Unit – 3

1. What is recursion in Python?

A) A function calling itself
B) A function calling another function
C) A loop inside a function
D) A function without return statement

✅ **Answer:** A) A function calling itself

---

2. What is the base case in recursion?

A) The part where function returns itself
B) The condition where recursion ends
C) The first call of the function
D) A loop condition

✅ **Answer:** B) The condition where recursion ends

---

3. What happens if the base case is not defined in recursion?

A) Infinite loop
B) Error occurs
C) Function will execute once
D) Code will not compile

✅ **Answer:** A) Infinite loop

---

4. What is the maximum recursion depth in Python by default?

A) 100
B) 500
C) 1000
D) 2000

✅ **Answer:** C) 1000

---

5. Which module can be used to change the recursion limit in Python?

A) sys
B) os
C) math
D) random

✅ **Answer:** A) sys

---

6. What is the output of the following code?

```
def fun(n):
    if n == 0:
        return 1
    return n * fun(n-1)
print(fun(4))
```

A) 24
B) 10
C) 4
D) 120

✅ **Answer:** A) 24

---

7. Which of the following problems is solved using recursion?

A) Fibonacci Series
B) Factorial Calculation
C) Tower of Hanoi
D) All of the above

✅ **Answer:** D) All of the above

---

8. What is the output of the following code?

```
def fun(n):
    if n <= 0:
        return 0
    return n + fun(n-1)
```

```
print(fun(3))
```

A) 6
B) 3
C) 5
D) 4

✅ **Answer:** A) 6

---

9. What is the purpose of the base case in recursion?

A) To stop the recursive calls
B) To print the result
C) To increase recursion depth
D) To repeat the loop

✅ **Answer:** A) To stop the recursive calls

---

10. What is tail recursion?

A) When the recursive call is the last statement
B) When recursion happens at the start
C) When recursion uses two functions
D) When recursion happens in loops

✅ **Answer:** A) When the recursive call is the last statement

---

11. What is the output of the following code?

```
def fun(n):
    if n == 1:
        return 1
    return n * fun(n-1)
print(fun(5))
```

A) 15
B) 20
C) 120
D) 25

✅ **Answer:** C) 120

---

12. Which keyword is used to define a function in Python?

A) def
B) function
C) func
D) lambda

✅ **Answer:** A) def

---

13. What happens if recursion depth exceeds the limit in Python?

A) Memory error
B) RecursionError
C) TypeError
D) ValueError

✅ **Answer:** B) RecursionError

---

14. What is the smallest possible recursion in Python?

A) 1
B) 0
C) -1
D) None

✅ **Answer:** B) 0

---

15. What is the output of the following code?

```
def fun(n):
    if n <= 0:
        return n
    return fun(n-1)
print(fun(5))
```

A) 0
B) 5
C) 1
D) Error

✅ **Answer:** A) 0

---

16. What is indirect recursion?

A) When two or more functions call each other
B) When a function calls itself directly
C) When recursion happens without stopping
D) When recursion happens in loops

✅ **Answer:** A) When two or more functions call each other

---

17. What is the output of the following code?

```
def fun(n):
    if n == 0:
        return 1
    else:
        return n * fun(n-1)
print(fun(3))
```

A) 6
B) 3
C) 9
D) 1

✅ **Answer:** A) 6

---

18. Which function is used to set the maximum recursion depth in Python?

A) sys.setrecursionlimit()
B) os.setrecursionlimit()
C) math.setrecursionlimit()
D) recursion.setlimit()

✅ **Answer:** A) sys.setrecursionlimit()

---

19. Which of the following is not an example of recursion?

A) Fibonacci series
B) Factorial
C) Loop iteration
D) Tower of Hanoi

✅ **Answer:** C) Loop iteration

---

20. What is the advantage of recursion?

A) Simplifies code
B) Faster execution
C) Uses less memory
D) No advantage

✅ **Answer:** A) Simplifies code

21. What is a stack diagram in recursion?

A) A diagram showing variables
B) A diagram showing how functions call each other
C) A diagram showing loops
D) A diagram showing memory usage

✅ **Answer:** B) A diagram showing how functions call each other

---

22. What does each stack frame represent in a stack diagram?

A) A variable
B) A function call
C) A loop
D) A return statement

✅ **Answer:** B) A function call

---

23. What happens to the stack when a function is called?

A) A new frame is added
B) A frame is removed
C) The stack remains the same
D) The stack is cleared

✅ **Answer:** A) A new frame is added

---

24. What happens to the stack when a function returns?

A) A new frame is added
B) The top frame is removed
C) The stack doubles in size
D) The stack becomes empty

