= 2 \n\nBUCLE INFINITO \n"

In [54]:
```
# Solución
# B
```

EJEMPLO SIMILAR, PERO EN ESTE CASO, SI FUNCIONA CORRECTAMENTE

In [55]:
```
x = 16

while x > 0:
    print('*')
    x //= 2      # x = x // 2  (cociente)

    # 16->8->4->2->1 => 5 *
    # 1//2 => 0
```

*
*
*
*
*

Question 15 (Basics)

Select the true statements?

(Select two answers)

A. Python is a good choice for creating and executing tests for applications

B. Python is a good choice for low-level programming, e.g. when you want to implement an effective driver

C. Python is free, open-source, and multiplatform

D. Python 3 is backwards compatible with Python 2

Solution 15

In [57]:
```
# A and C
```

Question 16 (Error Handling)

The part of your code where you think an exception may occur should be placed inside:

A. the `try:` branch

B. the `exception:` branch

C. the `except:` branch

Solution 16

In [59]:
```
# Solución
# A
```

Question 17 (Control Flow)

How many elements will the following list contain?

```
data = [i for i in range(-1, 2)]
```

A. three

B. one

C. zero

D. two

E. four

Solution 17

In [61]:
```
data = [i for i in range(-1, 2)]

# range(-1,2) => -1, 0, 1

print(data)
print(len(data))
```

```
[-1, 0, 1]
3
```

In [62]:
```
# Solución
# A
```

Question 18 (Basics)

Which one of the following is an example of a Python file extension?

A. p

B. py

C. pi

Solution 18

In [64]:
```
# Solución
# B
```

Question 19 (Data Types)

What is the expected output of the following code?

```
print(type(1J))
```

A. `<type 'float'>`

B. `<type 'dict'>`

C. `<type 'complex'>`

D. `<type 'unicode'>`

Solution 19

In [66]:
```
print(type(1J))
```

```
<class 'complex'>
```

In [67]:
```
# Solución
# C
```

Explicación: números complejos/imaginarios

In [68]:
```
print(type(2))
```

```
<class 'int'>
```

In [69]:
```
# print(type(1I))

# SyntaxError: invalid syntax
```

In [70]:
```
print(type(3+4J))
```

```
<class 'complex'>
```

In [71]:
```
3+2J
```

Out[71]: (3+2j)

In [72]:
```
(3+2J)+(4+3J)

# se suman las partes reales por un lado
# se suman las partes imaginarias por el otro
```

Out[72]: (7+5j)

Question 20 (Data Types)

You want to print each name of the list on a new line.

```
data = ['Peter', 'Paul', 'Mary', 'Jane']
```

Which statement will you use?

A. `print(data.join('%s\n', names))`

B. `print(data.join('\n'))`

C. `print(data.concatenate('\n'))`

D. `print('\n'.join(data))`

Solution 20

In [74]:
```
data = ['Peter', 'Paul', 'Mary', 'Jane']
data
```

Out[74]: ['Peter', 'Paul', 'Mary', 'Jane']

In [75]:
```
# A
# print(data.join('%s\n', names))

# AttributeError: 'list' object has no attribute 'join'
```

In [76]:
```
# B
# print(data.join('\n'))
```

```
# AttributeError: 'list' object has no attribute 'join'
```

In [77]:
```
# C
# print(data.concatenate('\n'))

# AttributeError: 'list' object has no attribute 'concatenate'
```

In [78]:
```
# D
print('\n'.join(data))
```

```
Peter
Paul
Mary
Jane
```

In [79]:
```
# Solución
# D
```

Question 21 (Functions)

What is the expected output of the following code?

```
def func(x, y=2):
    num = 1
    for i in range(y):
        num = num * x
    return num

print(func(4))
print(func(4, 4))
```

