

**Python Data Structures and Functions Test**

**Total Questions:** 30 (20 MCQs, 10 Coding Challenges, Total 80 MARKS) **Time Limit:** 60 minutes



**Section 1: Multiple Choice Questions (20 Questions , 2 Marks**

**Each)**

**Strings (2 Questions)**

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

s = "PythonProgramming"

print(s[2:10:2])

a) toPorm   
b) tPg   
c) toPg   
d) hnrg

**ANSWER:**

2. What does "Hello".replace('l', '') return?   
a) Heo   
b) Helo   
c) Hello   
d) H

**ANSWER: a)Heo**



**Lists and Tuples (3 Questions)**

3. What will be the output?

lst = [1, 2, 3]

lst.extend([4, 5])

print(lst)

a) [1, 2, 3, [4, 5]]   
b) [1, 2, 3, 4, 5]   
c) [1, 2, 3]   
d) [1, 2, 3, (4, 5)]

**ANSWER:** [1,2,3,4,5]

4. Which of the following statements is **incorrect** regarding tuples? a) Tuples are immutable.

b) Tuples consume less memory than lists.

c) Tuples support item assignment.   
d) Tuples can contain mutable objects.

**ANSWER:** c) Tuples support item assignment.

5. What will tuple([1, 2, 3, 4]) return?   
a) (1, 2, 3, 4)   
b) [1, 2, 3, 4]   
c) {1, 2, 3, 4}   
d) None

**ANSWER:** a) (1, 2, 3, 4)



**List Slicing (2 Questions)**

6. What does lst[-3:-1] return for lst = [10, 20, 30, 40, 50] ?

a) [30, 40]   
b) [40, 50]   
c) [30, 40, 50]   
d) [20, 30]

**ANSWER:** a) [30, 40]

7. What does list[::-1] return for list = ['a', 'b', 'c', 'd'] ? a) ['a', 'b', 'c', 'd']   
b) ['d', 'c', 'b', 'a']   
c) ['c', 'd']   
d) ['b', 'a']

**ANSWER:** b) ['d', 'c', 'b', 'a']



**List Comprehension (2 Questions)**

8. What does [x\*\*2 for x in range(5) if x % 2 == 0] return?

a) [0, 4, 16]   
b) [1, 4, 9, 16]   
c) [0, 1, 4, 9, 16]   
d) [0, 2, 4]

**ANSWER:** d) [0, 2, 4]

9. Identify the incorrect list comprehension syntax: a) [x for x in range(10) if x % 2 == 0] b) [x for x in range(10) x % 2 == 0]   
c) [x+1 for x in range(3)]   
d) [x for x in range(3) if x > 1]

**ANSWER:** b) [x for x in range(10) x % 2 == 0]



**Range (2 Questions)**

10. What is the output of range(1, 5, 2) in list form?

a) [1, 3]   
b) [1, 2, 3, 4]

c) [1, 2, 3, 4, 5]   
d) [1, 2, 4]

**ANSWER:** a) [1, 3]

11. What happens if range(5, 1, 1) is executed? a) [5, 4, 3, 2, 1]   
b) [5, 4, 3, 2]   
c) []   
d) [5]

**ANSWER:** c) []



**Dictionaries & Dictionary Comprehension (3 Questions)**

12. What is the output of the below?

d = {'x': 1, 'y': 2}   
d.update({'y': 3, 'z': 4})

print(d)

a) {'x': 1, 'y': 3, 'z': 4}   
b) {'x': 1, 'y': 2, 'z': 4}   
c) {'x': 1, 'y': 2}   
d) {'x': 1, 'z': 4}

**ANSWER:** a) {'x': 1, 'y': 3, 'z': 4}

13. What does {x: x\*\*3 for x in range(2, 5)} return? a) {2: 8, 3: 27, 4: 64}   
b) {2: 4, 3: 9, 4: 16}   
c) {2: 6, 3: 9, 4: 12}   
d) {2: 3, 3: 6, 4: 9}

**ANSWER:** a) {2: 8, 3: 27, 4: 64}

14. What will list(d.keys())[0] return for d = {'a': 100, 'b': 200} ? a) 100   
b) 'a'   
c) 'b'   
d) None

**ANSWER:** b) 'a'



**Functions (6 Questions)**

15. What is the output?

def f (x, y=[]):

y.append(x)

return y

print(f(1))

print(f(2))

a) [1] [2]   
b) [1] [1, 2]   
c) [1] []   
d) Error

**ANSWER:**

b) [1] [1, 2]

16. What will lambda x: x \* x return when x = 3 ?

a) 6   
b) 9   
c) 3   
d) None

**ANSWER:** b) 9

17. Which function is used to filter elements from an iterable?

a) map()   
b) reduce()   
c) filter()   
d) apply()

