Preguntas 'tipo' en la certificación PCAP:

You are going to read just one character from a stream called s. Which statement would you use?

- ch = read(s, 1)
- ch = s. input(1)
- ch = input(s, 1)
- ch = s. read(l)

This statement uses the read() method of the stream object s with an argument of 1, indicating that it will read one character from the stream.

What can you deduce from the following statement0 (Select two answers) str = open('file.txt', "rt") ?

- [x] **str is a string read in from the file named file. txt**
- [ ] **a newlina character translation will be performed during the reads**
- [x] **if file. txt does not exist, it will be created**
- [ ] **the opened file cannot be written with the use of the str variable**

From the statement str = open('file.txt', "rt"), we can deduce the following:

If file.txt does not exist, it will be created: This is true.
If the file named file.txt does not exist, it will be created in read-text mode ("rt").

The opened file cannot be written with the use of the str variable: This is also true.
The variable str is assigned the file object returned by the open() function,
which opened the file in read-text mode ("rt").
Therefore, attempting to write to this file using str would result in an error.

Python's built-in function named open () tries to open a file and returns ?

- ○ an integer value identifying an opened file
- ○ an error code (0 means success)
- ○ a stream object
- ○ always None

Python's built-in function open() tries to open a file and returns a stream object.
Therefore, the correct answer is:

a stream object: The open() function returns a file object,
which represents the stream of data from the file that has been opened.
This object can then be used for reading or writing operations on the file.

Assuming that String is six or more letters long, the following slice String [1:-2] is shorter than original string by?

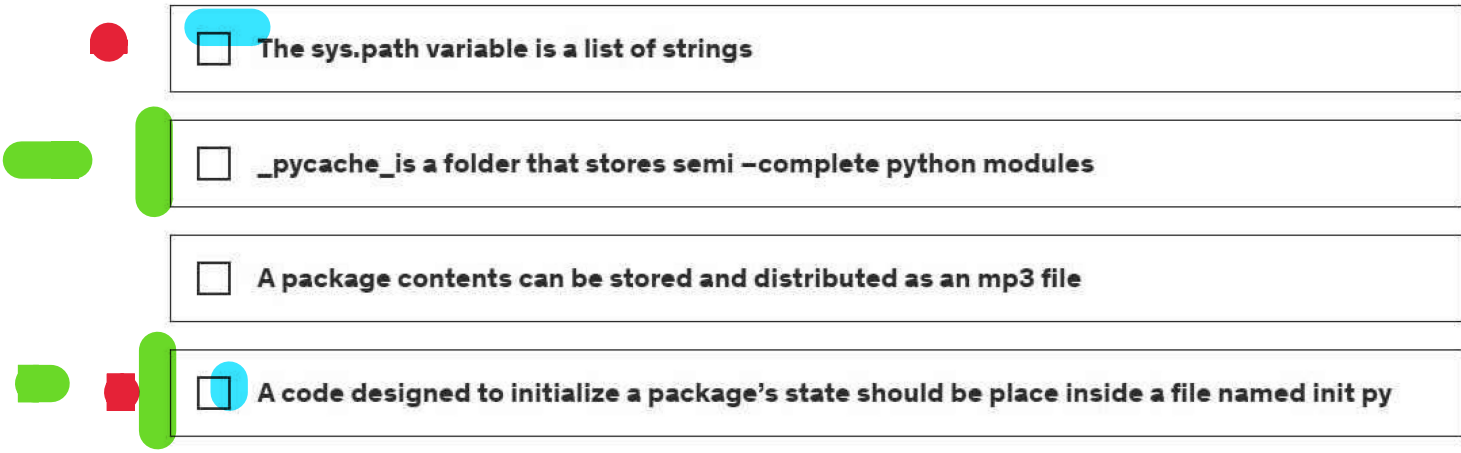
- Four chars
- Three chars
- One char
- Two chars

The slice String[1:-2] removes the first character and the last two characters from the original string.
Removing the first character reduces the length by 1.
Removing the last two characters reduces the length by 2.
So, in total, the resulting string will be shorter by 3 characters.

Therefore, the correct answer is:

Three chars: The resulting slice will be shorter by three characters.

What is true about python package? (Select two answers) ?



The sys.path variable is a list of strings:
This is correct.
sys.path is a list of directory names where Python searches for modules.
It can be modified to include additional directories.

A code designed to initialize a package's state should be placed inside a file named __init__.py:
This is correct.
The __init__.py file is a special Python file that is executed when a package is imported.
It is commonly used to initialize package-level variables, perform setup tasks,
or define package-level functions and classes.