✅ **Answer:** B) The top frame is removed

---

25. What does the base case represent in a recursive function's stack diagram?

A) The first function call
B) The middle function call
C) The last function call before returning
D) A loop

✅ **Answer:** C) The last function call before returning

---

26. What happens if there is no base case in recursion?

A) The program executes normally
B) Infinite recursion occurs
C) The function ends quickly
D) The program skips the function

✅ **Answer:** B) Infinite recursion occurs

---

27. What is the first function call at the bottom of the stack called?

A) Base case
B) Recursive call
C) Initial call
D) Loop

✅ **Answer:** C) Initial call

---

28. How is the stack diagram processed in recursion?

A) Top to bottom
B) Bottom to top
C) Left to right
D) Randomly

✅ **Answer:** B) Bottom to top

---

29. What is the purpose of a stack diagram?

A) To show loops
B) To understand how recursion works
C) To remove recursion
D) To print output

✅ **Answer:** B) To understand how recursion works

---

30. Which data structure is used internally to handle recursion?

A) Queue
B) List
C) Stack
D) Array

✅ **Answer:** C) Stack

---

31. What is stored in a stack frame?

A) Function name, arguments, and return address
B) Only variables
C) Only arguments
D) Loops

✅ **Answer:** A) Function name, arguments, and return address

---

32. What is the maximum depth of the stack in Python by default?

A) 500
B) 1000
C) 2000
D) Unlimited

✅ **Answer:** B) 1000

---

33. What happens if the recursion depth exceeds the maximum limit?

A) Recursion stops
B) StackOverflowError
C) RecursionError
D) Infinite loop

✅ **Answer:** C) RecursionError

---

34. Which function can be used to set the recursion depth in Python?

A) sys.setlimit()
B) recursion.limit()
C) sys.setrecursionlimit()
D) recursiondepth()

✅ **Answer:** C) sys.setrecursionlimit()

---

35. What happens first in a recursive stack?

A) Base case is reached
B) Function starts returning
C) Function calls itself
D) Stack becomes empty

✅ **Answer:** C) Function calls itself

---

36. How many stack frames are created for this code?

```
def fun(n):
    if n == 0:
        return
    fun(n-1)
fun(3)
```

A) 1
B) 2
C) 3
D) 4

✅ **Answer:** D) 4

---

37. What is the output of the following code?

```
def fun(n):
    if n == 0:
        return
    print(n)
    fun(n-1)
print(fun(3))
```

A) 3 2 1
B) 1 2 3
C) 3 2 1 None
D) None

✅ **Answer:** C) 3 2 1 None

---

38. Which case stops the recursion process?

A) Base Case
B) Recursive Case
C) Infinite Case
D) Loop Case

✅ **Answer:** A) Base Case

---

39. How does the stack diagram look when a function returns?

A) Frames are added
B) Frames are removed
C) Stack remains the same
D) Stack becomes empty

✅ **Answer:** B) Frames are removed

---

40. What happens to the stack when recursion ends?

A) Stack becomes empty
B) Stack remains full
C) Stack holds one frame
D) Stack doubles

✅ **Answer:** A) Stack becomes empty

---

41. What is the main purpose of the stack diagram?

A) To optimize recursion
B) To visualize the function calls
C) To speed up recursion
D) To store loops

✅ **Answer:** B) To visualize the function calls

---

42. What is the return value of this recursive function?

```
def fun(n):
```

```
    if n == 0:
        return 1
    return n + fun(n-1)
print(fun(3))
```

A) 3
B) 6
C) 5
D) 4

✅ **Answer:** B) 6

---

43. What will happen if a recursive function does not have a return statement?

A) It will return None
B) It will print nothing
C) It will cause an error
D) It will run indefinitely

✅ **Answer:** A) It will return None

---

44. What will happen if recursion depth is not controlled?

A) RecursionError
B) Infinite Loop
C) StackOverflowError
D) Program Crash

✅ **Answer:** A) RecursionError

---

45. What is the last frame in the stack diagram?

A) Base Case
B) First Function Call
C) Last Function Call
D) Initial Case

✅ **Answer:** A) Base Case

---

46. What type of recursion is shown when each recursive call happens at the end of the function?

A) Head Recursion
B) Tail Recursion
C) Indirect Recursion
D) Infinite Recursion

✅ **Answer:** B) Tail Recursion

---

47. What will be the stack height for this code?

```
def fun(n):
    if n == 0:
        return
    fun(n-1)
fun(5)
```

A) 5
B) 6
C) 4
D) 0

✅ **Answer:** B) 6

---

48. Which method is not used to visualize recursion?

A) Stack Diagram
B) Tree Diagram
C) Flowchart
D) Pie Chart

✅ **Answer:** D) Pie Chart

---

49. How does the system know which function call to return to next?

A) Base Case
B) Stack Order
C) Loop
D) Random Call

✅ **Answer:** B) Stack Order

---

50. What is the advantage of stack diagrams?

A) Easy to debug recursive functions
B) Speed up execution
C) Reduce memory usage
D) Prevent infinite loops

