Contenido creado por: Isabel Maniega

Question 1 (Data Aggregates)

What is the expected output of the following code?

```
data = ((1, 2),) * 7
print(len(data[3:8]))
```

A. `6`

B. The code is erroneous

C. `5`

D. `4`

Solution 1

In [2]:
```
data_inicial = ((1, 2),)
print('data_inicial:', data_inicial)
data = ((1, 2),) * 7
print('data:', data)
print('len(data):', len(data))
print('(data[3:8]):', (data[3:8]))
print('len(data[3:8]):', len(data[3:8]))
# otro ejemplo:
print('(data[3:20]):', (data[3:20]))
print('len(data[3:20]):', len(data[3:20]))
# y otro ejemplo mas:
print('(data[3:]):', (data[3:]))
print('len(data[3:]):', len(data[3:]))


# coge del 3 en adelante
```

```
data_inicial: ((1, 2),)
data: ((1, 2), (1, 2), (1, 2), (1, 2), (1, 2), (1, 2), (1, 2))
len(data): 7
(data[3:8]): ((1, 2), (1, 2), (1, 2), (1, 2))
len(data[3:8]): 4
(data[3:20]): ((1, 2), (1, 2), (1, 2), (1, 2))
len(data[3:20]): 4
(data[3:]): ((1, 2), (1, 2), (1, 2), (1, 2))
len(data[3:]): 4
```

In [3]:
```
data = ((1, 2),) * 7
print(len(data[3:8]))
```

```
4
```

In [4]:
```
# Solución
# D
```

Question 2 (Control Flow)

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

Expected output: adam_smit

Code:

```
    for ch in "adam_smit@openedg.org":

        if ch == "@":
            # insert code here

        print(ch, end="")
```

A. `break`

B. `continue`

C. `exit`

D. `print()`

Solution 2

In [6]:
```
# Code:

for ch in "adam_smit@openedg.org":
    if ch == "@":
        # insert code here
        break

    print(ch, end="")

# Expected output:
# adam_smit
```

```
adam_smit
```

In [7]:
```
# Solución
# A
```

Question 3 (Data Aggregates)

How would you remove all the items from the d dictionary?

Expected output:

```
{}
```

Code:

```
d = {'A' : 1, 'B' : 2, 'C' : 3}
```

A. `d.remove()`

B. `d.clear()`

C. `d.del()`

D. `del d`

Solution 3

In [9]:
```
# Code:
d = {'A' : 1, 'B' : 2, 'C' : 3}
print('d inicial:', d)

d.clear()
print('d después de clear:', d)

# Expected output:
# {}
```

```
d inicial: {'A': 1, 'B': 2, 'C': 3}
d después de clear: {}
```

In [10]:
```
# Solución
# B
```

Question 4 (Data Aggregates)

What is the expected output of the following code?

```
data = [1, 2, 3, None, (), [], ]
print(len(data))
```

A. `6`

B. `3`

C. `4`

D. `5`

Solution 4

In [12]:
```
data = [1, 2, 3, None, (), [], ]
print(len(data))
```

```
6
```

In [13]:
```
# demostración
data = [1, 2, 3, None, (), [], ]
for i in range(len(data)):
    print('i <0a5> es:', i, 'data[i]:', data[i])
```

```
i <0a5> es: 0 data[i]: 1
i <0a5> es: 1 data[i]: 2
i <0a5> es: 2 data[i]: 3
i <0a5> es: 3 data[i]: None
i <0a5> es: 4 data[i]: ()
i <0a5> es: 5 data[i]: []
```

In [14]:
```
# Solución
# A
```

Question 5 (Data Types)

You want to print the sum of two number.

