

PYTHON

1. Implement factorial using recursion

```
def recur_factorial(n):  
    if n == 1:  
        return n  
    else:  
        return n*recur_factorial(n-1)  
num =7  
if num < 0:  
    print("no", factorial does not exist for negative numbers")  
elif num == 0:  
    print("The factorial of 0 is 1")  
else:  
    print("The factorial of", num, "is", recur_factorial(num))
```

2.Program using doubly linked list

```
class Node:  
  
    def __init__(self, data):  
  
        self.item = data  
  
        self.next = None  
  
        self.prev = None  
  
class doublyLinkedList:  
  
    def __init__(self):  
  
        self.start_node = None  
  
    def InsertToEmptyList(self, data):  
  
        if self.start_node is None
```

```
new_node = Node(data)

self.start_node = new_node

else:

    print("The list is empty")

def InsertToEnd(self, data):

    if self.start_node is None:

        new_node = Node(data)

        self.start_node = new_node

        return

    n = self.start_node

    while n.next is not None:

        n = n.next

    new_node = Node(data)

    n.next = new_node

    new_node.prev = n

def DeleteAtStart(self):

    if self.start_node is None:

        print("The Linked list is empty, no element to delete")

        return

    if self.start_node.next is None:

        self.start_node = None

        return 0
```

```

self.start_node = self.start_node.next

self.start_prev = None;

def delete_at_end(self):

    if self.start_node is None:

        print("The Linked list is empty, no element to delete")

        return

    if self.start_node.next is None:

        self.start_node = None

        return

    n = self.start_node

    while n.next is not None:

        n = n.next

    n.prev.next = None

def Display(self):

    if self.start_node is None:

        print("The list is empty")

        return

    else:

        n = self.start_node

        while n is not None:

            print("Element is: ", n.item)

            n = n.next

```

```
print("\n")
NewDoublyLinkedList = doublyLinkedList()
NewDoublyLinkedList.InsertToEmptyList(10)
NewDoublyLinkedList.InsertToEnd(20)
NewDoublyLinkedList.InsertToEnd(30)
NewDoublyLinkedList.InsertToEnd(40)
NewDoublyLinkedList.InsertToEnd(50)
NewDoublyLinkedList.InsertToEnd(60)
NewDoublyLinkedList.Display()
NewDoublyLinkedList.DeleteAtStart()
NewDoublyLinkedList.DeleteAtStart()
NewDoublyLinkedList.Display()
```

3.Implement multiple inheritance using interface

```
class Mammal:
    def mammal_info(self):
        print("Mammals can give direct birth.")

class WingedAnimal:
    def winged_animal_info(self):
        print("Winged animals can flap.")

class Bat(Mammal, WingedAnimal):
    pass
b1 = Bat()

b1.mammal_info()
```

```
b1.winged_animal_info()
```

4.Print all pronic numbers between 1 and 100.

```
def isPronicNumber(num):  
    flag = False;  
    for j in range(1, num+1):  
        if((j*(j+1)) == num):  
            flag = True;  
            break;  
    return flag;  
  
    print("Pronic numbers between 1 and 100: ");  
for i in range(1, 101):  
    if(isPronicNumber(i)):  
        print(i),  
        print(" "),
```

5.Implement method overloading & overriding in python.

```
class A:  
    def fun1(self):  
        print('feature_1 of class A')  
    def fun2(self):  
        print('feature_2 of class A')  
    class B(A):  
        def fun1(self):
```

```
        print('Modified feature_1 of class A by class B')

    def fun3(self):
        print('feature_3 of class B')

obj = B()

obj.fun1()
```

6.Program to find duplicate values for ArrayList.

```
arr = [1, 2, 3, 4, 2, 7, 8, 8, 3];

print("Duplicate elements in given array:

for i in range(0, len(arr)):
    for j in range(i+1, len(arr)):
        if(arr[i] == arr[j]):
            print(arr[j]);
```

7. Python program to print the elements of an array in reverse order.

```
arr = [1, 2, 3, 4, 5];

print("Original array: ");

for i in range(0, len(arr)):
    print(arr[i]),

print("Array in reverse order: ");

#Loop through the array in reverse order

for i in range(len(arr)-1, -1, -1):
    print(arr[i]),
```

