Contenido creado por: Isabel Maniega

Question 1 (Data Aggregates)

What is the expected output of the following code?

```
data = (1, 2, 3, 4)
data = data[-2:-1]
data = data[-1]
print(data)
```

A. `1`

B. `3`

C. `4`

D. `2`

Solution 1

In [2]:
```
data = (1, 2, 3, 4)
print('data inicial:', data)

data = data[-2:-1]  # data = 3
print('data tras: data[-2:-1]:', data)

data = data[-1]     # 3
print('data tras: data[-1]:', data)
```

```
data inicial: (1, 2, 3, 4)
data tras: data[-2:-1]: (3,)
data tras: data[-1]: 3
```

In [3]:
```
# Solución
# B
```

Question 2 (Functions)

What is the expected output of the following code?

```
def func(n):
    s = '*'
    for i in range(n):
        s += s
    yield s
```

```
for x in func(2):
    print(x, end='')
```

A. `***`

B. `*`

C. `****`

D. `**`

Solution 2

In [5]:
```
def func(n):
    s = '*'
    for i in range(n):
        s += s           # s = s + s ===>  # s = 2 * s
    yield s


for x in func(2):
    print(x, end='')

# s = 2(*) = 2*
# s = 2(2*) = 4*
# ****
```

****

In [6]:
```
# Solución
# C
```

Question 3 (Data Aggregates)

What is the expected output of the following code?

```
data = {}
data['2'] = [1, 2]
data['1'] = [3, 4]

for i in data.keys():
    print(data[i][1], end=' ')
```

A. `4 2`

B. `1 3`

C. `3 1`

D. `2 4`

Solution 3

In [8]:
```
# explicación completa paso a paso
```

In [9]:
```
data = {}
data['2'] = [1, 2]   # {'2': [1,2]}
data['1'] = [3, 4]   # {'2': [1,2], '1': [3, 4]}

for i in data.keys():
    print('i <key>:', i)
    print('data[i] <apuntamos al valor>:', data[i])
    print('data[i][0] ese valor en posición 0:', data[i][0])
    print('data[i][1] ese valor en posición 1:', data[i][1])
    print("\n")

# {'2': [1,2],
#  '1': [3, 4]}
```

```
i <key>: 2
data[i] <apuntamos al valor>: [1, 2]
data[i][0] ese valor en posición 0: 1
data[i][1] ese valor en posición 1: 2


i <key>: 1
data[i] <apuntamos al valor>: [3, 4]
data[i][0] ese valor en posición 0: 3
data[i][1] ese valor en posición 1: 4
```

In [10]:
```
# el propio ejercicio
```

In [11]:
```
data = {}
data['2'] = [1, 2]   # {'2': [1,2]}
data['1'] = [3, 4]   # {'2': [1,2], '1': [3, 4]}

for i in data.keys():
    print(data[i][1], end=' ')
    # 2 4



# {'2': [1,2],
#  '1': [3, 4]}
```

```
2 4
```

In [12]:
```
# Solución
# D
```

Question 4 (Control Flow)

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

```
data = [[x for x in range(3)] for y in range(3)]
for i in range(3):
    for j in range(3):
        if data[i][j] % 2 != 0:
            print('*')
```

A. six

B. three

C. zero

D. nine

Solution 4

In [14]:
```
# ---[x for x in range(3)]---
```

In [15]:
```
[x for x in range(3)]
```

Out[15]:
```
[0, 1, 2]
```

In [16]:
```
# ---data---
```

In [17]:
```
data = [[x for x in range(3)] for y in range(3)]

# data = [[0,1,2] for y in range(3)]
'''
data=
[
[0,1,2],
[0,1,2],
[0,1,2]
]
'''

print(data)
```

```
[[0, 1, 2], [0, 1, 2], [0, 1, 2]]
```

In [18]:
```
# ----filas y columnas en la matriz----
```

In [19]:
```
data = [[0, 1, 2], [0, 1, 2], [0, 1, 2]]

for i in range(3):
    for j in range(3):
        print('Fila i:', i, ' - Columna j:', j,'- data[i][j]:', data[i][j
```

```
Fila i: 0  - Columna j: 0 - data[i][j]: 0
Fila i: 0  - Columna j: 1 - data[i][j]: 1
Fila i: 0  - Columna j: 2 - data[i][j]: 2
Fila i: 1  - Columna j: 0 - data[i][j]: 0
Fila i: 1  - Columna j: 1 - data[i][j]: 1
Fila i: 1  - Columna j: 2 - data[i][j]: 2
Fila i: 2  - Columna j: 0 - data[i][j]: 0
Fila i: 2  - Columna j: 1 - data[i][j]: 1
Fila i: 2  - Columna j: 2 - data[i][j]: 2
```