What snippet would you insert in the line indicated below:

```
x = input('Enter the first number: ')
y = input('Enter the second number: ')

# insert your code here
```

A. `print('The Result is ' + (int(x + y)))`

B. `print('The Result is ' + (int(x) + int(y)))`

C. `print('The Result is ' + str(int(x + y)))`

D. `print('The Result is ' + str(int(x) + int(y)))`

Solution 5

In [16]:
```
"""
x = input('Enter the first number: ')
```

```
y = input('Enter the second number: ')

# A
print('The Result is ' + (int(x + y)))
"""
# TypeError: can only concatenate str (not "int") to str
```

Out[16]:
```
"\nx = input('Enter the first number: ')\ny = input('Enter the second nu
mber: ')\n\n# A\nprint('The Result is ' + (int(x + y)))\n"
```

In [17]:
```
# 10 y 20 por ejemplo
x = input('Enter the first number: ')
y = input('Enter the second number: ')

# D
print('The Result is ' + str(int(x) + int(y)))
```

```
Enter the first number: 10
Enter the second number: 20
The Result is 30
```

In [18]:
```
# Solución
# D
```

Question 6 (Functions)

What is the expected output of the following code?

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

print(func(func(2)) + 1)
```

A. `None`

B. The code is erroneous

C. `2`

D. `1`

Solution 6

In [20]:
```
"""
def func(x):
    if x % 2 == 0:
        return 1
    else:
        return         # NO CONOCE EL RETORNO
```

```
print(func(func(2)) + 1)
"""
#             1
#     func(1)       + 1
#  no sabe el retorno
# TypeError:
# unsupported operand type(s) for +: 'NoneType' and 'int'
```

Out[20]: '\ndef func(x):\n   if x % 2 == 0:\n   return 1\n   else:\n return   # NO CONOCE EL RETORNO\n \n \nprint(func(func(2)) + 1)\n'

In [21]:
```
def func(x):
    if x % 2 == 0:
        return 1
    else:
        return 5


print(func(func(2)) + 1)

#             1
#     func(1)       + 1
#       5           + 1
#           6
```

6

In [22]:
```
# Solución
# B
```

Question 7 (Basics)

You have the following file.

index.py:

```
from sys import argv
print(argv[1] + argv[2])
```

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

```
python index.py 42 3
```

What is the expected oputput?

A. `45`

B. `424242`

C. `126`

D. `423`

E. The code is erroneous

Solution 7

In [24]:
```
# probado en index2
# 423
```

In [25]:
```
# Solución
# D
```

Question 8 (Control Flow)

What is the expected output of the following code?

```
data = [1, 2, [3, 4], [5, 6], 7, [8, 9]]
count = 0

for i in range(len(data)):
    if type(data[i]) == list:
        count += 1

print(count)
```

A. The code is erroneous

B. `9`

C. `3`

D. `6`

Solution 8

In [27]:
```
data = [1, 2, [3, 4], [5, 6], 7, [8, 9]]
count = 0
print('len(data):', len(data))

for i in range(len(data)):       # len(data) = 6
    if type(data[i]) == list:    # 3 de ellos
        count += 1               # 3

print('count:', count)
```

```
len(data): 6
count: 3
```

In [28]:
```
# Solución
# C
```

Question 9 (Basics)

ASCII is:

A. a standard Python module name

B. short for American Standard Code for Information Interchange

C. a character name

D. a predefined Python variable name

Solution 9

In [30]:
```
# short for American Standard Code for Information Interchange
```

In [31]:
```
# Solución
# B
```

Question 10 (Data Aggregates)

A data structure described as LIFO is actually a:

A. list

B. heap

C. stack

D. tree

Solution 10

In [33]:
```
# LIFO: stack
```

In [34]:
```
# Solución
# C
```

Question 11 (Data Aggregates)

What is the expected output of the following code?

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

A. `(4)`

B. `44`

C. `(4,)`

D. `4`

Solution 11

In [36]:
```
data = (1, 2, 4, 8)
print('data inicial:', data)
data = data[-2:-1]
print('despues de data[-2:-1]:', data)
data = data[-1]
print('despues de data[-1]:', data)
```

```
data inicial: (1, 2, 4, 8)
despues de data[-2:-1]: (4,)
despues de data[-1]: 4
```

In [37]:
```
# Solución
# D
```

Question 12 (Data Aggregates)

The fact that tuples belong to sequence types means:

A. they can be modified using the del instruction

B. they can be indexed and sliced like lists

C. they can be extended using the `.append()` method

D. they are actually lists

Solution 12

In [39]:
```
# A, C y D no son porque las tuplas NO se pueden modificar
# Y NO SON LISTAS
```

In [40]:
```
# Solución
# B
```

Question 13 (Operators)

What is the expected output of the following code?

