Expt. No.....
Date .....

Page No.....

# Implementation Of Real-Time / Technical Applications Using Lists And Tuples.

## Minimum Index Sum Of Two Lists

**Aim:**

To write a program to find minimum index sum of two lists.

**Program:**

```
def findRestaurant(self, list1, list2):
    a,m,k = [],[],[]
    v=0
    for i in list1:
        if j in list2:
            a.append(i)
        for i in a:
            m.append(list1.index(i)+list2.index(i))
        v.min(m)
        for i in range(0,(len(m)))
            if v==m[i]:
                k.append(a[i])
        return k.
```

**Output:** list1 = ["happy","sad","good"] list-2 = ["Sad", "happy","good"],["Sad","happy"]

Expt. No.....
Date .....

Page No.....

# Concatenate Two Lists Index-Wise.

**Aim:**

To write a program to concatenate two lists Index-Value.

**Program:**

```
L1 = ["Note", "tea", "cup"]
L2 = ["book", "cup", "board"]
L3 = []
for i in range(len(L1)):
    L3.append(L1[i] + L2[i])
print(L3)
```

**Output:**

```
["Notebook", "teacup", "cupboard"]
```

Expt. No.....
Date .....

Page No.....

# Copy Specific Elements From One Tuple To A New Tuple.

**Aim:**

To write a program to Copy Specific elements from one Tuple to a new Tuple.

**Program:**

```
t1 = ("apple","banana","Orange")
t2 = tuple()
element = input("Enter an element")
for i in t1:
    if(i==element)
        t2 = list(t2)
        t2 = append(element)
        t2 = tuple(t2)
        break.
Print(t2)
```

**Output:**

```
Enter an element : apple
("apple",)
```

Expt. No.....
Date .....

Page No.....

# TUPLE WITH SAME PRODUCT

AIM:

To write a program to find Tuple with Same Product.

PROGRAM:

```
def tupleSameProduct(self, nums: List[int]) -> int:
    count, ans, n = collections.Counter(), 0, len(nums)
    for i in range(n):
        for j in range(i+1, n):
            ans += 8 * count[nums[i] * nums[j]]
            count[nums[i] * nums[j]] += 1
    return ans
```

Output:

nums = [2, 3, 4, 6]

8

Expt. No.....
Date .....

Page No.....

# Implementation Of Real / Technical Applications Using Set And Dictionaries

## Magic Dictionary

**Aim:**

To write a program to find magic Dictionary

**Program:**

```
def __init__(self):
    self.dict = {}
def buildDict(self, dictionary: List[str])->None:
    for word in dictionary:
        for i, c in enumerate(word):
            replaced = self._getReplaced(word, i)
            Self.dict[replaced] = '*' if replaced in
                                  self.dict else c

def search(self, searchWord: str)-> bool:
    for i, c in enumerate(searchword):
        replaced = self._getReplaced(searchWord, i)
        if self.dict.get(replaced, c) != c:
            return True
    return False
def _getReplaced(self, s: str, i: int)->str:
    return s[:i] + '*' + s[i+1:]
```

**Output:**

["MagicDictionary", "buildDict", "search", "search", "search", "search"]
output : [null, null, false, true, false, false]

Expt. No.....
Date .....

Page No.....

# LONGEST WORD IN DICTIONARY

**AIM:**

To write a Program to find longest word in Dictionary.

**PROGRAM:**

```
m={}
for i,l in enumerate(s):
    if l not in m:
        m[l]=[]
    m[l].append(i)
Candidates=[]
for w in d:
    lastEnc=-1
    for i,l in enumerate(w):
        if l not in m:
            break
        pos=bisect_left(m[l],lastEnc)
        if pos>=len(m[l]):
            break
        lastEnc=m[l][pos]+1
        if i==len(w)-1:
            Candidates.append(w)
Candidates.sort()
Candidates.sort(key=lambda x: len(x),
reverse=True)
if Candidates:
    return Candidates[0]
else: return
```

Expt. No.....
Date .....

Page No.....

Output:

S= ["Smallest Infi"] dictionary = ["ale","appli","monkey","plea"]

Output : "apple"

Expt. No.....
Date .....

Page No.....

# SET MISMATCH.

AIM:

To write a program to find Set mismatch.

PROGRAM:

```
import Statistics
from Statistics import mode
Class Solution:
    def findErrorNums(self, nums: List[int]) -> List[int]:
        l = []
        s = Set(range(1, len(nums)+1))
        l.append(mode(nums))
        l.append((list(s - Set(nums)))[0])
        return l
```

Output:

nums = [1, 2, 2, 4]

Output = [2, 3].

Expt. No.....
Date .....

Page No.....

# SMALLEST NUMBER IN FINITE SET

AIM:

To write a program to find Smallest number in Finite Set.

PROGRAM:

```
Class SmallestInfiniteSet:
    def __init__(self):
        self.nums = [i for i in range(1, 1001)]
        self.popped_integers = set()
    def popSmallest(self) -> int:
        num = self.nums.pop(0)
        self.popped_integers.add(num)
        return num
    def addBack(self, num: int) -> None:
        if num in self.popped_integers:
            self.popped_integers.remove(num)
            self.nums.append(num)
            self.nums.sort()
```

OUTPUT:

["SmallestInfiniteSet", "addBack", "popSmallest", "popSmallest", "popSmallest", "addBack", "popSmallest", "popSmallest", "popSmallest"]

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

Expt. No.....
Date .....

Page No.....

# Implementation Of Functions In The Program. Factorial

**Aim:**

To write a program to find factorial of a number.

**Program:**

```
def factorial(n):
    if n==0:
        return 1
    else
        return n* factorial(n-1)
number = int(input('Enter the number:'))
result = factorial(number)
print(result).
```

**Ouput:**

number = 3

Ouput = 6.

Expt. No.....
Date .....

Page No.....

# LARGEST NUMBER IN A LIST.

**AIM:**

To write a program to find the Largest Number in a list.

**PROGRAM:**

```
def find_largest_number(numbers):
    if len(numbers)==0:
        return None
    largest = numbers[0]
    for numbers in numbers:
        if number > largest:
            largest = number
    return largest.
my_list = [5,10,3,8,15,1]
largest_number = find_largest_number(my_list)
print("The largest number in the list is:",
                                largest number)
```

**Output:**

15 → largest Number in a list.

Expt. No.....
Date .....

Page No.....

# AREA OF SHAPE:

**AIM:**

To write a program to find Area of Shape.

**PROGRAM:**

```
def Square(a):
    return a*a
def triangle(a,b):
    return 0.5*a*b
def rectangle(a,b):
    return a*b
print("Shapes are \n 1. Square \n 2. triangle
    \n 3. circle \n 4. rectangle \n")
n=int(input("Enter shape no").
if(n==3):
    a=int(input("Enter radius"))
    print("Area of Circle", circle(a))
elif n==1:
    a=int(input("Enter Sides"))
    print("Area of Square", Square(a))
elif n==2:
    a=int(input("Enter breadth")
    b=int(input("Enter length")
    print("Area of triangle", triangle(a,b))
elif n==4:
    a=int(input("Enter breath"))
    b=int(input("Enter length"))
    print("Area of rectangle", rectangle(a,b))
```

Expt. No.....
Date .....

Page No.....

```
else
    print ("Invalid Shape no").
```

Output:

Enter Shape no : 1 Enter radius : 5
Area of Square : 25.