Program :1   Python Numbers and List operations

Aim : To implement Numbers and lit operations

Algorithm:

1. Python Numbers and List operations

```
# Numbers integer, float and complex
seller-name="RAJA"
age=28
salary=20000.500
c=5j
print(type(seller-name))
print(type(age))
print(type(c))

fruits=["Apple","Banana","Orange", "Grapes"]
print("Fruits List:", fruits)

l=len(fruits)
print("Length of the List:",l)
fruits.append("Cherry")
print("Updated Fruits List:", fruits)
l=len(fruits)
print("Length of the List:",l)
fruits.sort()
print("Sorted Fruits List:", fruits)
# Search pine-apple in fruits list
```

```
print("Search pineapple in fruits list")
s="Pineapple" in fruits
print(s)
print("Search apple in fruits list")
s="Apple" in fruits
print(s)

print("Fruits List after deleting", fruits[3],":")
del fruits[3]
print(fruits)
```

Program 2. ***Tuple, Strings, Set***

# STRING HANDLING

```
name="SRM University"
name2="Trichy"
print("First Name:", name)
print("Second Name:", name2)
print("The first letter of a given Name")
print(name[0])
print("The first letter of a given Name2")
print(name2[0])
print(" " )
print("CONCATENATION OF TWO STRINGS")
print(name+name2)
print("Length of name-string:",name, len(name))
print("UPPER CASE: ", s.upper())
print("Lower Caser: ", s.lower())
```

```
# LIST CREATION
fruits=('Apple', 'Mango', 'Banana')
#print("Try to append the content of tuples")
#fruits.append('Orange') # Not possible
print("Elements of Tuple(Fruits)")
print(fruits)
print("Length of Tuple-Fruits")
print(len(fruits))
print("Max of Fruits:", max(fruits))
```

```
# SET FUNCTION / OPERATION (add, remove), length(len)
print("SET CREATION AND OPERATIONS/ FUNCTIONS")
fruits=set(["Apple", "Mango", "Banana"])
print("Fruit-SET", fruits)
print("")
print("Add 'Organge' in the set")
fruits.add("Orange")
print(fruits)
print("")
print("Remove 'Banana' from set")
fruits.remove("Banana")
print(fruits)
print("COUNT ITEMS IN THE SET")
print("No. of Items in the set:", len(fruits))
print("Search 'Papaya' in the set")

print("Papaya" in fruits)

print("Serach 'Apple'in the Set")

print("Apple" in fruits)
```

Program 3: Lambda & Filter

# Lamda and Filter Implementation

```
n=[10,15,20,25,30,35,40,45,50]
Even=list(filter(lambda x: x%2==0 , n))

print("Filtered Even Numbers are:",Even)
```

OUTPUT

Program 4: Creating Class in Python

```
# define a class
class Student:
    name = ""
    regno = ""
    year = ""
    GCPA = 0

# create object of class
s1=Student()

# access attributes and assign new values
s1.name="JANANE"
s1.regno="RA240001"
s1.year="I-MCA"
s1.CGPA=9.8
print(f"Name: {s1.name}, Register no.:{s1.regno}, Year:{s1.year}, CGPA:
{s1.CGPA}")
```

Program 5: ***Creating Object in Python***

```
# define a class
class Student:
    name = ""
    regno = ""
    year = ""
    CGPA = 0.0

# Create a list to store student objects
students = []

# Add data for more than 5 students
students.append(Student())
students[0].name = "JANANE"
students[0].regno = "RA240001"
students[0].year = "I-MCA"
students[0].CGPA = 9.8

students.append(Student())
students[1].name = "buvaneshwari"
students[1].regno = "RA240002"
students[1].year = "I-MCA"
students[1].CGPA = 9.8

students.append(Student())
students[2].name = "ALBIN"
students[2].regno = "RA240003"
students[2].year = "I-MCA"
students[2].CGPA = 9.7

students.append(Student())
students[3].name = "ALBERT"
students[3].regno = "RA240004"
students[3].year = "I-MCA"
students[3].CGPA = 9.6

students.append(Student())
students[4].name = "SITA"
students[4].regno = "RA240005"
students[4].year = "I-MCA"
```

```
students[4].CGPA = 9.9

# Print data for all students
for student in students:
    print(f"Name: {student.name}, Register no.: {student.regno}, Year:
{student.year}, CGPA: {student.CGPA}")
```

Program 6: Creating Methods in Python

```
# function with two arguments

def AreaRect(length,width):
    Area= length * width

    return Area

# function call with two values
A= AreaRect(15, 5)
print("Area of the Rectangle: ", A)
```

Program 7: Process standard streams.

```
def take_order():

    """Step 1: Take customer order details."""

    print("Taking customer order...")

    order = {"customer": "Alice", "item": "Laptop", "quantity": 1}

    return order


def process_payment(order):

    """Step 2: Process the payment."""

    print(f"Processing payment for {order['customer']}...")
    order["payment_status"] = "Paid"
    return order

def prepare_order(order):
    """Step 3: Prepare the order for shipping."""
    if order["payment_status"] == "Paid":
        print(f"Preparing {order['item']} for shipping...")
        order["order_status"] = "Ready for Shipment"
    return order

def ship_order(order):
    """Step 4: Ship the order to the customer."""
    if order["order_status"] == "Ready for Shipment":
        print(f"Shipping {order['item']} to {order['customer']}...")
        order["delivery_status"] = "Shipped"
    return order

# **Process Execution: Order Workflow**
order = take_order()
order = process_payment(order)
order = prepare_order(order)
order = ship_order(order)

print("\nFinal Order Status:", order)
```

Program 8: ***Command-line arguments, shell variables***

```
import sys

# sys.argv[0] is the script name

print("Script name:", sys.argv[0])


# Command-line arguments

if len(sys.argv) > 1:

    print("Arguments:", sys.argv[1:])

else:

    print("No arguments provided.")
```

Program 14:Dictionaries.

```
# DICTIONARY AND FUNCTIONS : SIMILAR TO LIST , any data types can be used


d={}

d[0]='Apple'

d[1]='Banana'

d[2]='Mango'

d[3]='Orange'

print("Elements of dictionary D")

print(d)

print("Length of dictionary")

print(len(d))
print("Max of dictionary:", max(d))

print("Minimum value of dictionary:", min(d))

print("Serach Orange in dictionary:")

print(3 in d)


print("Serach Papaya in dictionary")

print('Papaya' in d)
```

Program 15 : ***Read and write data from/to files in Python Programs***

```
# Standard: PEP 8 (Proper indentation, naming conventions, and docstrings)


defwritefile(filename, content):
    """Write data to a file (Output Stream - Writing Mode)."""
    with open(filename, "w") as file:
file.write(content)
print(f"Data written to {filename}")


defreadfile(filename):
    """Rfad data from a file (Input Stream - Reading
Mode)."""
    try:
        with open(filename, "r") as file:
            data = file.read()
print(f"Data read from {filename}: \n{data}")
    except FileNotFoundError:
print(f"Error: {filename} not found.")


defappendfile(filename, content):
    """Append data to an existing file (Output
Stream - Append Mode)."""
    with open(filename, "a") as file:
file.write("\n" + content)
print(f"Data appended to {filename}")


# Streams Example: File I/O Operations
```

```
filename = "D:\kk\MCADATA.txt"


# Writing to a file (Output Stream)
writefile(filename, "HI MCA STUDENTS,
WELCOME TO SRM IST TO LEARN PYTHON")
```