In [20]:
```
# ----ejemplos de resto 0 y distinto de 0 en división por 2----
```

In [21]:
```
0%2, 1%2, 2%2
```

Out[21]:
```
(0, 1, 0)
```

In [22]:
```
# ----todo junto en el ejercicio----
```

In [23]:
```
data = [[x for x in range(3)] for y in range(3)]
for i in range(3):
    for j in range(3):
        if data[i][j] % 2 != 0:    # si el resto de data[i][j] entre 2 es
            print('en este número se cumple que es distinto de 0 el resto
            print('*')
```

```
en este número se cumple que es distinto de 0 el resto: 1
*
en este número se cumple que es distinto de 0 el resto: 1
*
en este número se cumple que es distinto de 0 el resto: 1
*
```

In [24]:
```
# Solución
# B
```

explicación

obviamente solo el número 1 cumple que 1%2 != 0

Question 5 (Data Types)

Consider the following code.

```
start = input('How old were you at the time of joining?')
now = input('How old are you today?')
```

Which of the following codes will print the right output?

A.

```
print(
     'Congrats on '
     + int(now - start)
```

```
        + ' years of service!'
    )
```

B.

```
    print(
        'Congrats on '
        + str(now - start)
        + ' years of service!'
    )
```

C.

```
    print(
        'Congrats on '
        + (int(now) - int(start))
        + ' years of service!'
    )
```

D.

```
    print(
        'Congrats on '
        + str(int(now) - int(start))
        + ' years of service!'
    )
```

Solution 5

In [26]:
```
# D
```

In [27]:
```
start = input('How old were you at the time of joining?')     # 23   (es
now = input('How old are you today?')                         # 28

print(
     'Congrats on '
     + str(int(now) - int(start))
     + ' years of service!'
)
```

```
How old were you at the time of joining?23
How old are you today?28
Congrats on 5 years of service!
```

In [28]:
```
# Solución
# D
```

explicación

no pide por teclado el número entero: int(input)

y concatena strings, para que ejecute

In [29]:
```
"""
ESTA ES LA C, LA OTRA CON LA QUE SE PODÍA DUDAR

NO SIRVE PUESTO QUE ESTAMOS CONCATENANDO STRINGS


start = input('How old were you at the time of joining?')
now = input('How old are you today?')

print(
     'Congrats on '
     + int(now) - int(start)
     + ' years of service!'
)
"""


# TypeError: can only concatenate str (not "int") to str
```

Out[29]:
```
"\nESTA ES LA C, LA OTRA CON LA QUE SE PODÍA DUDAR\n\nNO SIRVE PUESTO QU
E ESTAMOS CONCATENANDO STRINGS\n\n\nstart = input('How old were you at t
he time of joining?')\nnow = input('How old are you today?')\n\nprint(\n
'Congrats on '\n     + int(now) - int(start)\n     + ' years of servic
e!'\n)\n"
```

Question 6 (Control Flow)

Which one of the lines should you put in the snippet below to match the expected output?

Expected output:

```
1245
```

Code:

```
c = 0
while c < 5:
    c = c + 1
    if c == 3:
        # enter code here
    print(c, end="")
```

A. `exit`

B. `continue`

C. `print()`

D. `break`

Solution 6

In [31]:
```
# expected output 1245

c = 0
while c < 5:
    c = c + 1
    if c == 3:
        continue
    print(c, end="")
```

1245

In [32]:
```
# Solución
# B
```

Question 7 (Functions)

What is the expected output of the following code?

```
def func(x=2, y=3):
    return x * y

print(func(y=2, 3))
```

A. The code is erroneous

B. `4`

C. `6`

D. `2`

Solution 7

In [34]:
```
"""
def func(x=2, y=3):
    return x * y


print(func(y=2, 3))
"""

# SyntaxError: positional argument follows keyword argument
```

Out[34]:
```
'\ndef func(x=2, y=3):\n    return x * y\n \n\nprint(func(y=2, 3))\n'
```

In [35]:
```
# Solución
# A
```

un ejemplo similar que funciona

In [36]:
```
def func(x=2, y=3):
    return x * y


print(func(y=5, x=4))
```

```
20
```

Question 8 (Data Aggregates)

What is the expected output of the following code?

```
data = {1: 0, 2: 1, 3: 2, 0: 1}
x = 0

for _ in range(len(data)):
    x = data[x]

print(x)
```