Therefore, the correct answers are:

The sys.path variable is a list of strings.
A code designed to initialize a package's state should be placed inside a file named __init__.py.

The following expression

2 ** 3 ** 2 **1

Is :

- ◯ **Invalid**
- ◯ **Equal to 256**
- ◯ **Equal to 512**
- ◯ **Equal to 64**
- ◯ **Equal to 128.0**

e need to remember that the exponentiation operator (**) is right-associative, meaning it's evaluated from right to left.

So, let's break it down step by step:

2 ** 1 equals 2.
2 ** 2 equals 4.
3 ** 4 equals 81.
Now, we evaluate:

Calculating 2 ** 81 results in an extremely large number.

However, since none of the provided options match this very large number, the expression 2 ** 3 ** 2 ** 1 should be considered as "invalid" given that it results in an excessively large value. Therefore, the correct choice is "invalid".

if you want to build a string that reads:

Peters 's sister ‘s name ‘s “Anna”

Which of the following liberals would you use? (Select all that apply)

- [x] **“Peter ‘s sister ‘s name ‘s \”Anna \ “”**
- [x] **‘Peter ‘s sister ‘s name ‘s \”Anna \ “ ‘**
- [ ] **“Peter ‘s sister ‘s name ‘s ”Anna “”**
- [ ] **‘Peter ‘s sister ‘s name ‘s ”Anna “ ‘**

To construct the desired string, you need to handle the apostrophes and double quotes appropriately.

Here are the options with the correct escaping:

"Peter's sister's name's \"Anna\""
This option correctly escapes the double quotes within the string.

What is the expected output of the following snippet?

```
I = 250
While len (str(i) ) > 72;
I *= 2
Else:
I // =2
Print(i)
```

- 125
- 250
- 400
- 75

What snippet would you insert in the line indicated below:

```
n = 0
while n < 4:
n += 1
# insert your code here
```

to print the following string to the monitor after the loop finishes its execution:

1 2 3 4

- ○ **print(n)**
- ◉ **print(n, sep=" ")**
- ○ **print(n, end=" ")**
- ○ **print(n, " ")**

If you want to print the values of n on a single line separated by spaces, you can use the end parameter of the print() function to specify an alternative string to print at the end, instead of the default newline character.

What is the value type returned after executing the following snippet?

```
x = 0
y = 2
z = len("Python")
x = y > z
print(x)
```

- (●) **int**
- ( ) **float**
- ( ) **str**
- ( ) **bool**
- ( ) **NoneType**

Let's analyze the snippet step by step:

x = 0: Here, x is assigned the integer value 0.
y = 2: Here, y is assigned the integer value 2.
z = len("Python"): Here, z is assigned the length of the string "Python", which is 6.
x = y > z: Here, y > z evaluates to False (since 2 is not greater than 6), and this Boolean value (False) is ass
print(x): Here, the value of x (which is False) is printed.

Given that x is assigned a Boolean value (False) and the result of print(x) is False,
the value type returned after executing the snippet is:

>> bool

What will the final value of the Val variable be when the following snippet finishes its execution?

```
Val = 1
Val2 = 0
Val = Val ^ Val2
Val2 = Val ^ Val2
Val = Val ^ Val2
```

- ◯ 0
- ◯ 1
- ◯ 2
- ◯ 4

Val = str(1) ^ str(0): This performs a bitwise XOR operation between the string representations of the integers 1 and 0. The string representations of these integers are "1" and "0", respectively.

Performing XOR operation character by character:

'1' ^ '0' results in the string '1'.
So, Val becomes '1'.

Val2 = str(0) ^ str(1): This performs a bitwise XOR operation between the string representations of the integers 0 and 1.
The string representations of these integers are "0" and "1", respectively.

Performing XOR operation character by character:

'0' ^ '1' results in the string '1'.
So, Val2 becomes '1'.

Val = str(1) ^ str(1): This performs a bitwise XOR operation between the string representations of the integers 1 and 1.
The string representations of these integers are both "1".

Performing XOR operation character by character:

'1' ^ '1' results in the string '0'.
So, Val becomes '0'.

Therefore, the final values of Val and Val2 after executing the snippet are '0' and '1', respectively.

Which line can be used instead of the comment to cause the snippet to produce the following expected output? (Select all that apply)

Code:

```
z, y, x = 2, 1, 0
x, z = z, y
y = y - z
# put line here
print(x, y, z)
```

Expected output:

```
0, 1, 2
```

- [ ] **x, y, z = y, z, x**
- [x] **z, y, x = x, z, y**
- [ ] **y, z, x = x, y, z**
- [ ] **The code is error**

this line will swap the values of z, y, and x, effectively assigning x to the original value of Z, z to the original value of y, and y to the original value of x.