8. Python program to determine whether the given number is a Harshad Number

```
num = 156;

rem = sum = 0;

n = num;

while(num > 0):

    rem = num%10;

    sum = sum + rem;

    num = num//10;

if(n%sum == 0):

    print(str(n) + " is a harshad number");

else:

    print(str(n) + " is not a harshad number");
```

9. Implement a program to merge two Arrays.

```
def find(array1, array2, n1, n2):

    for i in array2:

        array1.append(i)

    array1 = list(set(sorted(array1)))

    array2 = array1[len(array1) - n2:]

    array1 = array1[:len(array1) - n2]

print("After")

print("Array1: ", array1, "\nArray2: ", array2)
```

```
array1 = [1, 2, 3, 5, 8, 9, 10, 13, 15, 20]
array2 = [2, 3, 8, 13]
print("Before: ")
print("Array1: ", array1)
print("Array2: ", array2)
find(array1, array2, len(array1), len(array2))
```

10.Program to find duplicate values for ArrayList.

```
arr = [1, 2, 3, 4, 2, 7, 8, 8, 3];
print("Duplicate elements in given array: ");
for i in range(0, len(arr)):
    for j in range(i+1, len (arr)):
        if(arr[i] == arr[j]):
            print(arr[j]);
```

11.Implement a program to sort a map by value / Key

```
myDict = {'ravi': 10, 'rajnish': 9,
'sanjeev': 15, 'yash': 2, 'suraj': 32}

myKeys = list(myDict.keys())
myKeys.sort()
sorted_dict = {i: myDict[i] for i in myKeys}

print(sorted_dict)
```

12.Write a python Program for Fibonacci series.

```
def Fibonacci(n):
```



```
if n < 0:
    print("Incorrect input")
elif n == 0:
    return 0
elif n == 1 or n == 2:
    return 1
else:
    return Fibonacci(n-1) + Fibonacci(n-2)
print(Fibonacci(9))
```

13. Python program to print the elements of an array in reverse order.

```
arr = [1, 2, 3, 4, 5];
print("Original array: ");
for i in range(0, len(arr)):
    print(arr[i]),
print("Array in reverse order: ");
for i in range(len(arr)-1, -1, -1):
    print(arr[i]),
```

14. Write a python Program for Fibonacci series.

```
def Fibonacci(n):
    if n < 0:
        print("Incorrect input")
    elif n == 0:
```

```
        return 0

    elif n == 1 or n == 2:

        return 1

    else:

        return Fibonacci(n-1) + Fibonacci(n-2)

print(Fibonacci(9))
```

15.Constructor Overloading.

The constructor overloading can be defined as the concept of having more than one constructor with different parameters so that every constructor can perform a different task.

16.Python program to print the elements of an array in reverse order.

```
arr = [1, 2, 3, 4, 5];

print("Original array: ");

for i in range(0, len(arr)):

    print(arr[i]),

print("Array in reverse order: ");

for i in range(len(arr)-1, -1, -1):

    print(arr[i]),
```

17.Implement Exception Handling without Catch block.

```
try:

    numerator = 10

    denominator = 0
```

```
result = numerator/denominator
```

```
print(result)
```

```
except:
```

```
    print("Error: Denominator cannot be 0.")
```

18. Python program to determine whether the given number is a Harshad Number.

```
num = 156;
```

```
rem = sum = 0;
```

```
n = num;
```

```
while(num > 0):
```

```
    rem = num%10;
```

```
    sum = sum + rem;
```

```
    num = num//10;
```

```
if(n%sum == 0):
```

```
    print(str(n) + " is a harshad number");
```

```
else:
```

```
    print(str(n) + " is not a harshad number");
```

19. Compare StringBuffer with a string.

String is an immutable class and its object can't be modified after it is created but definitely reference other objects. They are very useful in multithreading

environment because multiple threads can't change the state of the object so immutable objects are thread safe.

String buffer is mutable classes which can be used to do operation on string object such as reverse of string, concating string and etc. We can modify string without creating new object of the string. String buffer is also thread safe. Also, string concat + operator internally uses StringBuffer or StringBuilder class.