✅ **Answer:** A) Easy to debug recursive functions

51. What is multiple assignment in Python?

A) Assigning multiple values to a single variable
B) Assigning a single value to multiple variables
C) Assigning multiple values to multiple variables in one line
D) Assigning a list to a variable

✅ **Answer:** C) Assigning multiple values to multiple variables in one line

---

52. Which of the following is the correct example of multiple assignment in Python?

A) `x, y = 5, 10`
B) `x = y = 10`
C) `x = 5, 10`
D) `x = y, z`

✅ **Answer:** A) `x, y = 5, 10`

---

53. What will be the output of the following code?

```
x, y, z = 1, 2, 3
print(y)
```

A) 1
B) 2
C) 3
D) Error

✅ **Answer:** B) 2

---

54. What happens if the number of variables does not match the number of values in multiple assignment?

A) Error
B) First variable gets all values
C) Last variable gets all values
D) Python ignores extra values

✅ **Answer:** A) Error

---

55. What is the output of the following code?

```
x = y = z = 100
print(x, y, z)
```

A) 100 100 100
B) 100 0 0
C) 100 100 0
D) Error

✅ **Answer:** A) 100 100 100

---

56. Which type of assignment is used here?

```
x, y = 5, 5
```

A) Single Assignment
B) Multiple Assignment
C) Chained Assignment
D) Tuple Assignment

✅ **Answer:** B) Multiple Assignment

---

57. What is the output of this code?

```
a, b = 10, 20
```

```
a, b = b, a
print(a, b)
```

A) 10 20
B) 20 10
C) 10 10
D) Error

✅ **Answer:** B) 20 10

---

58. What will the following code produce?

```
x, y = (5, 10)
print(x + y)
```

A) 15
B) 510
C) 10
D) Error

✅ **Answer:** A) 15

---

59. Which of the following is chained assignment?

A) `x, y = 10, 20`
B) `x = y = z = 50`
C) `x, y, z = 5, 5, 5`
D) `x = 10, y = 20`

✅ **Answer:** B) `x = y = z = 50`

---

60. What is the output of this code?

```
python
CopyEdit
a, b, c = 2, 4, 6
print(a + c)
```

A) 10
B) 12

C) 8
D) Error

✅ **Answer:** C) 8

---

61. How many values can be assigned in multiple assignment?

A) Only 2
B) Only 3
C) Unlimited
D) Only 1

✅ **Answer:** C) Unlimited

---

62. What is the output of the following code?

```
x, y = 5, 10
x, y = y, x
print(x)
```

A) 5
B) 10
C) Error
D) None

✅ **Answer:** B) 10

---

63. Which of the following is correct for swapping two variables using multiple assignment?

A) `x, y = y, x`
B) `x = y, y = x`
C) `x = y`
D) `swap(x, y)`

✅ **Answer:** A) `x, y = y, x`

---

64. What happens if you assign fewer values to more variables?

A) Error
B) Remaining variables are set to `None`
C) Remaining variables are set to `0`
D) Python ignores extra variables

✅ **Answer:** A) Error

---

65. What is the output of this code?

```
x, y, z = 1, 2, 3
print(z)
```

A) 1
B) 2
C) 3
D) Error

✅ **Answer:** C) 3

---

66. Can multiple assignment be used with lists?

A) Yes
B) No
C) Only with tuples
D) Only with integers

✅ **Answer:** A) Yes

---

67. What is the output of this code?

```
a, b, c = [10, 20, 30]
print(a, c)
```

A) 10 20
B) 10 30
C) 20 30
D) Error

✅ **Answer:** B) 10 30

---

68. Which operator is used for multiple assignment in Python?

A) `=`
B) `==`
C) `,`
D) `+=`

✅ **Answer:** A) `=`

---

69. What happens if more values are assigned to fewer variables?

A) Error
B) Remaining values are ignored
C) Remaining values are set to `None`
D) Python splits the values equally