Inserting this line after the line y = y - z would correctly swap the values and lead to the expected output "0, 1, 2". So, this line can be used instead of the comment to produce the expected output.

z, y, x = x, z, y

What is the expected output of the following snippet?

```
a = 0
b = a ** 0
if b < a + 1:
c = 1
elif b == 1:
c = 2
else:
c = 3
print(a + b + c)
```

- ○ 1
- ○ 2
- ◉ 3
- ○ The code is erroneous

Let's break down the code snippet step by step:

a = 0: a is assigned the value 0.

b = a ** 0: b is assigned the result of a raised to the power of 0.

Since any number raised to the power of 0 is 1, b becomes 1.

The if statement checks conditions based on the value of b:
if b < a + 1:: Since b is 1 and a is 0, the condition b < a + 1 evaluates to 1 < 1, which is False.
elif b == 1:: This condition checks if b is equal to 1, which it is. So, this condition is True.
Since the first condition is False and the second condition is True, the else block is skipped.

c = 2: Since the condition b == 1 is True, c is assigned the value 2.

Finally, print(a + b + c) prints the sum of a, b, and c.

Now, let's calculate the values:

a = 0
b = 1
c = 2
So, a + b + c = 0 + 1 + 2 = 3.

Therefore, the expected output of the snippet is 3
.

How many stars (*) does the following snippet print?

```
i = 10
while i > 0 :
i -= 3
print("*")
if i <= 3:
break
else:
print("*")
```

- ○ three
- ○ two
- ○ one
- ◉ The code is erroneous

Let's analyze the given snippet:

Initialize the variable i with the value 10.

Enter a while loop:
Decrement i by 3 in each iteration (i -= 3).
Print a single asterisk ("*").
Check if i is less than or equal to 3:
If true, exit the loop (break).
Otherwise, print another asterisk ("*").
Let's trace the loop:

Iteration 1: i becomes 7. Print one asterisk.
Iteration 2: i becomes 4. Print one asterisk.
Iteration 3: i becomes 1. Print one asterisk.
At this point, i is not greater than 0, so the loop exits.

So, the snippet prints 3 asterisks (*).

Therefore, the snippet prints three stars (*).

How many lines does each of the following code examples output when run separately ?

```
# Example 1
for i in range(1, 4, 2):
print("*")
# Example 2
for i in range(1, 4, 2):
print("*", end="")
# Example 3
for i in range(1, 4, 2):
print("*", end="**")
# Example 4
for i in range(1, 4, 2):
print("*", end="**")
print("***")
```

- ○ **Example 1: two, Example 2: one, Example 3: one, Example 4: one**
- ○ **Example 1: two, Example 2: one, Example 3: one, Example 4: two**
- ○ **Example 1: two, Example 2: one, Example 3: two, Example 4: three**
- ◉ **Example 1: one, Example 2: one, Example 3: one, Example 4: two**

Which of the following statements are true? (Select all that apply)

- [ ] **UNICODE is the name of an operating system**
- [ ] **UTF-8 is the name of a data transmission device**
- [x] ***The Python Language Reference* is the official reference manual that describes the syntax and semantics of the Python language**
- [ ] **ASCII is an acronym for *Automatic Systems of Computer Inner Interoperability***
- [ ] **Python strings are immutable, which means they cannot be sliced**
- [x] **Lists and strings in Python can be sliced**

What is the result of the following comparison?

```
x = "20"
y = "30"
print(x > y)
```

- ○ **True**
- ◉ **False**
- ○ **None**
- ○ **The comparison causes a runtime exception/error**

What is the expected output of the following snippet?

```
s = "Hello, Python!"
print(s[-14:15])
```

- ◯ **Hello, Python!**
- ◯ **!nohtyP ,olleH**
- ◯ **Hello, Python!Hello, Python!**
- ◯ **The program causes a runtime exception/error**
- ◉ **The result cannot be predicted**

- ◯ **Hello, Python!** (Correcto)
- ◯ **!nohtyP ,olleH**
- ◯ **Hello, Python!Hello, Python!**
- ◯ **The program causes a runtime exception/error**
- ◯ **The result cannot be predicted**

- [x] [2, 4] (Correcto)
- [ ] ['C', 2, 4]
- [ ] ['B', 'C', 2, 4]
- [ ] ['A', 'B']

- [x] a
  b
  c
  (Correcto)

- [ ] 1
  2
  3

- [ ] a:1
  b:2
  c:3

- [ ] The code is erroneous