**Test 2**

**Phase-1**

Functions, List comprehensions

1. **Q:** Write a function to add two numbers.
2. **Q:**  **Q:** Function with \*args to return sum of all numbers.
3. Q: Print squares of numbers 1–5, using list comprehension.
4. Write a function to print numbers from 1 to 100 without using loop.
5. **Q:** Replace negative numbers with 0 in a list, using list comprehension.
6. **Q:** Create list of unique chars from string, using list comprehension.
7. **Q:** Generate 3×3 matrix with numbers, using list comprehension.
8. **Q:** Function to check prime number.
9. **Q:** Function to generate all prime numbers up to “n”.
10. **Q:** Function to check Armstrong number.

**Phase-2**

ALL Methods

1. Remove all characters that occur more than once in "programming".
2. Find the longest word in a sentence.
3. Find all index positions of a given element in a tuple (1,2,3,2,4,2,5) for element 2.
4. Given a tuple (10,20,30,40,50), split it into two tuples of equal length.
5. Sort a dictionary by its values. (d = {"a":3, "b":1, "c":2} )
6. Add a new key "grade":"A" to the dictionary above. ( {"name":"Kiran", "age":20} )
7. Add 100 into the set {10,20,30}
8. Check if 5 exists in {1,2,3,4}.
9. Replace the 2nd element of list [1,2,3] with 99
10. Find the second largest number in a list. (nums = [10, 20, 4, 45, 99] )

Note: Answers are mentioned Below…..!

Question 1 :

Create a Rectangle class that takes length and width as an argument in the constructor

Then create a method to calculate area and another methodto calculate perimeter

Finally create an object and print area and perimeter

class Rectangel:

  def \_\_init\_\_(self,length, width):

    self.length = length

    self.width = width

  def area(self):

    return self.length + self.width

  def perimeter(self):

    return 2\*(self.length + self.width)

r1 = Rectangel(10,20)

print(r1.area())

print(r1.perimeter())

Question 2 :

Create student class that takes name and marks of 3 subjects as arguments in constructor.

Then create a method to print the average

class Student:

  def \_\_init\_\_(self, name, marks):

    self.name = name

    self.marks = marks

  def average(self):

    #return sum(self.marks) / len(self.marks)

    sum = 0

    for val in self.marks:

      sum += val

    #print("hi", self.name, "your avg score is :", sum/len(self.marks))

    #this above line if declared and in calling object if i use print statement

    #then it will print None

    return "hi", self.name, "your avg score is :", sum/len(self.marks)

s1 = Student("kiran", [99,98,97])

print(s1.average())

s1.name = "changed"

s1.average()

**Phase-1**

Functions, List comprehensions

1. **Q:** Write a function to add two numbers.

def add(a, b):

    return a + b

print(add(3, 5))  # 8

1. **Q:**  **Q:** Function with \*args to return sum of all numbers.

def add\_all(\*args):

    return sum(args)

print(add\_all(1, 2, 3, 4, 5))  # 15

1. Q: Print squares of numbers 1–5, using list comprehension.

[x\*\*2 for x in range(1,6)]   # [1, 4, 9, 16, 25]

1. Write a function to print numbers from 1 to 100 without using loop.

def p(n=1):

  if n > 100:

    return

  print(n)

  p(n + 1)

p()

1. **Q:** Replace negative numbers with 0 in a list, using list comprehension.

nums = [-3, 5, -1, 7, -9]

result = [x if x > 0 else 0 for x in nums]

print(result)  # [0,5,0,7,0]

1. **Q:** Create list of unique chars from string, using list comprehension.

unique = [ch for ch in set("hello")]

print(unique)

1. **Q:** Generate 3×3 matrix with numbers, using list comprehension.

matrix = [[i+j\*3 for i in range(1,4)] for j in range(3)]

print(matrix)

# [[1,2,3],[4,5,6],[7,8,9]]

1. **Q:** Function to check prime number.

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(7))  # True

print(is\_prime(10)) # False

1. **Q:** Function to generate all prime numbers up to “n”.

def primes\_upto(n):

    primes = []

    for num in range(2, n+1):

        if all(num % i != 0 for i in range(2, int(num\*\*0.5)+1)):

            primes.append(num)

    return primes

print(primes\_upto(30))

1. **Q:** Function to check Armstrong number.

def is\_armstrong(n):

    s = str(n)

    power = len(s)

    return n == sum(int(d)\*\*power for d in s)

print(is\_armstrong(153))  # True

print(is\_armstrong(123))  # False

**Phase-2**

ALL Methods

1. Remove all characters that occur more than once in "programming".

s = "programming"

result = "".join([ch for ch in s if s.count(ch)==1])

print(result)

1. Find the longest word in a sentence.

s = "Python is an amazing programming language"

words = s.split()

longest = max(words, key=len)

print(longest)

1. Find all index positions of a given element in a tuple (1,2,3,2,4,2,5) for element 2.

t = (1,2,3,2,4,2,5)

indices = [i for i,v in enumerate(t) if v==2]

print(indices)

1. Given a tuple (10,20,30,40,50), split it into two tuples of equal length.

t = (10,20,30,40,50)

mid = len(t)//2

t1, t2 = t[:mid], t[mid:]

print(t1, t2)

1. Sort a dictionary by its values. (d = {"a":3, "b":1, "c":2} )

d = {"a":3, "b":1, "c":2}

sorted\_d = dict(sorted(d.items(), key=lambda x:x[1]))

print(sorted\_d)

1. Add a new key "grade":"A" to the dictionary ( {"name":"Kiran", "age":20} )

student["grade"] = "A"

print(student)

1. Add 100 into the set {10,20,30}

s = {10, 20, 30}

s.add(100)

print(s)

1. Check if 5 exists in {1,2,3,4}.

s = {1, 2, 3, 4}

print(5 in s)

1. Replace the 2nd element of list [1,2,3] with 99

nums = [1, 2, 3]

nums[1] = 99

print(nums)

1. Find the second largest number in a list. (nums = [10, 20, 4, 45, 99] )

nums = [10, 20, 4, 45, 99]

nums.sort()

print(nums[-2])