**ANSWER:** c) filter()

18. What is the output?

from functools import reduce

reduce(lambda x, y: x \* y, [1, 2, 3, 4])

a) 10   
b) 24   
c) [1, 2, 3, 4]   
d) None

**ANSWER:** b) 24

19. What does filter(lambda x: x > 2, [1, 2, 3, 4]) return?

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

**ANSWER:** a) [3, 4]

20. Write a function that accepts a variable number of arguments and prints them.

**ANSWER:** def arguments(\*args):  
 fro arg in args:

Print(args)

arguments(1,2,3,”hello”)

Output: 1,2,3,”hello”



**Section 2: Coding Challenges (10 Questions , 4 Marks Each)**

1. **Reverse a string without using slicing**

def reverse\_string(s):  
 string(“”):

for char in s:  
 string=char +string

reverse\_string(“hello”)

output:”olleh”

# Example

print(reverse\_string("Python"))

**Expected Output:**"nohtyP"

**ANSWER:**



2. **Write a function to remove duplicates from a list**

def remove\_duplicates(lst):

# Your code here

# Example

print(remove\_duplicates([1, 2, 2, 3, 4, 4, 5]))

**Expected Output:**[1, 2, 3, 4, 5]

**ANSWER:**

removeDuplicates=set()

result=[];

for item in 1st:

result.append(item);

removeDuplicates.add(item)

return result;

print(remove\_duplicates([1, 2, 2, 3, 4, 4, 5]))

**Output:**[1, 2, 3, 4, 5]



3. **Write a dictionary comprehension that reverses keys and values**

def reverse\_dict(d):

# Your code here

# Example

print(reverse\_dict({'a': 1, 'b': 2, 'c': 3}))

**Expected Output:**{1: 'a', 2: 'b', 3: 'c'}

**ANSWER:** fdef reverse\_dict(d):

return {v: k for k, v in d.items()}

print(reverse\_dict({'a': 1, 'b': 2, 'c': 3}))

**Output:**{1: 'a', 2: 'b', 3: 'c'}



4. **Implement map() to find the cube of a list of numbers**

def cube\_numbers(lst):

# Your code here

# Example

print(cube\_numbers([1, 2, 3, 4]))

**Expected Output:**[1, 8, 27, 64]

**ANSWER:**

def cube\_numbers(lst):

return list(map(lambda x: x\*\*3, lst))

print(cube\_numbers([1, 2, 3, 4]))

**Output:**[1, 8, 27, 64]



5. **Implement a function using filter() to remove vowels from a string**

def remove\_vowels(s):

# Your code here

# Example

print(remove\_vowels("hello world"))

**Expected Output:**"hll wrld"

**ANSWER:**

def remove\_vowels(s):

vowels = {'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'}

return "".join(filter(lambda char: char not in vowels, s))

print(remove\_vowels("hello world"))

**Output:**"hll wrld"



6. **Write a function that returns a dictionary of squares from 1 to n**

def squares\_dict(n):

# Your code here

# Example

print(squares\_dict(5))

**Expected Output:**{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

**ANSWER:**

def squares\_dict(n):

return {x: x\*\*2 for x in range(1, n+1)}

print(squares\_dict(5))

Output**:**{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}



7. **Write a function to merge two dictionaries**

def merge\_dicts(d1, d2):

# Your code here

# Example

print(merge\_dicts({'a': 1, 'b': 2}, {'b': 3, 'c': 4}))

**Expected Output:**{'a': 1, 'b': 3, 'c': 4}

**ANSWER:**

def merge\_dicts(d1, d2):

d1\_copy = d1.copy()

d1\_copy.update(d2)

return d1\_copy

print(merge\_dicts({'a': 1, 'b': 2}, {'b': 3, 'c': 4}))

**Output:**{'a': 1, 'b': 3, 'c': 4}



8. **Implement reduce() to compute factorial of a number**

from functools import reduce

def factorial(n):

# Your code here

# Example

print(factorial(5))

**Expected Output:**120

**ANSWER:**

from functools import reduce

def factorial(n):

def factorial(n):

return reduce(lambda x, y: x \* y, range(1, n +1))

print(factorial(5))

Output**:**120



9. **Write a function to flatten a nested list**

def flatten\_list(nested\_lst):

# Your code here

# Example   
print(flatten\_list([[1, 2], [3, 4], [5]]))

**Expected Output:**[1, 2, 3, 4, 5]

**ANSWER:**

def flatten\_list(nested\_list):

flatList = [ ]

for item in nested\_list:

if instance(item, list):

flatList.extend(flatten\_list(item))

print(flatten\_list([[1, 2], [3, 4], [5]]))

Output:[1, 2, 3, 4, 5]



10. **Write a lambda function to check if a number is prime**

is\_prime = lambda n: # Your code here

# Example   
print(is\_prime(7))   
print(is\_prime(10))

**Expected Output:**   
True   
False

**ANSWER**:

is\_prime = lambda n:

n > 1 and all(n % i != 0)

for i in range(2, int(n\*\*0.5) + 1))

print(is\_prime(7))   
print(is\_prime(10))

**Output:**   
True   
False