✅ **Answer:** A) Error

---

70. What will the following code output?

```
a, b = 5, 6
a, b = b, a
print(a, b)
```

A) 5 6
B) 6 5
C) Error
D) None

✅ **Answer:** B) 6 5

---

71. Which data type is mostly used in multiple assignment?

A) List
B) Tuple

C) Dictionary
D) All of the above

✅ **Answer:** D) All of the above

---

72. What is the output of this code?

```
x, y = 10, 20
x, y = y, x + y
print(x, y)
```

A) 10 20
B) 20 30
C) 10 30
D) 20 10

✅ **Answer:** B) 20 30

---

73. What is the output of the following code?

```
a, b = 5, 6
a, b, c = a + b, a, b
print(a, b, c)
```

A) 11 5 6
B) 5 6 11
C) Error
D) 6 5 11

✅ **Answer:** A) 11 5 6

---

74. Can multiple assignment be used with strings?

A) Yes
B) No
C) Only with integers
D) Only with lists

✅ **Answer:** A) Yes

---

75. What is the output of this code?

```
x, y, z = "ABC"
print(x, z)
```

A) A C
B) B C
C) A B
D) Error

✅ **Answer:** A) A C

---

76. What will happen if we assign the same value to two variables?

A) Both variables point to the same memory
B) Error
C) Different memory locations
D) None

✅ **Answer:** A) Both variables point to the same memory

---

77. Which of the following supports multiple assignment?

A) List
B) Tuple
C) String
D) All of the above

✅ **Answer:** D) All of the above

---

78. What will be the output?

```
a, b, c = "abc"
print(b)
```

A) a
B) b

C) c
D) Error

✅ **Answer:** B) b

---

79. Can multiple assignment be used for unpacking lists?

A) Yes
B) No
C) Only with integers
D) Only with tuples

✅ **Answer:** A) Yes

---

80. What is the output?

```
x, y, z = range(3)
print(x, y, z)
```

A) 0 1 2
B) 1 2 3
C) Error
D) 0 2 1

✅ **Answer:** A) 0 1 2

81. What is the primary purpose of the while statement in Python?

- a) To execute a block of code a fixed number of times
- b) To execute a block of code as long as a condition is true ✅
- c) To iterate over a list
- d) To define a function

---

82. What will be the output of the following code?

```
x = 5
while x > 0:
    print(x)
    x -= 1
```

- a) 5 4 3 2 1 ✅
- b) 5 4 3 2
- c) 4 3 2 1
- d) Infinite loop

---

83. What happens if the condition in the while loop is False at the start?

- a) The loop executes once
- b) The loop executes infinitely
- c) The loop does not execute ✅
- d) The loop throws an error

---

84. How can you exit a while loop before the condition becomes False?

- a) Using the **continue** statement
- b) Using the **break** statement ✅
- c) Using the **return** statement
- d) Using the **exit()** function

---

85. What is the output of the following code?

```
i = 1
while i < 6:
    if i == 3:
        break
    print(i)
    i += 1
```

- a) 1 2 ✅
- b) 1 2 3
- c) 1 2 3 4 5
- d) 1 2 4 5

---

86. What is the role of the else clause in a while loop?

- a) Executes when the loop is terminated by **break**
- b) Executes when the loop condition is **False** ✅
- c) Executes only when the loop starts

- d) It is not allowed in a **while** loop

---

87. What will happen if the condition in the while loop always evaluates to True?

- a) The loop will run infinitely ✅
- b) The loop will run once
- c) The loop will throw an error
- d) The loop will not execute

---

88. What is the correct syntax for a while loop in Python?

- a) `while condition:` ✅
- b) `while (condition)`
- c) `while: condition`
- d) `while {condition}`

---

89. What will be the output of the following code?

```
x = 0
while x < 3:
    print(x)
    x += 1
else:
    print("Done")
```

- a) 0 1 2
- b) 0 1 2 Done ✅
- c) 0 1 2 3 Done
- d) Done

---

90. Which statement skips the current iteration and moves to the next iteration in a while loop?

- a) break
- b) continue ✅
- c) pass
- d) return

91. What is the initial value of the loop control variable in the following code?

```
i = 0
while i < 5:
    print(i)
    i += 1
```

- a) 5
- b) 0 ✅
- c) 1
- d) None

---

92. How many times will the following loop execute?

```
x = 10
while x > 0:
    print(x)
    x -= 2
```

- a) 5 ✅
- b) 10
- c) Infinite
- d) 2

---

93. What will be the output of the following code?

```
x = 3
while x > 0:
    x -= 1
    if x == 1:
        continue
    print(x)
```

- a) 3 2
- b) 2 0 ✅
- c) 3 2 1
- d) 2 1 0

---

94. Which keyword is used to stop the execution of a while loop?

- a) return
- b) continue

- c) break ✅
- d) stop

---

95. What is the output of this code?

```
i = 1
while i <= 5:
    print(i, end=" ")
    i += 1
else:
    print("End")
```