A. `0`

B. `1`

C. The code is erroneous

D. `2`

Solution 8

In [38]:
```
data = {1: 0,
        2: 1,
        3: 2,
        0: 1}

x=0

for _ in range(len(data)):   # len(data) = 4
    x = data[x]
 #  x = data[0] ==> x = 1
 #  x = data[1] ==> x = 0
 #  x = data[0] ==> x = 1
 #  x = data[1] ==> x = 0

print(x)   # 0

# LO HACE 4 VECES POR range(len(data))
# pero, en cada ocasión se tiene en cuenta el valor anterior
```

```
0
```

In [39]:
```
# Solución
# A
```

OJO!!! NO CONFUNDIR CON ESTE EJERCICIO----

In [40]:
```
data = {1: 0,
        2: 1,
        3: 2,
        0: 1}


# aqui no es necesario x  !!!
# x=0


for x in range(len(data)):   # len(data) = 4
    x = data[x]
# RANGE(LEN(3)) ===> 0-1-2-3

 #  x = data[0] ==> x = 1 (SI, PERO NO AFECTA, SOLO IMPORTA EL ÚLTIMO)
 #  x = data[1] ==> x = 0 (SI, PERO NO AFECTA, SOLO IMPORTA EL ÚLTIMO)
 #  x = data[2] ==> x = 1 (SI, PERO NO AFECTA, SOLO IMPORTA EL ÚLTIMO)
 #  x = data[3] ==> x = 2  ÚLTIMA VEZ QUE REASIGNA VALOR

print(x)   # 2
```

```
2
```

ejercicio original x=1

In [41]:
```
data = {1: 0,
        2: 1,
        3: 2,
        0: 1}

x=1

for _ in range(len(data)):   # len(data) = 4
    x = data[x]
 #  x = data[1] ==> x = 0
 #  x = data[0] ==> x = 1
 #  x = data[1] ==> x = 0
 #  x = data[0] ==> x = 1

print(x)   # 0

# LO HACE 4 VECES POR range(len(data))
# pero, en cada ocasión se tiene en cuenta el valor anterior
```

```
1
```

ejercicio original x=2

In [42]:
```
data = {1: 0,
        2: 1,
        3: 2,
        0: 1}
```

```
x=2

for _ in range(len(data)):   # len(data) = 4
    x = data[x]
 #  x = data[2] ==> x = 1
 #  x = data[1] ==> x = 0
 #  x = data[0] ==> x = 1
 #  x = data[1] ==> x = 0

print(x)   # 0

# LO HACE 4 VECES POR range(len(data))
# pero, en cada ocasión se tiene en cuenta el valor anterior
```

```
0
```

ejercicio original x=3

In [43]:
```
data = {1: 0,
        2: 1,
        3: 2,
        0: 1}

x=3

for _ in range(len(data)):   # len(data) = 4
    x = data[x]
 #  x = data[3] ==> x = 2
 #  x = data[2] ==> x = 1
 #  x = data[1] ==> x = 0
 #  x = data[0] ==> x = 1

print(x)   # 0

# LO HACE 4 VECES POR range(len(data))
# pero, en cada ocasión se tiene en cuenta el valor anterior
```

```
1
```

ejercicio original x=4

In [44]:
```
"""
data = {1: 0,
        2: 1,
        3: 2,
        0: 1}

x=4

for _ in range(len(data)):   # len(data) = 4
    x = data[x]
 #  x = data[4] ==> ??


print(x)
"""

# KeyError: 4
```

Out[44]:
```
'\ndata = {1: 0, \n        2: 1, \n        3: 2, \n        0: 1}\n\nx=4
\n \nfor _ in range(len(data)):   # len(data) = 4\n    x = data[x]
\n #  x = data[4] ==> ??\n\n    \nprint(x)   \n'
```

diferente longitud de data

In [45]:
```
data = {1: 0,
        2: 1}

x=2

for _ in range(len(data)):   # len(data) = 2
    x = data[x]
    print('dentro del bucle for:',  x)
#   x = data[2] ==> x = 1
#   x = data[1] ==> x = 0


print('al final vale:', x)   # 0
```

```
dentro del bucle for: 1
dentro del bucle for: 0
al final vale: 0
```

Question 9 (Operators)

Which of the following statements are correct?

Choose two.

