Infosys Internship 6.0
Python Tasks

1. Basic Input/Output and Arithmetic

1. Take user input and display it

```
name = input("Enter your name: ") print(f"Hello, {name}!") # Prints a personalized greeting
```

2. Add two numbers

```
a, b = 10, 5 print("Sum =", a + b) # Basic arithmetic operation
```

3. Calculate area of a circle

```
r = 7 area = 3.14 * r**2 print("Area of circle:", area) # Formula: πr²
```

4. Check even or odd number

```
num = 8 print("Even" if num % 2 == 0 else "Odd") # Conditional check
```

5. Swap two variables without temp

```
x, y = 5, 10 x, y = y, x # Pythonic swapping print(x, y)
```

2. Creating Lists

1. Create and access list elements

```
nums = [10, 20, 30, 40] print(nums[0], nums[-1]) # Access first and last
```

2. Add and remove elements

```
nums.append(50) nums.remove(20) print(nums)
```

3. Sort and reverse a list

```
nums.sort() nums.reverse() print(nums)
```

4. List comprehension

```
squares = [x**2 for x in range(1, 6)] print(squares)
```

5. Find max and min in list

```
nums = [3, 8, 1, 6] print("Max:", max(nums), "Min:", min(nums))
```

3. Loops and Conditionals

1. Print first 10 natural numbers

```
for i in range(1, 11): print(i, end=" ")
```

2. Sum of N natural numbers

```
n = 10 print("Sum =", sum(range(1, n+1)))
```

3. Multiplication table

```
n = 5 for i in range(1, 11): print(f"{n} x {i} = {n*i}")
```

4. Check prime number

```
n = 13 is_prime = all(n % i != 0 for i in range(2, int(n**0.5)+1))
print("Prime" if is_prime else "Not Prime")
```

5. Print Fibonacci sequence

```
a, b = 0, 1 for _ in range(10): print(a, end=" ") a, b = b, a + b
```

4. Strings

1. Reverse a string

```
s = "Python" print(s[::-1]) # Reverses the string
```

2. Check palindrome

```
s = "madam" print("Palindrome" if s == s[::-1] else "Not Palindrome")
```

3. Count vowels

```
s = "beautiful" vowels = "aeiou" count = sum(1 for ch in s if ch.lower()
in vowels) print("Vowel count:", count)
```

4. Find substring occurrence

```
s = "banana" print(s.count("a")) # Counts occurrences of 'a'
```

5. Change case

```
text = "Python" print(text.upper(), text.lower(), text.title())
```

5. Functions

1. Function to calculate factorial

```
def factorial(n): return 1 if n == 0 else n * factorial(n - 1)
print(factorial(5))
```

2. Function to check prime

```
def is_prime(n): if n < 2: return False for i in range(2, int(n**0.5)+1):
if n % i == 0: return False return True print(is_prime(11))
```

3. Lambda + map example

```
nums = [1, 2, 3, 4] print(list(map(lambda x: x**2, nums)))
```

4. Args and kwargs example

```
def details(*args, **kwargs): print(args) print(kwargs) details("Amit",
age=25, city="Delhi")
```

5. Function returning multiple values

```
def calc(x, y): return x+y, x*y print(calc(4, 5))
```

6. File Handling

1. Write to a file

```
with open("demo.txt", "w") as f: f.write("Hello, Python!")
```

2. Read from a file

```
with open("demo.txt", "r") as f: print(f.read())
```

3. Append to a file

```
with open("demo.txt", "a") as f: f.write("\nNew line added")
```

4. Read file line by line

```
with open("demo.txt") as f: for line in f: print(line.strip())
```

5. Handle missing file

```
try: with open("missing.txt") as f: print(f.read()) except
FileNotFoundError: print("File not found!")
```

7. Miscellaneous

1. Random number generation

```
import random print(random.randint(1, 100))
```

2. Get current date and time

```
import datetime print(datetime.datetime.now())
```

3. Sort dictionary by values

```
data = {"a": 3, "b": 1, "c": 2} print(sorted(data.items(), key=lambda x:
x[1]))
```

4. Flatten nested list

```
nested = [[1, 2], [3, 4], [5]] flat = [num for sub in nested for num in
sub] print(flat)
```

5. Enumerate with index

```
for i, fruit in enumerate(["apple", "banana", "cherry"]): print(i, fruit)
```

INTERMEDIATE TASKS

1. Dictionary comprehension

```
squares = {x: x**2 for x in range(5)} print(squares)
```

2. Exception handling

```
try: result = 10 / 0 except ZeroDivisionError: print("Cannot divide by zero!")
```

3. Basic class (OOP)

```
class Student: def __init__(self, name, marks): self.name = name self.marks = marks def info(self): print(f"{self.name}: {self.marks}") s1 = Student("Ravi", 85) s1.info()
```

4. Using JSON module

```
import json data = {"name": "Riya", "age": 22} json_data = json.dumps(data) print(json_data)
```

5. Working with sets

```
a, b = {1, 2, 3}, {3, 4, 5} print("Union:", a | b, "Intersection:", a & b)
```

ADVANCED TASKS

1. Decorators (Function Wrappers)

```
def logger(func): def wrapper(): print("Start") func() print("End") return wrapper @logger def hello(): print("Inside function") hello()
```

2. NumPy basics

```
import numpy as np arr = np.array([1, 2, 3, 4]) print(arr.mean(), arr + 5)
```

3. Pandas DataFrame filtering

```
import pandas as pd df = pd.DataFrame({"Name": ["A", "B", "C"], "Score": [40, 80, 60]}) print(df[df["Score"] > 50])
```

4. API data fetching

```
import requests response = requests.get("https://jsonplaceholder.typicode.com/posts/1") print(response.json())
```

5. OS module operations

```
import os print(os.getcwd())
```