```
x = 1
print(++++x)
```

A. `3`

B. `4`

C. `1`

D. `2`

Solution 13

In [42]:
```
x = 1
print(++++x)
```

```
1
```

In [43]:
```
# Solución
# C
```

Question 14 (Error Handling)

An assertion can be used to:

A. Stop the program when some data have improper values

B. Make the Programmer more assertive

C. Import a module

Solution 14

In [45]:
```
assert 2==2
```

In [46]:
```
"""
assert 2==3
```

```
"""

# AssertionError:
```

Out[46]: '\nassert 2==3\n'

In [47]:
```
# Solución
# A
```

Question 15 (Data Aggregates)

What is the output of the following snippet?

```
my_list = [1, 2]

for v in range(2):
    my_list.insert(-1, my_list[v])

print(my_list)
```

A. `[1, 1, 2, 2]`

B. `[1, 1, 1, 2]`

C. `[1, 2, 1, 2]`

D. `[1, 2, 2, 2]`

Solution 15

In [49]:
```
my_list = [1, 2]

for v in range(2):                  # range(2) => 0,1
    my_list.insert(-1, my_list[v])
    # index -1 insert list[0]=1  ==> 1, 1, 2
    # index -1 insert list[1]=1  ==> 1, 1, 1, 2

print(my_list)  # 1, 1, 1, 2
```

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

In [50]:
```
# Solución
# B
```

Question 16 (Control Flow)

What is the expected output of the following code?

```
marks = [80, 70, 90, 90, 80, 100]

average = sum(marks) // len(marks)

grade = ''

if 90 <= average <= 100:
    grade = 'A'
elif 80 <= average < 90:
    grade = 'B'
elif 70 <= average < 80:
    grade = 'C'
elif 65 <= average < 70:
    grade = 'D'
else:
    grade = 'F'

print(grade)
```

A. `D`

B. The code is erroneous

C. `B`

D. `F`

E. `A`

F. `C`

Solution 16

In [52]:
```
marks = [80, 70, 90, 90, 80, 100]

print('sum(marks):', sum(marks))    # 510
print('len(marks):', len(marks))    # 6

average = sum(marks) // len(marks)

print('average:', average)  # 510/6 => entre 80 y 90


grade = ''

if 90 <= average <= 100:
    grade = 'A'
elif 80 <= average < 90:    # B
    grade = 'B'
elif 70 <= average < 80:
    grade = 'C'
elif 65 <= average < 70:
    grade = 'D'
else:
    grade = 'F'
```

```
print(grade)  # B
```

```
sum(marks): 510
len(marks): 6
average: 85
B
```

In [53]:
```
# Solución
# C
```

Question 17 (Basics)

What is IDLE?

A. An acronym that stands for Interactive Development and Learning extension

B. A version of Python

C. An acronym that stands for Integrated Development and Learning Environment for Python

Solution 17

In [55]:
```
# IDLE is Python's Integrated Development and Learning Environment.
```

In [56]:
```
# Solución
# C
```

Question 18 (Functions)

A built-in function is a function which ...

A. is hidden from programmers

B. has been placed within your code by another programmer

C. comes with Python and is an integral part of Python

D. has to be imported before use

Solution 18

In [58]:
```
# comes with Python and is an integral part of Python
```

In [59]:
```
# Solución
# C
```

Question 19 (Basics)

What is CPython?

A. Another name for CPython, a superset of the Python programming language

B. A compiled language used to perform high-level programming functions

C. The default implementation of the Python programming language

Solution 19

In [61]:
```
# Solución
# C
```

Question 20 (Data Aggregates)

What is the expected output of the following code?