A. `True and False` evaluates to `true`

B. `7 + False` evaluates to `False`

C. `type('')` returns `<class 'bool'>`

D. `True + 1` evaluates to `2`

E. `True or False` evaluates to `True`

Solution 9

In [47]:
```
# A
True and False # False

# True and False evaluates to true  ===> NO ES CORRECTO
```

Out[47]:
```
False
```

In [48]:
```
# B
7 + False # 7
```

```
# 7 + False evaluates to False  ======> NO ES CORRECTO
```

Out[48]: 7

In [49]:
```
# C
type('') # str

# type('') returns <class 'bool'>  ===> NO ES CORRECTO
```

Out[49]: str

In [50]:
```
# D
True + 1  # 1+1 = 2

# True + 1 evaluates to 2  ===========> ES CORRECTO
```

Out[50]: 2

In [51]:
```
# E
True or False # True

# True or False evaluates to True ====> ES CORRECTO
```

Out[51]: True

In [52]:
```
# Solución
# D y E
```

no confundir estos ejemplos..

In [53]:
```
type(''), type(' '), type('miString')
```

Out[53]: (str, str, str)

In [54]:
```
bool(''), bool('  '), bool('miString')
```

Out[54]: (False, True, True)

Question 10 (Error Handling)

What is the output of the following code if the user enters a 0?

```
try:
    value = input("Enter a value: ")
    print(int(value)/int(value))
except ValueError:
    print("Bad input...")
except ZeroDivisionError:
    print("Very bad input...")
except TypeError:
    print("Very very bad input...")
```

```
    except:
        print("Booo!")
```

A. `1.0`

B. `Bad input...`

C. `Booo!`

D. `Very bad input...`

E. `Very very bad input...`

F. `0.0`

Solution 10

In [56]:
```
# ejemplos previos
```

In [57]:
```
# numero entero
```

In [58]:
```
try:
    value = input("Enter a value: ")    # 10
    print(int(value)/int(value))
except ValueError:
    print("Bad input...")
except ZeroDivisionError:
    print("Very bad input...")
except TypeError:
    print("Very very bad input...")
except:
    print("Booo!")
```

```
Enter a value: 10
1.0
```

In [59]:
```
# numero decimal
```

In [60]:
```
try:
    value = input("Enter a value: ")      # 10.5
    print(int(value)/int(value))
except ValueError:
    print("Bad input...")
except ZeroDivisionError:
    print("Very bad input...")
except TypeError:
    print("Very very bad input...")
except:
    print("Booo!")
```

```
Enter a value: 10.5
Bad input...
```

In [61]:
```
# string
```

In [62]:
```
try:
    value = input("Enter a value: ")    # hola
    print(int(value)/int(value))
except ValueError:
    print("Bad input...")
except ZeroDivisionError:
    print("Very bad input...")
except TypeError:
    print("Very very bad input...")
except:
    print("Booo!")
```

```
Enter a value: hola
Bad input...
```

In [63]:
```
# 0 (el propio ejemplo)
```

In [64]:
```
try:
    value = input("Enter a value: ")    # 0
    print(int(value)/int(value))
except ValueError:
    print("Bad input...")
except ZeroDivisionError:
    print("Very bad input...")
except TypeError:
    print("Very very bad input...")
except:
    print("Booo!")
```

```
Enter a value: 0
Very bad input...
```

In [65]:
```
# Solución
# D (no B)
```

Question 11 (Operators)

What is the output of the following snippet?

```
y = 2 + 3 * 5.
print(y)
```

A. `17.0`

B. `25.0`

C. the snippet will cause an execution error

D. `17`

Solution 11

In [67]:
```
3*5
```

Out[67]: 15

In [68]:
```
3*5.
```

Out[68]: 15.0

In [69]:
```
3*5.0
```

Out[69]: 15.0

In [70]:
```
y = 2 + 3 * 5.    # 2 + 3 * 5.0  (EQUIVALENTE A 5.0)
print(y)          # 17.0
```

```
17.0
```

In [71]:
```
# Solución
# A
```

Question 12 (Functions)

What is the expected output of the following code?

```
def func(x):
    if x % 2 == 0:
        return 1
    else:
        return 2

print(func(func(2)))
```

A. `0`

B. `1`

C. `2`

D. The code is erroneous

Solution 12

In [73]:
```
def func(x):
    if x % 2 == 0:
        return 1
    else:
        return 2


print(func(func(2)))
```

```
#     func(1)        (el resto de 2/2 es 0 entonces retorna 1)
#     # 2
```

2

In [74]:
```
# Solución
# C
```

Question 13 (Operators)

Consider the following code.

```
languages = ['English', 'Spanish', 'German']
more_languages = ['English', 'Spanish', 'German']
extra_languages = more_languages
```

Which statement will print True to the monitor?

Choose two.