A.

```
16
256
```

B.

```
8
16
```

C.

```
128
512
```

D.

```
32
1024
```

Solution 21

In [81]:
```
def func(x, y=2):
    num = 1
    for i in range(y):
        num = num * x
    return num

print(func(4))
# x=4, y=2
# range(2) => 0,1
# num = num * x
# num = 1   * 4 => num = 4   (0)
# num = 4   * 4 => num = 16  (1)
# ===> 16 ===> A única opción posible

# PERO HAREMOS LA SEGUNDA LLAMADA IGUALMENTE, Por confirmar.

print(func(4, 4))
# x=4, y=4
# range(4) => 0,1,2,3
# num = num * x
# num =  1   * 4 => num =  4    (0)
# num =  4   * 4 => num =  16   (1)
# num = 16   * 4 => num =  64   (2)
# num = 64   * 4 => num = 256   (3)
# ===> 256 ===> A es la única opción posible
```

```
16
256
```

In [82]:
```
# Solución
# A
```

Question 22 (Functions)

What is the expected output of the following code?

```
def test(x, y=23, z=10):
    print('x is', x, 'and y is', y, 'and z is', z)

test(3, 7)
test(42, z=24)
test(z=60, x=100)
```

A.

```
x is 7 and y is 3 and z is 10
x is 42 and y is 23 and z is 24
```

```
x is 23 and y is 100 and z is 60
```

B.

```
x is 3 and y is 7 and z is 10
x is 23 and y is 42 and z is 24
x is 60 and y is 100 and z is 23
```

C. `The code is erroneous`

D.

```
x is 3 and y is 7 and z is 10
x is 42 and y is 23 and z is 24
x is 100 and y is 23 and z is 60
```

Solution 22

In [84]:
```
def test(x, y=23, z=10):
    print('x =', x, ', y =', y, ', z =', z)
```

In [85]:
```
test(3, 7)           # 3, 7, 10
```

```
x = 3 , y = 7 , z = 10
```

In [86]:
```
test(42, z=24)       # 42, 23, 24
```

```
x = 42 , y = 23 , z = 24
```

In [87]:
```
test(z=60, x=100)    # 100, 23, 60
```

```
x = 100 , y = 23 , z = 60
```

In [88]:
```
# Solución
# D
```

Question 23 (Operators)

What is the expected output of the following code?

```
x = True
y = False
z = False

if x or y and z:
    print('TRUE')
else:
    print('FALSE')
```

A. `TRUE`

B. `FALSE`

C. None of the above

D. The code is erroneous

Solution 23

In [90]:
```
x = True
y = False
z = False

if x or y and z:     # True or (False and False)
    print('TRUE')    # TRUE
else:
    print('FALSE')
```

TRUE

In [91]:
```
# Solución
# A
```

Question 24 (Data Aggregates)

What is the expected output of the following code?

```
nums = [1, 2, 3]
data = ('Peter',) * (len(nums) - nums[::-1][0])
print(data)
```

A. `()`

B. `('Peter', 'Peter')`

C. `'Peter')`

D. `PeterPeter`

E. The code is erroneous

Solution 24

In [93]:
```
nums = [1, 2, 3]
```

In [94]:
```
nums[::-1]
```

Out[94]: [3, 2, 1]

In [95]:
```
nums[::-1][0]
```

Out[95]:
```
3
```

In [4]:
```
nums = [1, 2, 3]
data = ('Peter',)  * (len(nums) - nums[::-1][0])
# data = ('Peter') *     3      -       3
# data = ('Peter') *            0
print(data) # ()
```

```
()
```

In [97]:
```
# Solución
# A
```

Question 25 (Error Handling)

What is the expected behavior of the following program?

```
foo = (1, 2, 3)
foo.index(0)
```

A. The program will cause a `SyntaxError` exception

B. The program will cause a `ValueError` exception

C. The program will output `1` to the screen

D. The program will cause a `TypeErro` exception

E. The program will cause a `AttributeError` exception

Solution 25

In [99]:
```
"""foo = (1, 2, 3)
foo.index(0)"""


# ValueError: tuple.index(x): x not in tuple
```