```
data = {'one': 'two', 'two': 'three', 'three': 'one'}
res = data['three']

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

print(res)
```

A. `two`

B. `three`

C. `one`

D. `('one', 'two', 'three')`

Solution 20

In [63]:
```
data = {'one': 'two',
        'two': 'three',
        'three': 'one'}
```

```
res = data['three']           # res = one

for _ in range(len(data)):    # len(data) = 3
    res = data[res]
    # res = data[one]    =>  res = two
    # res = data[two]    =>  res = three
    # res = data[three]  =>  res = one


print(res)   # one
```

one

In [64]:
```
# Solución
# C
```

Question 21 (Operators)

What is the expected output of the following code?

```
list1 = [3, 7, 23, 42]
list2 = [3, 7, 23, 42]
print(list1 is list2)
print(list1 == list2)
```

A.

```
True
True
```

B.

```
False
True
```

C.

```
False
False
```

D.

```
True
False
```

Solution 21

In [66]:
```
list1 = [3, 7, 23, 42]
list2 = [3, 7, 23, 42]
```

```
print('list1:', list1)
print('list2:', list2)

print('list1 is list2 :')
print(list1 is list2)

print('list1 == list2 :')
print(list1 == list2)
```

```
list1: [3, 7, 23, 42]
list2: [3, 7, 23, 42]
list1 is list2 :
False
list1 == list2 :
True
```

In [1]:
```
# Solución
list1 = [3, 7, 23, 42]
list2 = [3, 7, 23, 42]
print(id(list2))
print(id(list1))
# Presentan distintas posiciones en memoria, por lo tanto no es
# una lista, es otra, pero si son iguales
print(list1 is list2)
print(list1 == list2)
```

```
140296126035776
140296126375680
False
True
```

In [2]:
```
# Solución
list1 = [3, 7, 23, 42]
list2 = list1
print(id(list2))
print(id(list1))
# Presentan iguales posiciones en memoria, por lo tanto,
# una lista es la otra y son iguales
print(list1 is list2)
print(list1 == list2)
```

```
140296126035072
140296126035072
True
True
```

In [67]:
```
# Solución
# B
```

Question 22 (Control Flow)

Consider the following code.

```
nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
x = 0
```

```
while x < 10:         # Line 3
    print(nums(x))    # Line 4
    if nums[x] = 7:   # Line 5
        break         # Line 6
    else:             # Line 7
        x += 1        # Line 8
```

You want to print the numbers 1 to 7 to the monitor.

But the code does not work.

What do you have to change?

Choose two.

A. `print(nums[x])` # Line 4

B. `while(x<10):` # Line 3

C. `if nums[x]==7:` # Line 5

D. `x = x + 1` # Line 8

Solution 22

In [69]:
```
nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
x = 0

while x < 10:          # Line 3
    print(nums[x])     # Line 4  (Opción A)
    if nums[x] == 7:   # Line 5  (Opción C)
        break          # Line 6
    else:              # Line 7
        x += 1         # Line 8

# Imprimir números de 1 a 7 en el monitor

# x+=1 equivalente a x=x+1 ============> D descartado
# While puede o no tener paréntesis ==> B descartado
```

```
1
2
3
4
5
6
7
```

In [70]:
```
# Solución
# A y C
```

Question 23 (Operators)

What is the expected output of the following code?

```
x = 2

y = 6

x += 2 ** 3

x //= y // 2 // 3

print(x)
```

A. `10`

B. `0`

C. `11`

D. `9`

Solution 23

In [72]:
```
# ejemplos previos
```

In [73]:
```
6//2//3
# va de derecha a izquierda
# diferente que como venia siendo
```

Out[73]: 1

In [74]:
```
# ejemplo del ejercicio
```

In [1]:
```
x = 2
y = 6

x += 2 ** 3          # x = x + 2**3
                     # x = 2 + 8
                     # x = 10

x //= y // 2 // 3    # x = x // y // 2 // 3
# x = 10 // (6 // 2) // 3)
# x = 10 // (3 // 3)
print(x) # 10 // 1 ==> 10
```

```
10
```

In [76]:
```
# Solución
# A
```

Question 24 (Functions)

Which of the following lines properly starts a parameterless function definition?

A. `def fun():`

B. `fun function():`

C. `def fun:`

D. `function fun():`

Solution 24

In [78]:
```
def fun():
    pass
```

In [79]:
```
# Solución
# A
```

Question 25 (Data Aggregates)

What is the expected output of the following code?