A. `print(more_languages is extra_languages)`

B. `print(languages == more_languages)`

C. `print(languages is extra_languages)`

D. `print(languages is more_languages)`

Solution 13

In [76]:
```
languages = ['English', 'Spanish', 'German']
more_languages = ['English', 'Spanish', 'German']
extra_languages = more_languages
```

In [77]:
```
# A.
print(more_languages is extra_languages)  # True
```

True

In [78]:
```
# B
print(languages == more_languages)  # True
```

True

In [79]:
```
# C.
print(languages is extra_languages)   # False
```

False

In [80]:
```
# D.
print(languages is more_languages)   # False
```

False

In [81]:
```
# Solución
# A y B
```

Question 14 (Data Aggregates)

What is the output of the following snippet?

```
my_list = ['Mary', 'had', 'a', 'little', 'lamb']

def my_list(my_list):
    del my_list[3]
    my_list[3] = 'ram'

print(my_list(my_list))
```

A. `['Mary', 'had', 'a', 'lamb']`

B. `['Mary', 'had', 'a', 'ramb']`

C. no output, the snippet is erroneous

D. `['Mary', 'had', 'a', 'little', 'lamb']`

Solution 14

In [83]:
```
"""
my_list = ['Mary', 'had', 'a', 'little', 'lamb']


def my_list(my_list):    # ['Mary', 'had', 'a', 'little', 'lamb']
    del my_list[3]       # ['Mary', 'had', 'a',           'lamb']
    my_list[3] = 'ram'   # ['Mary', 'had', 'a',           'ram']


print(my_list(my_list))
"""


# TypeError: 'function' object does not support item deletion
```

Out[83]:
```
"\nmy_list = ['Mary', 'had', 'a', 'little', 'lamb']\n \n\ndef my_list(my
_list):    # ['Mary', 'had', 'a', 'little', 'lamb']\n    del my_list[3]
# ['Mary', 'had', 'a',           'lamb']\n    my_list[3] = 'ram'   # ['M
ary', 'had', 'a',           'ram']\n \n \nprint(my_list(my_list))\n"
```

In [84]:
```
# Solución
# C
```

Question 15 (Control Flow)

The ABC company needs a way to find the count of particular letters in their publications to ensure that there is a good balance. It seems that there have been complaints about overuse of the letter e You need to create a function to meet the requirements.

Function accepts a list of words from a file, and a letter to search for. Returns count of the words containing that letter.

```
def count_letter(letter, word_list):
    count = 0
    for ???:
        if ???:
            count += 1
return count

word_list = []
# word_list is populated from a file. Code not shown.

letter = input('Which letter would you like to count?')
letter_count = count_letter(letter, word_list)
print('There are', letter_count, 'words with the letter',
letter)
```

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

A.

```
word is word_list
letter in word
```

B.

```
word in word_list
letter is word
```

C.

```
word_list in word
word in letter
```

D.

```
word_list in word
word in letter
```

E.

```
words in word_list
letter in word
```

F.

```
word in word_list
 letter in word
```

Solution 15

In [86]:
```python
def count_letter(letter, word_list):
    count = 0
    for word in word_list:
        if letter in word:
            count += 1
    return count

# -----para word_list nos inventamos un string cualquiera-----
# word_list = []
# word_list is populated from a file. Code not shown.
# ---
# word_list simulacion (ES MI EJEMPLO QUE SIMULA LOS DATOS DEL ARCHIVO)
word_list = 'ejemplo'


letter = input('Which letter would you like to count?')         # ejemplo
letter_count = count_letter(letter, word_list)
print('There are', letter_count, 'words with the letter', letter)
```

```
Which letter would you like to count?e
There are 2 words with the letter e
```

In [87]:
```python
# Solución
# F
```

Question 16 (Data Types)

The following code:

```
print(float("1, 3"))
```

A. `prints 1.3`

B. `prints 1,3`

C. raises a ValueError exception

D. `prints 13`

Solution 16

In [89]:
```
# print(float("1, 3"))

# ValueError: could not convert string to float: '1, 3'
```

In [90]:
```
# Solución
# C
```

ejemplos similares

In [91]:
```
# print(float("1. 3"))
# ValueError: could not convert string to float: '1. 3'
```

In [92]:
```
print(float("1.3"))    # ESTE NO TIENE ESPACIO ENTRE AMBOS NÚMEROS Y TIEN
```

```
1.3
```

In [93]:
```
# print(float("1, 3"))
# ValueError: could not convert string to float: '1, 3'
```

In [94]:
```
# print(float("1,3"))
# ValueError: could not convert string to float: '1,3'
```

Question 17 (Data Types)

Consider the following Python code:

```
distance = 1876.23
amount = +42E7
country = 'Italy'
```

What are the types of the variables distance, amount and country?

