

1. Can we use the "else" block for **for** loop?

for example:

```
for i in range(1, 5):  
    print(i)  
else:  
    print("this is else block statement" )
```

- No

- **Yes**

Explanation

We can use the else block after the end of [for loop](#) and [while loop](#). The else block is used to check the successful execution of a loop. If the loop executed successfully without any issues, the else block executes.

2. What is the output of the following code?

```
var1 = 1
var2 = 2
var3 = "3"

print(var1 + var2 + var3)
```

- 6
- 33
- 123

- **Error. Mixing operators between numbers and strings are not supported**

Explanation

We cannot add strings and numbers together using the `+` [operator](#). Either we can use the `+` operator to concatenate strings or add [numbers](#).

3. What is the output of the following

```
x = 36 / 4 * (3 + 2) * 4 + 2  
print(x)
```

Hint: [Python Operators Precedence](#)

- **182.0**

- 37
- 117
- The Program executed with errors

Explanation:

To choose the correct answer, You must know the operator precedence and associativity.

4. What is the Output of the following code?

```
for x in range(0.5, 5.5, 0.5):  
    print(x)
```

- [0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5]
- [0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5]

- **The Program executed with errors**

Explanation

We cannot use float numbers in `range()` function. Please refer to [How to generate a range of float numbers](#).

5. What is the output of the following code?

```
valueOne = 5 ** 2
valueTwo = 5 ** 3

print(valueOne)
print(valueTwo)
```

- 10
15

- **25**
125

- Error: invalid syntax

Explanation:

Using two multiplication symbols, we can make a power relationship in Python. We call `**` operator an exponent operator. For example, the result of expression `5 ** 3` is 125.

6. What is the output of the following code?

```
sampleSet = {"Jodi", "Eric", "Garry"}  
sampleSet.add(1, "Vicki")  
print(sampleSet)
```

- {'Vicki', 'Jodi', 'Garry', 'Eric'}
- {'Jodi', 'Vicki', 'Garry', 'Eric'}

- **The program executed with error**

Explanation:

The [set](#) is an unordered data structure. Therefore, we cannot access/add/remove its elements by index number.

7. What is the output of the following code?

```
listOne = [20, 40, 60, 80]
listTwo = [20, 40, 60, 80]

print(listOne == listTwo)
print(listOne is listTwo)
```

- True
True

- **True**
False

- False
True

Explanation

The `==` (Equal To) operator used to compare the values of two objects and
The `is` operator compares the identity of two objects.

8. The `in` operator is used to check if a value exists within an iterable object container such as a list. Evaluate to `True` if it finds a variable in the specified sequence and `False` otherwise.

- **True**

- False

9. What is the output of the following code?

```
for i in range(10, 15, 1):  
    print( i, end=', ')
```

Hint: [Python range function](#)

- **10, 11, 12, 13, 14,**

- 10, 11, 12, 13, 14, 15,

Explanation

Remember, the range doesn't include the stop number in the output. Read [Python range function](#) for more details.

10. What is the output of the following code?

```
str = "pynative"  
print (str[1:3])
```

- py

- **yn**

- pyn
- yna

Explanation

Remember, the index always starts from 0. Therefore, str [1 : 3] is "yn"

11. What is the output of the following code?

```
var= "James Bond"  
print(var[2::-1])
```

- Jam
- dno

- **maJ**

- dnoB semaJ

Explanation:

Pick a range of items starting in the reverse direction starting from index 2 with step 1.

12. What is the output of the following code?

```
sampleList = ["Jon", "Kelly", "Jessa"]  
sampleList.append(2, "Scott")  
print(sampleList)
```

- **The program executed with errors**

- ['Jon', 'Kelly', 'Scott', 'Jessa']
- ['Jon', 'Kelly', 'Jessa', 'Scott']
- ['Jon', 'Scott', 'Kelly', 'Jessa']

Explanation:

The `append()` method appends an item to the end of the [list](#). Therefore, we cannot pass the index number to it.

13. What is the output of the following code?

```
var = "James" * 2 * 3
print(var)
```

- **JamesJamesJamesJamesJamesJames**

- JamesJamesJamesJamesJames
- Error: invalid syntax

Explanation:

We can use `*` operator to repeat the string `n` number of times. For example, in the above question, First, we repeated the string two times, and again we repeated the output string three times.

14. A string is immutable in Python?

Every time when we modify the string, Python Always create a new String and assign a new string to that variable.

- **True**

- False

Explanation:

Yes, strings are immutable in Python. You cannot modify a string once created. If you change a string, Python builds a new string with the updated value and assigns it to the [variable](#).

Example:

```
str1 = "first"
id(str1)
str1 = str1+ " Two"
id(str1)
```

Output:

```
140560663354704
140560640152496
```

Earlier `str1` was pointing to memory address "140560663354704" now, it's pointing to "140560640152496" which means Python created a new string after you updated it.

15. What is the output of the following code?

```
def calculate (num1, num2=4):
    res = num1 * num2
    print(res)

calculate(5, 6)
```

Hint: [functions in Python](#)

- 20
- The program executed with errors

- **30**

Explanation

In Python, we can set default values for arguments. If the function is called without the argument, the default value is used.

16. Which operator has higher precedence in the following list

- % (Modulus)
- & (BitWise AND)

- **** (Exponent)**

- > (Comparison)

17. What is the output of the following code

```
salary = 8000

def printSalary():
    salary = 12000
    print("Salary:", salary)

printSalary()
print("Salary:", salary)
```

- **Salary: 12000 Salary: 8000**

- Salary: 8000 Salary: 12000
- The program failed with errors

Explanation:

If you define a variable with the same name inside the function and global scope, a function will refer to the local variable by default.

18. What is the output of the following code?

```
p, q, r = 10, 20 ,30  
print(p, q, r)
```

- 10 20

- **10 20 30**

- Error: invalid syntax

Explanation

In Python, We can do simultaneous assignments to more than one [variable](#).