```
data = {'name': 'Peter', 'age': 30}
person = data.copy()
print(id(data) == id(person))
```

A. `False`

B. `True`

C. `1`

D. `0`

Solution 25

In [81]:
```
# The id() method returns a unique integer (identity)
# of a passed argument object.
```

In [82]:
```
# id of 1
print("id de 1 =", id(1))


a = 10
```

```
# id de a
print("id of a =", id(a))


b = a
# id de b
print("id de b =", id(b))


c = 20.0
# id de c
print("id de c =", id(c))
```

```
id de 1 = 9793088
id of a = 9793376
id de b = 9793376
id de c = 140050568969136
```

In [83]:
```
data = {'name': 'Peter',
        'age': 30}
id(data)
```

Out[83]: 140050568992576

In [84]:
```
data = {'name': 'Peter',
        'age': 30}

person = data.copy()

print(id(data) == id(person))
```

```
False
```

In [85]:
```
# Solución
# A
```

Question 26 (Functions)

What is the expected output of the following code?

```
x = 42

def func():
    global x
    print('1. x:', x)
    x = 23
    print('2. x:', x)

func()
print('3. x:', x)
```

A.

```
1. x: 42
2. x: 23
3. x: 23
```

B.

```
None of the above
```

C.

```
1. x: 42
2. x: 23
3. x: 42
```

D.

```
1. x: 42
2. x: 42
3. x: 42
```

Solution 26

In [87]:
```
x = 42

def func():
    global x
    print('1. x:', x)   # 42
    x = 23
    print('2. x:', x)   # 23

func()                  # 42, 23
print('3. x:', x)       # 23
```

```
1. x: 42
2. x: 23
3. x: 23
```

In [88]:
```
# Solución
# A
```

Question 27 (Basics)

A keyword is a word:

(Select two answers)

A. that cannot be used as a function name

B. that cannot be used as a variable name

C. is the most important word in the whole program

Solution 27

In [90]:
```
# C no es
```

In [91]:
```
# Solución
# A y B
```

Question 28 (Operators)

The result of the following division:

```
1 / 1
```

A. cannot be predicted

B. cannot be evaluated

C. is equal to `1.0`

D. is equal to `1`

Solution 28

In [93]:
```
1 / 1
```

Out[93]:
```
1.0
```

In [94]:
```
# Solución
# C
```

Question 29 (Data Aggregates)

What is the output of the following code?

```
my_list = [3, 1, -1]
my_list[-1] = my_list[-2]
print(my_list)
```

A. `[3, -1, 1]`

B. `[3, 1, 1]`

C. `[1, 1, -1]`

Solution 29

In [96]:
```
my_list = [3, 1, -1]

my_list[-1] = my_list[-2]   # my_list[-1] = 1

print(my_list)              # [3, 1, 1]
```

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

In [97]:
```
# Solución
# B
```

Question 30 (Data Aggregates)

What is the output of the following snippet?

```
tup = (1, ) + (1, )
tup = tup + tup
print(len(tup))
```

A. the snippet is erroneous (invalid syntax)

B. `4`

C. `2`

Solution 30

In [99]:
```
tup = (1, ) + (1, )
print('tupla inicial:', tup)    # (1, 1)
tup = tup + tup                 # (1, 1, 1, 1)
print('tup + tup:', tup)
print('len(tup):', len(tup))    # 4
```

```
tupla inicial: (1, 1)
tup + tup: (1, 1, 1, 1)
len(tup): 4
```

In [100…
```
# Solución
# B
```

Gracias por la atención

Isabel Maniega