- a) 1 2 3 4 5
- b) 1 2 3 4 5 End ✅
- c) End 1 2 3 4 5
- d) 1 2 3 4

---

96. Which of the following will NOT cause a while loop to stop?

- a) The condition becomes **False**
- b) The **break** statement is used
- c) The **continue** statement is used ✅
- d) The loop finishes its iterations

---

97. What will be the output of this code?

```
count = 0
while count < 3:
    print("Hello")
    count += 1
else:
    print("Bye")
```

- a) Hello Hello Hello
- b) Hello Hello Hello Bye ✅
- c) Bye Hello Hello
- d) Infinite loop

---

98. Which statement is used to do nothing inside a while loop?

- a) break
- b) continue
- c) pass ✅
- d) skip

---

99. What will happen if there is no increment or decrement in the loop body?

```
x = 5
while x > 0:
    print(x)
```

- a) The loop will execute 5 times
- b) The loop will execute once
- c) Infinite loop ✅
- d) Error

---

100. Which loop structure is best used when the number of iterations is unknown?

- a) for loop
- b) while loop ✅
- c) do-while loop
- d) switch loop

Part 1: Two-Dimensional Tables in Python

---

101. How is a two-dimensional list represented in Python?

- a) List of tuples
- b) List of lists ✅
- c) Tuple of lists
- d) Dictionary

---

102. What is the output of the following code?

```
matrix = [[1, 2], [3, 4]]
print(matrix[1][0])
```

- a) 1
- b) 2
- c) 3 ✅
- d) 4

---

103. How can you initialize a 3x3 matrix with zeros?

- a) `matrix = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]` ✅
- b) `matrix = [0, 0, 0]`
- c) `matrix = [[0]*3]*3`
- d) `matrix = [[0]*3 for i in range(3)]` ✅

---

104. How can you access the second row, third column of the matrix?

```
matrix = [[5, 6, 7], [8, 9, 10], [11, 12, 13]]
```

- a) `matrix[1][2]` ✅
- b) `matrix[2][1]`
- c) `matrix[2][3]`
- d) `matrix[1][3]`

---

105. What is the result of this code?

```
matrix = [[2, 4], [6, 8]]
print(matrix[0][-1])
```

- a) 2
- b) 4 ✅
- c) 6
- d) 8

---

106. How can you append a row to a 2D list?

- a) `matrix.insert([4, 5, 6])`
- b) `matrix.append([4, 5, 6])` ✅
- c) `matrix.add([4, 5, 6])`
- d) `matrix.extend([4, 5, 6])`

---

107. What is the default value of elements when creating a 2D list using:

```
matrix = [[0]*3 for i in range(3)]
```

- a) 0 ✅
- b) None
- c) 1
- d) -1

Part 2: Strings in Python

---

108. What is the output of the following code?

```
string = "Python"
print(string[1:4])
```

- a) Pyt
- b) yth ✅
- c) ytho
- d) hon

---

109. Strings in Python are:

- a) Mutable
- b) Immutable ✅
- c) Dynamic
- d) Constant

---

110. What is the result of this code?

```
text = "Python"
print(text.upper())
```

- a) python
- b) PYTHON ✅

- c) Python
- d) pYTHON

---

111. Which method is used to replace a substring in Python?

- a) `replace()` ✅
- b) `update()`
- c) `change()`
- d) `swap()`

---

112. What will the following code output?

```
string = "hello world"
print(string.split())
```

- a) ['hello', 'world'] ✅
- b) ['helloworld']
- c) ['hello', ' ', 'world']
- d) ['h', 'e', 'l', 'l', 'o', 'w', 'o', 'r', 'l', 'd']

---

113. What does the strip() method do in Python?

- a) Removes spaces from the start only
- b) Removes spaces from the end only
- c) Removes spaces from both start and end ✅
- d) Converts string to lowercase

---

114. What will be the output?

```
s = "Python"
print(s[::-1])
```

- a) nohtyP ✅
- b) Python
- c) Pytho
- d) Error

---

115. What is the correct syntax to concatenate two strings in Python?

- a) `string1.join(string2)`
- b) `string1 + string2` ✅
- c) `string1.append(string2)`
- d) `string1.extend(string2)`

Part 3: Lists in Python

---

116. How can you create a list in Python?

- a) `{}`
- b) `[]` ✅
- c) `()`
- d) `<>`

---

117. What is the output of this code?

```
lst = [1, 2, 3]
lst.append(4)
print(lst)
```

- a) [1, 2, 3, 4] ✅
- b) [4, 1, 2, 3]
- c) [1, 2, 3]
- d) [1, 2, 3, [4]]

