PAGE NO.:
DATE: / /

* Exercise 6 & 7 *

1) Creation of list and changing of any one element, also display the length of list.

→
```
a = ['One', 'Two', 'Three', 'Five']
a[4] = 'Four'
print(a)
print(len(a)).
```

2) Create a list and append two elements in it

→
```
stud = ['Madhuri', 'Kirti', 'Rahul', 'Sumit', 'Mayuri']
stud[2:4] = ['Nandini', 'Sanika']
print(stud).
```

3) Create a list and sort it.

→
```
num = [10, 4, 3, 25, 90]
num.sort().
print("Sorted list is")
print(num)
```

4) Create a list of numbers and print sum of all the elements.

→
```
num = [2, 4, 3, 20, 40]
sum = 0
for x in num:
    sum = sum + x.
print("sum of all elements of list is :", sum)
```

5) program to compare elements of list.

→
```
Comp = [10, 15, 17, 21, 30]
for i in range(len(comp) - 1)
    print("comparing {comp[i]} and {comp[i+1]} :
                Difference = {abs(comp[i] - comp[i+1])}")
```

PAGE NO.:
DATE : / /

6) Program to find maximum and minimum of list

→ numbers = [10, 45, 7, 69, 85]
print("Maximum element of list is", max(numbers))
print("Minimum element of list is", min(numbers))

7) Count the occurrence of element in list.

→ from collections import Counter
num = [2, 2, 7, 6, 5, 9, 5]
Print(num)
occur = Counter(num)
print (occur)

8) Reverse a list.

→ lst = ['Man', 'human', 'animal', birdy']
lst.reverse().

9) Write a loop that traverse the previous list & prints the length of each element. what happen if you send an integer to len.

→ lst = ["apple", "banana", "cherry"]
for item in list:
    print("length of '{item}' : ", len(item))

* Exercise 8 *

1) Create a tuple and sort it.

→ a = (20, 10, 50, 5, 90)
a.sort()
print("Sorted tuple is", a)

2) Create a tuple of number and print sum of all element

→ a = (2, 4, 6, 8, 10, 15)
sum=0
for i in a:
    sum=sum+i
print("Sum of elements is ", sum)

3) program to find Maximum & minimum of tuple

→ tup = (20, 90, 10, 95, 36, 12)
print("Maximum element is", Max(tup))
print("Minimum element is", Min(tup))

4) Count the occurence of element in tuple in a specific range

→ from collection import Counter
a = (10, 10, 20, 30, 35, 10)
occur = Counter(a)
print(occur)

5) Reverse a tuple

→ name = ('India', 'is', 'my', 'country')
print(name)
name.reverse()
print(name)

6) Write a loop that traverses the previous tuple & print length of each element. What happen if you send an floating number in index?

→ tup = ('banana', 'apple', 'pear', 'mango')
for item in tup:
    print("length of", {item}, "is", len(item))

* Exercise 9 *

1) python program to find power of a number using exponential ~~po~~ operator available in python

→ num = int(input("Enter a number"))
pow = int(input("Enter power of number"))
ans = num ** pow
print("Power of", num, "is", ans)

fibonacci number ⇒ if $5n^2+4$ & $5n^2-4$ are both perfect square or

PAGE NO.:
DATE: / /

2) Write a python program to check whethe given number is a fibonacci number or not if it is Fibonacci then print "Great" otherwise oops. "its' wrong".

→
```
import math
num = int(input("Enter a number:"))
sqrt1 = int(math.sqrt(5 * num * num + 4))
sqrt2 = int(math.sqrt(5*num *num -4))
if (sqrt1 * sqrt1 == (5 * num *num +4)) or (sqrt2*sqrt2==(5*num*num-4))
    print("Great")
else
    print("Opps! It's wrong").
```

3) check whether binary representation of a given number is a palindrome or not in if it is palindrome then print "Yes, IT Is", otherwise print "NO"?

→
```
num= int(input("Enter a number")
binary_rep = bin(num)[2:]
if  binary_rep == binary_rep[::-1]
    print("Yes, IT.IS")
else
    print("NO")
```

4) Find the numbers of integers from 1 to N which contain zero digit

→
```
N= int(input("Enter number")
count=0
for num in range(1, N):
    if '0' in str(num):
        count= count+1
print("Number of integers from 1 to", N, "which contain zero is:",
count)
```

PAGE NO.:
DATE: / /

5) Find the sum of all number below 1000 whic are multiples of 2 or 4

→ sum=0
for i in range(1, 1000)
if i%.2==0 or i%.4==0:
sum= sum+i
print("Sum of intenumbers multiple of 2 or 4 is :", sum)

* Exercise 10 *

1) Write program to create a class by name student & initialize attributes like name, age, and grade while creating an object.

→

```
class student:
    def __init__(self, name, age, grade)
        self.name = name
        self.age = age
        self.grade = grade
    def show(self)
        print("Name :", self.name)
        print("Age :", self.age)
        print("Grade :", self.grade)
obj = student('Madhuri', 20, 'A')
obj.show().
```

PAGE NO.:
DATE: / /

2) Write a program to create a child class Teacher that will inherit the properties of parent class Staff.

→
```
class staff:
    def __init__(self, Name, department):
        self.Name = Name
        self.dept = department
    def display(self):
        print("Name : {self.Name}, Department: {self.dept}")
class Teacher(staff):
    def __init__(self, Name, department, age, subject):
        super().__init__(Name, department)
        self.age = age
        self.dept = self.subj = subject
    def display_teacher(self):
        self.display()
        print(" Subject : {self.subject}, age : {self.age}")

teach = Teacher('Anjali', 'AIDS', '30', 'Maths')
teach.display_teacher()
```

3) Write a program, to create class and using the class instance print all the writable attributes of that object.

PAGE NO.:
DATE : / /

4) Create a class Teacher with name, age and salary attributes, where salary must be private attribute that cannot be accessed outside the class.

→
```
class Teacher
    def __init__(self, name, age, salary)
        self.name = name
        self.age = age
        self.__salary = salary
    def display(self)
        print("Name : {self.name}, Age : {self.age},
              Salary : {self.__salary}")
teach = Teacher('Anjali', 30, 55000)
teach.display()
```

6)

5) Write a python program that overloads the operator + and > for a custom class.

→
```
class over()
    def __init__(self, x):
        self.x = x
    def __add__(self, other):
        print("The value of obj1 is :", self.x)
        print("The value of obj2 is :", other.x)
        print("Addition of two object is :")
        return(self.x + other.x)
```

PAGE NO.:
DATE : / /

```
obj1 = over(20)
obj2 = over(10)
obj3 = obj1 + obj2.
if obj1 > obj2 :
    print(" obj1 is greater than obj2")
else:
    print(" obj1 is smaller than obj2").
```

6) Write a python class square, & define two methods that define two methods that return the square area & perimeter

→
```
class square:
    def __init(:-(self, side):
        self.side = side.
    def squared(self):
        return self.side **2
    def perimeter(self):
        return 4 * self.side
square1 = square(6)
print(" Area of square:", square1.area())
print("Perimeter of squar:", square1.perimeter()).
```