A. `float, str, str`

B. `float, float, str`

C. `double, str, float`

D. `float, int, str`

Solution 17

In [96]:
```
distance = 1876.23
amount = +42E7
country = 'Italy'

print(distance)
print(amount)
print(country)
```

```
1876.23
420000000.0
Italy
```

In [97]:
```
print('type(distance):', type(distance))   # float
print('type(amount):', type(amount))       # float
print('type(country):', type(country))     # str
```

```
type(distance): <class 'float'>
type(amount): <class 'float'>
type(country): <class 'str'>
```

In [98]:
```
# Solución
# B
```

Question 18 (Operators)

What is the result of the following operation?

```
1 + 1.0
```

A. `2`

B. the operation is illegal in Python

C. `11.0`

D. `2.0`

Solution 18

In [100…
```
1 + 1.0     # 2.0
```

Out[100… 2.0

In [101…
```
# Solución
# D
```

EVAL

eval is una built-in- function de python,

evalúa el string como una expresión de Python y retorna un entero

In [102…
```
# eval: ejemplo 1 (sin eval)
```

In [103…
```
x1 = input("Introduce un número")       # 10
print(x1)
print(type(x1))
```

```
Introduce un número10
10
<class 'str'>
```

In [104…
```
# ejemplo 1 (con eval)
```

In [105…
```
x2 = eval(input("Introduce un número"))    # 10
print(x2)
print(type(x2))
```

```
Introduce un número10
10
<class 'int'>
```

In [106…
```
# eval: ejemplo 2
```

In [107…
```
x = 10
x3 = eval('15 + x')
print(x3)
```

```
25
```

In [108…
```
# eval: ejemplo 3
```

In [109…
```
# HAY QUE AÑADIR LOS NUMEROS CON UNA COMA
# 10,30
# PARA QUE SEPA QUE ES X3 UNO DE ELLOS, Y3 EL OTRO

x3, y3 = eval(input('Introduce 2 números: '))    # 10, 20    (de una sola
print('x3:', x3, ', y3:', y3)
```

```
Introduce 2 números: 10,20
x3: 10 , y3: 20
```

In [110…
```
# eval: ejemplo 4
```

In [111…
```
x4, y4 = eval('3, 4')
print('x4:', x4, ', y4:', y4)
```

```
x4: 3 , y4: 4
```

In [112…
```
# eval: ejemplo 5
```

In [113…
```
"""
x5, y5 = eval('3 4')
print('x5:', x5, ', y5:', y5)
"""

# SyntaxError: unexpected EOF while parsing
```

Out[113…
```
"\nx5, y5 = eval('3 4') \nprint('x5:', x5, ', y5:', y5)\n"
```

In [114…
```
# el propio ejercicio 19
```

Question 19 (Data Types)

The following code reads two numbers.

Which of the following is the correct input for the code?

```
x, y = eval(input('Enter two numbers: '))
print(x)
print(y)
```

A. `3, 4`

B. `3 4`

C. None of the above

D. `<pre>3 4</pre>`

Solution 19

In [116…
```
x, y = eval(input('Enter two numbers: '))  # 3, 4    (DEBES AÑADIRLOS TAL
print(x)
print(y)
```

```
Enter two numbers: 3,4
3
4
```

In [117…
```
# Solución
# A
```

Question 20 (Data Aggregates)

What is the output of the following snippet?

```
my_list_1 = [1, 2, 3]
my_list_2 = []
for v in my_list_1:
    my_list_2.insert(0, v)
print(my_list_2)
```

A. `[1, 1, 1]`

B. `[3, 2, 1]`

C. `[1, 2, 3]`

D. `[3, 3, 3]`

Solution 20

In [119…
```
my_list_1 = [1, 2, 3]
my_list_2 = []
for v in my_list_1:         # v es: 1, 2, 3
    my_list_2.insert(0, v)  # index 0 valor v
    # inicial:        my_list_2 = []
    # insert(0, 1)    my_list_2 = [1]
    # insert(0, 2)    my_list_2 = [2, 1]
    # insert(0, 3)    my_list_2 = [3, 2, 1]

print(my_list_2)
```

```
[3, 2, 1]
```

In [120…
```
# Solución
# B
```

Question 21 (Operators)

The `//` operator:

A. does not exist

B. performs integer division

C. performs regular division

Solution 21

In [122…
```
7/2, 7//2, 7%2
```

Out[122… (3.5, 3, 1)

In [123…
```
# Solución
# B
```

Question 22 (Operators)