---

118. How can you remove the last element from a list?

- a) `list.delete()`
- b) `list.remove()`
- c) `list.pop()` ✅
- d) `list.cut()`

119. What will be the result of this code?

```
lst = [1, 2, 3]
lst.insert(1, 100)
print(lst)
```

- a) [100, 1, 2, 3]
- b) [1, 100, 2, 3] ✅
- c) [1, 2, 100, 3]
- d) [1, 2, 3, 100]

---

120. How do you sort a list in ascending order?

- a) `list.sort()` ✅
- b) `list.sorted()`
- c) `list.reverse()`
- d) `list.order()`

Part 4: Advanced Concepts

---

121. What is the output of this code?

```
lst = [1, 2, [3, 4], 5]
print(lst[2][1])
```

- a) 3
- b) 4 ✅
- c) [3, 4]
- d) Error

---

122. Which method is used to copy a list?

- a) `copy()` ✅
- b) `clone()`
- c) `duplicate()`
- d) `append()`

---

123. How can you reverse a list in Python?

- a) `list.reverse()` ✅
- b) `list[::-1]`
- c) `reversed(list)`
- d) All of the above ✅

---

124. What will this code produce?

```
lst = [1, 2, 3]
print(len(lst))
```

- a) 2
- b) 3 ✅
- c) 4
- d) Error

---

125. Which method is used to remove elements from a list by index?

- a) `pop()` ✅
- b) `remove()`
- c) `delete()`
- d) `discard()`

---

126. How do you create a deep copy of a list?

- a) `list.copy()`
- b) `copy.deepcopy(list)` ✅
- c) `list.clone()`
- d) `deepcopy(list)`

---

127. What is the default sorting order of `sort()`?

- a) Descending
- b) Ascending ✅
- c) Random
- d) None

---

128. How can you join list elements into a string?

- a) `join()` ✅
- b) `concat()`
- c) `merge()`
- d) `append()`

---

129. What is the output of this code?

```
lst = ['a', 'b', 'c']
print(lst.index('b'))
```

- a) 0
- b) 1 ✅
- c) 2
- d) Error

---

130. What is the result of `max([4, 8, 2])`?

- a) 4
- b) 8 ✅
- c) 2
- d) None

131. What is a string in Python?

- a) Mutable data type
- b) Immutable data type ✅
- c) Numeric data type
- d) List

---

132. What does the `len()` function return when applied to a string?

```
s = "Hello World"
print(len(s))
```

- a) 10

- b) 11 ✅
- c) 12
- d) 9

---

133. How are strings stored in Python?

- a) List of integers
- b) List of characters ✅
- c) Dictionary
- d) Tuples

---

134. What is the output of the following code?

```
s = "Python"
print(s[3])
```

- a) t
- b) h ✅
- c) y
- d) n

---

135. What will happen if you try to modify a string in Python?

- a) It will modify successfully
- b) It will throw an error ✅
- c) It will add the new value
- d) It will replace the old value

---

136. What is the output of this code?

```
s = "Python"
print(s[-1])
```

- a) P
- b) n ✅
- c) o
- d) y

---

137. How can you access the first character of a string `s`?

- a) `s[1]`
- b) `s[0]` ✅
- c) `s[-1]`
- d) `s[2]`

---

138. What is the output of the following code?

```
s = "Python"
print(s[::2])
```

- a) Pto ✅
- b) yhn
- c) Pyth
- d) Ptn

---

139. What will be the output of this code?

```
s = "Hello World"
print(s[:5])
```

- a) Hello ✅
- b) World
- c) Hello World
- d) ello

---

140. How can you reverse a string `s` in Python?

- a) `s.reverse()`
- b) `s[::-1]` ✅
- c) `reverse(s)`
- d) `s.flip()`

---

141. What will be the output of this code?

```
s = "Python"
print(s[1:4])
```

- a) yth ✅
- b) Pyt
- c) tho
- d) thon

---

142. Which method is used to count the occurrences of a substring in a string?

- a) `len()`
- b) `count()` ✅
- c) `find()`
- d) `replace()`

---

143. How can you traverse a string in Python?

- a) Using a for loop ✅
- b) Using a while loop ✅
- c) Using if-else
- d) Both a and b ✅