Out[99]:
```
'foo = (1, 2, 3)\nfoo.index(0)'
```

In [100…
```
# Solución
# B
```

In [101…
```
# foo.index?
```

In [102…
```
foo = (1, 2, 3)

print(foo.index(1))
```

```
# index donde está el 1

print(foo.index(2))
# index donde está el 2

print(foo.index(3))
# index donde está el 3
```

```
0
1
2
```

In [103…
```
# otro ejemplo

vocales = ('a', 'e', 'i', 'o', 'u')

# index de una vocal
# ejemplo 'i'

index = vocales.index('i')
index
```

Out[103… 2

Question 26 (Control Flow)

What is the expected output of the following code?

```
print(len([i for i in range(0, -2)]))
```

A. `3`

B. `0`

C. `2`

D. `1`

Solution 26

In [105…
```
[i for i in range(0, -2)]   # no hay elementos en ese rango
```

Out[105… []

In [106…
```
import numpy as np
np.arange(0,-2)
```

Out[106… array([], dtype=int64)

In [107…
```
print(len([i for i in range(0, -2)]))

# 0
```

0

In [108…
```
# Solución
# B
```

Question 27 (Control Flow)

The ABC company is creating a program that allows customers to log the number of miles biked. The program will send messages based on how many miles the customer logs. You create the following Python code.

```
???
    name = input('What is your name? ')
    return name

???
    calories = miles * calories_per_mile
    return calories

distance = int(input('How many miles did you bike this
week? '))
burn_rate = 50
biker = get_name()
calories_burned = calc_calories(distance, burn_rate)
print(biker + ', you burned about', calories_burned,
'calories.')
```

What would you insert instead of ??? and ???

A. `def calc_calories(miles, burn_rate):`

B. `def calc_calories(miles, calories_per_mile):`

C. `def calc_calories():`

D. `def get_name():`

E. `def get_name(name):`

F. `def get_name(biker):`

Solution 27

In [110…
```
# ???
def get_name():
    name = input('What is your name? ')
    return name

# ???
def calc_calories(miles, calories_per_mile):
```

```
    calories = miles * calories_per_mile
    return calories


distance = int(input('How many miles did you bike this week? '))
burn_rate = 50
biker = get_name()
calories_burned = calc_calories(distance, burn_rate)
print(biker + ', you burned about', calories_burned, 'calories.')
```

```
How many miles did you bike this week? 50
What is your name? jose
jose, you burned about 2500 calories.
```

In [111…
```
# Solución
# B y D
```

Question 28 (Control Flow)

How many stars (*) will the following code output to the screen?

```
n = 0
if n > 0:
    print("*")
elif n == True:
    print("**")
else:
    print("***")
```

A. one

B. two

C. three

D. six

Solution 28

In [113…
```
n = 0
if n > 0:
    print("*")
elif n == True:
    print("**")
else:
    print("***")  # ejecuta 3 *
```

```
***
```

In [114…
```
# Solución
# C
```

Question 29 (Data Types)

The most important difference between integer and floating-point numbers lies in the fact that:

A. The are stored differently in the computer memory

B. they cannot be used simultaneously

C. integers cannot be literal, while floats can

Solution 29

In [116…
```
# Solución
# A
```

Question 30 (Functions)

What is the expected output of the following code?

```
def func(item):
    item += [1]

data = [1, 2, 3, 4]
func(data)
print(len(data))
```

A. `2`

B. The code is erroneous

C. `5`

D. `4`

Solution 30

In [118…
```
# Apendizado de listas  (otra posible opción)

[1, 2, 3, 4] + [1]
```

Out[118… `[1, 2, 3, 4, 1]`

In [119…
```
def func(item):
    item += [1]             # item = item + [1]
```

```
data = [1, 2, 3, 4]
func(data)                  # data = [1, 2, 3, 4] + [1]

# data = [1, 2, 3, 4, 1]

print(len(data))            # 5
```

```
5
```

In [120…

```
# Solución
# C
```

Gracias por la atención

Isabel Maniega