What is the expected output of the following code?

```
list1 = ['Peter', 'Paul', 'Mary', 'Jane']
list2 = ['Peter', 'Paul', 'Mary', 'Jane']

print(list1 is list2)
print(list1 == list2)

list1 = list2
```

```
print(list1 is list2)
print(list1 == list2)
```

A.

```
False
True
False
True
```

B.

```
False
True
True
True
```

C.

```
False
True
True
False
```

D.

```
False
False
True
True
```

Solution 22

In [125…
```
list1 = ['Peter', 'Paul', 'Mary', 'Jane']
list2 = ['Peter', 'Paul', 'Mary', 'Jane']

print(list1 is list2)   # False
print(list1 == list2)   # True

list1 = list2           # AL ASIGNAR UNA A LA OTRA, CAMBIA LA SENTENCIA Q

print(list1 is list2)   # True
print(list1 == list2)   # True
```

```
False
True
True
True
```

In [126…
```
# Solución
# B
```

Question 23 (Data Aggregates)

Which one of the lines should you put in the snippet below to match the expected output?

Expected output:

```
[1, 2, 4, 7]
```

Code:

```
list = [2, 7, 1, 4]
# enter code here
print(list)
```

A. `sort(list)`

B. `sorted(list)`

C. `list.sort()`

D. `list.sorted()`

Solution 23

In [128…
```
lista = [2, 7, 1, 4]
# # enter code here

lista.sort()
print(lista)



# DEBERIAN NO USAR UNA PALABRA RESERVADA
# COMO NOMBRE DE VARIABLE

# Expected output:
# [1, 2, 4, 7]
```

```
[1, 2, 4, 7]
```

In [129…
```
# Solución
# C
```

explicación: por defecto es reverse=False

In [130…
```
# lista.sort?
```

In [131…
```
# lista = [2, 7, 1, 4]
```

```
# NOS FIJAMOS EN SUS PARÁMETROS
# lista.sort?
```

In [132…
```
# reverse=False
```

In [133…
```
lista = [2, 7, 1, 4]
lista.sort(reverse=False)
print(lista)
```

```
[1, 2, 4, 7]
```

In [134…
```
# reverse=True
```

In [135…
```
lista = [2, 7, 1, 4]
lista.sort(reverse=True)
print(lista)
```

```
[7, 4, 2, 1]
```

Question 24 (Control Flow)

Which of the following code snippets will print all prime numbers between `2` and `100` to the monitor?

A.

```
num = 2
is_prime = True
while num <= 100:
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == True:
        print(num)
    num += 1
```

B.

```
num = 2
is_prime = True
while num <= 100:
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == False:
        print(num)
    num += 1
```

C.

```
num = 2
while num <= 100:
    is_prime = True
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == True:
        print(num)
```

D.

```
num = 2
while num <= 100:
    is_prime = True
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == True:
        print(num)
    num += 1
```

Solution 24

In [137…
```
# A
```

In [138…
```
num = 2
is_prime = True                 # aqui se encuentra: is_prime fuera de w
while num <= 100:
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == True:
        print(num)
    num += 1
```

```
2
3
```

In [139…
```
# D
```

In [140…
```
num = 2
while num <= 100:
    is_prime = True             # variable 'is_prime' fijada a True para
    for i in range(2, num):     # número testeado no primo ==> 'is_prime
        if num%i == 0:
            is_prime = False
            break
    if is_prime == True:        # 'is_prime' sigue siendo True ==> impri
        print(num, end=' ')     # añadido: end= ' ' para verlos todos en
    num += 1
```

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

In [141…
```
# B
```

In [142…
```
"""
num = 2
is_prime = True
while num <= 100:
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == False:
        print(num)
    num += 1

"""
```

Out[142…
```
'\nnum = 2\nis_prime = True\nwhile num <= 100:\n    for i in range(2, nu
m):\n        if num%i == 0:\n            is_prime = False\n            b
reak\n    if is_prime == False:\n        print(num)\n    num += 1\n
\n'
```

In [143…
```
# C
```

In [144…
```
"""num = 2
while num <= 100:
    is_prime = True
    for i in range(2, num):
        if num%i == 0:
            is_prime = False
            break
    if is_prime == True:
        print(num)"""
```

Out[144…
```
'num = 2\nwhile num <= 100:\n    is_prime = True\n    for i in range(2,
num):\n        if num%i == 0:\n            is_prime = False\n
break\n    if is_prime == True:\n        print(num)'
```

In [145…
```
# Solución
# D
```

Question 25 (Data Types)

What is the expected output of the following code?

```
print('x', 'y', 'z', sep='sep')
```