---

144. What is the output of the following code?

```
s = "Python"
for i in s:
    print(i, end=" ")
```

- a) P y t h o n ✅
- b) Python
- c) n o h t y P
- d) Error

---

145. What is the result of this code?

```
s = "Hello"
print(s[1:4:2])
```

- a) el
- b) eo
- c) Hl
- d) el ✅

---

146. Which method converts the string into uppercase letters?

- a) `upper()` ✅
- b) `lower()`
- c) `capitalize()`
- d) `title()`

---

147. What will be the output of this code?

```
s = "Python Programming"
print(s[7:18])
```

- a) Programming ✅
- b) Python
- c) rammi
- d) on Pro

---

148. What is the correct method to remove whitespace from both ends of a string?

- a) `strip()` ✅
- b) `split()`
- c) `remove()`
- d) `clean()`

---

149. What will `len("Welcome")` return?

- a) 6
- b) 7 ✅
- c) 8
- d) 5

---

150. How do you concatenate two strings `s1` and `s2`?

- a) `s1.append(s2)`
- b) `s1.join(s2)`
- c) `s1 + s2` ✅
- d) `s1.merge(s2)`

---

151. What is the output of the following code?

```
s = "Python"
print(len(s[1:4]))
```

- a) 2
- b) 3 ✅
- c) 4
- d) 6

---

152. Which method returns the index of the first occurrence of a substring?

- a) `index()` ✅
- b) `find()`
- c) `search()`
- d) `replace()`

---

153. What is the output of this code?

```
s = "Python"
print(s.upper())
```

- a) python
- b) PYTHON ✅
- c) Python
- d) pYTHON

---

154. How can you iterate through each character in a string?

- a) Using a `for` loop ✅
- b) Using a `while` loop ✅
- c) Both a and b ✅
- d) None

---

155. What is the result of this code?

```
s = "python"
print(s.capitalize())
```

- a) Python ✅
- b) PYTHON
- c) python
- d) pYTHON

---

156. What will this code print?

```
s = "hello world"
print(s.title())
```

- a) Hello World ✅
- b) Hello world
- c) HELLO WORLD
- d) hello world

---

157. Which method is used to check if a string starts with a specific substring?

- a) `startswith()` ✅
- b) `endswith()`
- c) `isalpha()`
- d) `find()`

---

158. What will be the output of this code?

```
s = "Hello World"
print(s.split())
```

- a) ['Hello', 'World'] ✅

- b) ['H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd']
- c) Hello World
- d) ['Hello World']

---

159. Which method joins a list of strings into one string?

- a) `join()` ✅
- b) `concat()`
- c) `merge()`
- d) `combine()`

---

160. What will be the output of this code?

```
s = "Python Programming"
print(s.replace("Python", "Java"))
```

- a) Python Programming
- b) Java Programming ✅
- c) Programming Java
- d) Error

161. What is the primary purpose of the `for` loop in Python?

- a) Infinite loop
- b) Conditional execution
- c) Iterating over sequences ✅
- d) Defining functions

---

162. Which keyword is used to start a `for` loop in Python?

- a) while
- b) loop
- c) for ✅
- d) if

---

163. What will be the output of the following code?

```
for i in range(3):
    print(i)
```

- a) 0 1 2 ✅
- b) 1 2 3
- c) 0 1 2 3
- d) 1 2

---

164. What is the default starting value of `range()` in Python?

- a) 0 ✅
- b) 1
- c) -1
- d) None

---

165. How many times will this loop execute?

```
for i in range(5):
    print("Hello")
```

- a) 4
- b) 5 ✅
- c) Infinite
- d) 0

---

166. What is the correct syntax for iterating through a list using a `for` loop?

- a) `for i in range(list):`
- b) `for i in list:` ✅
- c) `for i = list:`
- d) `for i from list:`

---

167. What is the output of the following code?

```
for i in "Python":
    print(i, end=" ")
```

- a) Python

- b) P y t h o n ✅
- c) Pyt hon
- d) Error

---

168. What will the following code print?

```
for i in range(1, 6, 2):
    print(i)
```

- a) 1 2 3 4 5
- b) 1 3 5 ✅
- c) 2 4 6
- d) 1 4 6

---

169. What is the output of this code?

```
numbers = [10, 20, 30]
for num in numbers:
    print(num)
```

- a) 10 20 30 ✅
- b) [10, 20, 30]
- c) num num num
- d) Error

---

170. Which function is commonly used with `for` loops to generate numbers?

- a) `generate()`
- b) `range()` ✅
- c) `count()`
- d) `numbers()`

---

171. What will be the output of this code?

```
for i in range(4, 0, -1):
    print(i)
```

- a) 4 3 2 1 ✅

- b) 0 1 2 3 4
- c) 4 0 -1 -2
- d) Error

---

172. What is the output of this loop?

```
for i in range(2, 10, 3):
    print(i)
```

- a) 2 5 8 ✅
- b) 2 4 6 8
- c) 3 6 9
- d) 2 3 4

---

173. What will the following code print?

```
for i in range(5):
    if i == 3:
        break
    print(i)
```

- a) 0 1 2 ✅
- b) 0 1 2 3
- c) 0 1 2 3 4
- d) 1 2 3

---

174. Which statement is used to skip the current iteration in a `for` loop?

- a) break
- b) continue ✅
- c) pass
- d) return

---

175. What will this code print?

```
for i in range(5):
    if i == 3:
        continue
    print(i)
```

- a) 0 1 2 3 4
- b) 0 1 2 4 ✅
- c) 0 1 3 4
- d) 1 2 3 4

---

176. What does the `else` block in a `for` loop do?

- a) Executes when the loop condition is false
- b) Executes when the loop completes normally ✅
- c) Executes only if the loop is infinite
- d) Executes before the loop starts

