

TYPE-A:STRING MANIPULATION:CH-10

- 1) Write a Python script that traverses through an input string and prints its characters in different lines — two characters per line.

```
sol:
    # Input string from user
    s = input("Enter a string: ")

    # Traverse through the string two characters at a time
    i = 0
    while i < len(s):
        # Print two characters per line
        print(s[i:i+2])
        i += 2
```

- 2) Out of the following operators, which ones can be used with strings in Python?
=, -, *, /, //, %, >, <>, in, not in, <=

```
sol:
    =, *, >, in, not in, <=
```

- 3) What is the result of following statement, if the input is 'Fun'? print(input("...") + "trial" + "Ooty" * 3)

```
sol:
    FuntrialOotyOotyOoty
```

- 4) Which of the following is not a Python legal string operation?

- (a) 'abc' + 'abc'
- (b) 'abc' * 3
- (c) 'abc' + .3
- (d) 'abc'.lower()

```
sol:
    (c) 'abc' + .3
```

- 5) Can you say strings are character lists? Why? Why not?

```
sol:
    Strings are not exactly character lists in Python. Although you can access individual characters of a string using indexing, strings are immutable, meaning their characters cannot be changed, added, or removed. Lists, on the other hand, are mutable, so their elements can be modified. Therefore, strings behave like sequences of characters but are not true lists.
```

6) Given a string `S = "CARPE DIEM"`. If `n` is `length/2` (length is the length of the given string), then what would following return?

- (a) `S[: n]`
- (b) `S[n :]`
- (c) `S[n : n]`
- (d) `S[1 : n]`
- (e) `S[n : length - 1]`

sol:

- (a) `S[:n]` → "CARPE" (from start to index 4)
- (b) `S[n:]` → "DIEM" (from index 5 to end)
- (c) `S[n:n]` → "" (empty string, start and end index are the same)
- (d) `S[1:n]` → "ARPE " (from index 1 to 4)
- (e) `S[n:length-1]` → "DIE" (from index 5 to 8)

7) From the string `S = "CARPE DIEM"`, which ranges return "DIE" and "CAR"?

sol:

- To get "DIE", use the range `S[6:9]` (index 6 to 8).
- To get "CAR", use the range `S[0:3]` (index 0 to 2).

8) What happens when from a string slice you skip the start and/or end values of the slice?

sol:

When you skip the start value in a string slice, Python assumes it as the beginning of the string (index 0). When you skip the end value, Python assumes it as the end of the string. This allows you to easily slice from the start or up to the end without specifying exact indexes.

9) What would the following expressions return?

1. `"Hello World".upper().lower()`
2. `"Hello World".lower().upper()`
3. `"Hello World".find("Wor", 1, 6)`
4. `"Hello World".find("Wor")`
5. `"Hello World".find("wor")`
6. `"Hello World".isalpha()`
7. `"Hello World".isalnum()`
8. `"1234".isdigit()`
9. `"123FGH".isdigit()`

sol:

1. `"Hello World".upper().lower()` → "hello world" (converted to uppercase first, then back to lowercase)
2. `"Hello World".lower().upper()` → "HELLO WORLD" (converted to lowercase first, then to uppercase)
3. `"Hello World".find("Wor", 1, 6)` → -1 (substring "Wor" not found between index 1 and 5)
4. `"Hello World".find("Wor")` → 6 (substring "Wor" starts at index 6)

- 5. "Hello World".find("wor") → -1 (case-sensitive search; "wor" not found)
- 6. "Hello World".isalpha() → False (string contains a space, not only letters)
- 7. "Hello World".isalnum() → False (string contains a space, not only letters/numbers)
- 8. "1234".isdigit() → True (all characters are digits)
- 9. "123FGH".isdigit() → False (string contains letters as well as digits)

10) Which functions would you choose to use to remove leading and trailing white spaces from a given string?

sol:

To remove leading and trailing white spaces from a string, you can use the strip() function. If you want to remove only leading spaces, use lstrip(), and for only trailing spaces, use rstrip().

11) Try to find out if for any case, the string functions isalnum() and isalpha() return the same result

sol:

The string functions isalnum() and isalpha() can return the same result only when the string contains letters only. In that case, isalpha() returns True because all characters are letters, and isalnum() also returns True because letters are considered alphanumeric.

12) Suggest appropriate functions for the following tasks:

1. To check whether the string contains digits
2. To find for the occurrence a string within another string
3. To convert the first letter of a string to upper case
4. to capitalize all the letters of the string
5. to check whether all letters of the string are in capital letters
6. to remove from right of a string all string-combinations from a given set of letters
7. to remove all white spaces from the beginning of a string

sol:

- 1.isdigit()
- 2.find()
- 3.capitalize()
- 4.upper()
- 5.isupper()
- 6.rstrip()
- 7.lstrip()

13) In a string slice, the start and end values can be beyond limits. Why?

sol:

In a string slice, the start and end values can be beyond the string limits because Python automatically adjusts out-of-range indexes to the nearest valid index. This prevents errors and allows slicing even if the specified indexes are too large or too small.

14) Can you specify an out of bound index when accessing a single character from a string? Why?

sol:

No, you cannot specify an out-of-bound index when accessing a single character from a string. If you try to do so, Python will raise an `IndexError` because unlike slicing, accessing a single character requires the index to be within the valid range of the string.

15) Can you add two strings? What effect does ' + ' have on strings?

sol:

Yes, you can add two strings in Python. The '+' operator concatenates the strings, meaning it joins them together to form a single string.