A. `x y z`

B. `xsepysepzsep`

C. `xsepysepz`

D. `xyz`

Solution 25

In [147…
```
print('x', 'y', 'z', sep='sep')
# xsepysepz
```

```
xsepysepz
```

In [148…
```
# Solución
# C
```

Question 26 (Functions)

What is the expected output of the following code?

```
def func():
    print(x + 1, end=' ')

x = 1
func()
print(x)
```

A. `2 1`

B. `1 1`

C. `1 2`

D. `2 2`

Solution 26

In [150…
```
def func():
    print(x + 1, end=' ')


x = 1
func()      # x + 1  ===> 1 + 1 = 2
print(x)    # 2 1
```

```
2 1
```

In [151…
```
# Solución
# A
```

Question 27 (Data Aggregates)

What is the expected output of the following code?

```
list = ['Peter', 'Paul', 'Mary']

def list(data):
    del data[1]
    data[1] = 'Jane'
    return data

print(list(list))
```

A. The code is erroneous

B. `['Peter', 'Jane']`

C. `['Peter', 'Jane', 'Mary']`

D. `['Paul', 'Mary', 'Jane']`

Solution 27

In [153…
```
# ESTE CÓDIGO ES UNA CHAPUZA
# solución que si funcionaría en celda más abajo

"""
list = ['Peter', 'Paul', 'Mary']

def list(data):
    del data[1]
    data[1] = 'Jane'
    return data

print(list(list))
"""

# TypeError: 'function' object does not support item deletion
```

Out[153…
```
"\nlist = ['Peter', 'Paul', 'Mary']\n \ndef list(data):\n    del data[1]
\n    data[1] = 'Jane'\n    return data\n \nprint(list(list))\n"
```

In [154…
```
# Solución
# A
```

ejemplo simple de código que si funciona

In [155…
```
data = ['Peter', 'Paul', 'Mary']

def funcion_listado(data):
    # data original :    # data = ['Peter', 'Paul', 'Mary']
    del data[1]          # data = ['Peter',       , 'Mary']   (borrado el
    data[1] = 'Jane'     # data = ['Peter',       , 'Jane']   (cambiado e
    return data          # data = ['Peter', 'Jane']           (obviamente

print(funcion_listado(data))
```

```
['Peter', 'Jane']
```

Question 28 (Data Aggregates)

What snippet would you insert in the line indicated below to print

The highest number is `10` and the lowest number is `1.` to the monitor?

```
data = [10, 2, 1, 7, 5, 6, 4, 3, 9, 8]
# insert your code here
print(
    ('The highest number is {} ' +
     'and the lowest number is {}.').format(high, low)000
)
```

A.

```
def find_high_low(nums):
    nums.sort()
    return nums[-1], nums[0]

high, low = find_high_low(data)
```

B.

```
def find_high_low(nums):
    nums.sort()
    return nums[len(nums)], nums[0]

high, low = find_high_low(data)
```

C.

```
None of the above
```

D.

```
def find_high_low(nums):
    nums.sort()
    return nums[0], nums[-1]

high, low = find_high_low(data)
```

Solution 28 (Basics)

In [157…

```
# A
```

In [158…
```
# The highest number is 10 and the lowest number is 1. to the monitor?



data = [10, 2, 1, 7, 5, 6, 4, 3, 9, 8]
# insert your code here
# A.
def find_high_low(nums):
    nums.sort()                     # por defecto: reverse=False ==> list
    return nums[-1], nums[0]        # maximo el de index -1, minimo el de

high, low = find_high_low(data)

print(
    ('The highest number is {} ' +
     'and the lowest number is {}.').format(high, low)
)
```

```
The highest number is 10 and the lowest number is 1.
```

In [159…
```
# Solución
# A
```

Question 29 (Basics)

What is true about compilation?

(Select two answers)

A. The code is converted directly into machine code executable by the processor

B. Both you and the end user must have the compiler to run your code

C. It tends to be slower than interpretation

D. It tends to be faster than interpretation

Solution 29

In [161…
```
# Solución
# A y D
```

Question 30 (Functions)

What is the expected output of the following code?

```
def func(a, b):
    return a ** a
```

```
print(func(2))
```

A. `2`

B. `None`

C. The code is erroneous

D. `4`

Solution 30

In [163…
```
"""
def func(a, b):
    return a ** a

print(func(2))
"""

# TypeError: func() missing 1 required positional argument: 'b'
```

Out[163… `'\ndef func(a, b):\n    return a ** a\n \nprint(func(2))\n'`

In [164…
```
# Solución
# C
```

Gracias por la atención

Isabel Maniega