---

177. What will be the output of this code?

```
for i in range(3):
    print(i)
else:
    print("Done")
```

- a) 0 1 2 Done ✅
- b) 0 1 2
- c) Done 0 1 2
- d) Error

---

178. How can you iterate through both keys and values of a dictionary?

- a) `for k, v in dict:`
- b) `for k, v in dict.items():` ✅
- c) `for k, v in dict.values():`
- d) `for k in dict.keys():`

---

179. What will be the output of the following code?

```
for i in range(3):
    pass
print("End")
```

- a) 0 1 2

- b) End ✅
- c) Error
- d) None

---

180. What will happen if the `range()` function is given only one argument?

- a) It starts from 1
- b) It starts from 0 ✅
- c) It starts from the given number
- d) It gives an error

181. What is string slicing in Python?

- a) Breaking a string into two parts
- b) Extracting a portion of the string ✅
- c) Adding two strings
- d) Comparing two strings

---

182. What is the output of the following code?

```
s = "Python Programming"
print(s[0:6])
```

- a) Python ✅
- b) Pytho
- c) Pyth
- d) Python Programming

---

183. How do you extract the last 4 characters from the string `"Programming"`?

- a) `s[-4:]` ✅
- b) `s[4:-1]`
- c) `s[4:8]`
- d) `s[0:4]`

---

184. What will be the output of this code?

```
s = "Python"
print(s[:4])
```

- a) Pyth ✅
- b) on
- c) Pyt
- d) ytho

---

185. What is the result of the following code?

```
s = "Python"
print(s[2:5])
```

- a) yth
- b) tho ✅
- c) ytho
- d) on

---

186. How can you compare two strings `s1` and `s2` in Python?

- a) `s1 == s2` ✅
- b) `s1.compare(s2)`
- c) `s1.equal(s2)`
- d) `s1 = s2`

---

187. What is the result of this code?

```
s1 = "Hello"
s2 = "hello"
print(s1 == s2)
```

- a) True
- b) False ✅
- c) Error
- d) None

---

188. Which method is used to compare two strings ignoring case sensitivity?

- a) `lower()` ✅
- b) `find()`
- c) `slice()`
- d) `title()`

---

189. What is the correct way to find the index of the first occurrence of `"o"` in `"Hello World"`?

- a) `s.index("o")`
- b) `s.find("o")` ✅
- c) `s.locate("o")`
- d) `s.search("o")`

---

190. What is the output of this code?

```
s = "Python Programming"
print(s.find("g"))
```

- a) 10 ✅
- b) 11
- c) 12
- d) 13

---

191. What will the `find()` method return if the substring is not found?

- a) 0
- b) -1 ✅
- c) 1
- d) None

---

192. How can you search for the substring `"World"` in `"Hello World"`?

- a) `s.index("World")`
- b) `s.find("World")` ✅
- c) `s.locate("World")`
- d) `s.search("World")`

---

193. What is the output of this code?

```
s = "Hello World"
print(s.find("l", 4))
```

- a) 2
- b) 3
- c) 9 ✅
- d) 4

---

194. What is the result of this code?

```
s = "Programming"
print(s[3:7])
```

- a) gramm
- b) gram ✅
- c) gramming
- d) ming

---

195. What is the output of this code?

```
s = "Python"
print(s[-3:])
```

- a) hon ✅
- b) Pyt
- c) yth
- d) tho

---

196. Which function is used to find the last occurrence of a substring?

- a) `rfind()` ✅
- b) `find()`
- c) `index()`
- d) `count()`

---

197. What will be the output of the following code?

```
s = "Python Programming"
print(s.find("z"))
```

- a) -1 ✅
- b) 0
- c) 1
- d) Error

---

198. What is the result of this code?

```
s = "Hello World"
print(s.find("o", 5))
```

- a) 4
- b) 7 ✅
- c) 5
- d) 6

---

199. Which method is used to search for the first index of a substring in a string?

- a) `index()` ✅
- b) `find()`
- c) `locate()`
- d) `match()`

---

200. What will be the output of this code?

```
s = "Hello World"
print(s.find("World"))
```

- a) -1
- b) 6 ✅
- c) 5
- d) 7