

**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**. **ANSWER: c**

**2. ANSWER: a**

**Lists and Tuples (3 Questions)**

3. **ANSWER: b**

4. **ANSWER: C**

5. **ANSWER: a**

**List Slicing (2 Questions)**

6. **ANSWER: a**

7. **ANSWER: b**

**List Comprehension (2 Questions)**

8. **ANSWER: a**

9. **ANSWER: b**

**Range (2 Questions)**

10. **ANSWER: a**

11. **ANSWER: c**

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

12. **ANSWER: a**

13. **ANSWER: a**

14. **ANSWER: b**

**Functions (6 Questions)**

15. **ANSWER: b**

16. **ANSWER: b**

**17. ANSWER: c**

**18. ANSWER: b**

**19. ANSWER: a**

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

**ANSWER:**

**Def print\_args(\*args):**

**For arg in args:**

**Print(arg)**

**Print\_args(1, “hello”, 3.14, [1, 2, 3])**

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

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

def reverse\_string(s):  
 # Your code h

# Example

print(reverse\_string("Python"))

**Expected Output:**"nohtyP"

**ANSWER:**

**Def reverse\_string(s):**

**Reversed\_str = “”**

**For char in s:**

**Reversed\_str = char + reversed\_str**

**Return reversed\_str**

**Print(reverse\_string(“Python”))**

**Output: “nohtyP”**



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:**

Def remove\_duplicates(lst):

Unique\_list = []

For item in lst:

If item not in unique\_list: Unique\_list.append(item)

Return unique\_list

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:**

Def 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]**

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 {i: i\*\*2 for I 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):**

**Merged = d1.copy()**

**Merged.update(d2)**

**Return merged**

**Print(merge\_dicts({‘a’: 1, ‘b’: 2}, {‘b’: 3, ‘c’: 4}))**

**Output: {‘a’: 1, ‘b’: 3, ‘c’: 4}**