20. Python program to print the elements of an array in reverse order.

```
arr = [1, 2, 3, 4, 5];  
print("Original array: ");  
for i in range(0, len(arr)):  
    print(arr[i]),  
print("Array in reverse order: ");  
for i in range(len(arr)-1, -1, -1):  
    print(arr[i]),
```

21. remove duplicates from sorted array.

```
def removeDuplicates(arr, n)  
    if n == 0 or n == 1:  
        return n  
    temp = list(range(n))  
    j = 0  
    for i in range(0, n-1):  
        if arr[i] != arr[i+1]:
```

```

        temp[j] = arr[i]

        j += 1

temp[j] = arr[n-1]

j += 1

for i in range(0, j):

    arr[i] = temp[i]


return j

if __name__ == '__main__':

    arr = [1, 2, 2, 3, 4, 4, 4, 5, 5]

    n = len(arr)

    n = removeDuplicates(arr, n)

    for i in range(n):

        print("%d" % (arr[i]), end=" ")

```

22. Python program to print the elements of an array in reverse order.

```

arr = [1, 2, 3, 4, 5];

print("Original array: ");

for i in range(0, len(arr)):

    print(arr[i]),

print("Array in reverse order: ");

for i in range(len(arr)-1, -1, -1):

    print(arr[i]),

```

23.swap two numbers without using temporary variable.

```
x = 10  
  
y = 5  
  
x = x ^ y;  
  
y = x ^ y;  
  
x = x ^ y;  
  
print ("After Swapping: x = ", x, " y =", y)
```

24.Python program to create a doubly linked list from a ternary tree.

```
class Node:
```

```
    def __init__(self,data):
```

```
        self.data = data;
```

```
        self.left = None;
```

```
        self.middle = None;
```

```
        self.right = None;
```

```
class TernaryTreeToDLL:
```

```
    def __init__(self):
```

```
        self.root = None;
```

```
        self.head = None;
```

```
        self.tail = None;
```

```
    def convertTernaryToDLL(self, node):
```

```
if(node == None):
    return;
left = node.left;
middle = node.middle;
right = node.right;
if(self.head == None):
    self.head = self.tail = node;
    node.middle = None;
    self.head.left = None;
    self.tail.right = None;
else:
    self.tail.right = node;
    node.left = self.tail;
    node.middle = None;
    self.tail = node;
    self.tail.right = None;
self.convertTernaryToDLL(left);
self.convertTernaryToDLL(middle);
self.convertTernaryToDLL(right);
def displayDLL(self):
    current = self.head;
    if(self.head == None):
```

```
        print("List is empty");  
  
        return;  
  
    print("Nodes of generated doubly linked list: ");  
  
    while(current != None):  
        print(current.data),  
        current = current.right;
```

```
tree = TernaryTreeToDLL();  
tree.root = Node(5);  
tree.root.left = Node(10);  
tree.root.middle = Node(12);  
tree.root.right = Node(15);  
tree.root.left.left = Node(20);  
tree.root.left.middle = Node(40);  
tree.root.left.right = Node(50);  
tree.root.middle.left = Node(24);  
tree.root.middle.middle = Node(36);  
tree.root.middle.right = Node(48);  
tree.root.right.left = Node(30);  
tree.root.right.middle = Node(45);  
tree.root.right.right = Node(60);  
tree.convertTernaryToDLL(tree.root);
```



```
tree.displayDLL();
```

25.Find Maximum repeated charcter count in a string.

```
ASCII_SIZE = 256
```

```
def getMaxOccurringChar(str):
```

```
    count = [0] * ASCII_SIZE
```

```
    max = -1
```

```
    c = ""
```

```
    for i in str:
```

```
        count[ord(i)] += 1
```

```
    for i in str:
```

```
        if max < count[ord(i)]:
```

```
            max = count[ord(i)]
```

```
            c = i
```

```
    return c
```

```
str = "sample string"
```

```
print("Max occurring character is", getMaxOccurringChar(str))
```

26.Implement quick sorting.

```
def partition(array, low, high):
```

```
    pivot = array[high]
```

```
    i = low - 1
```

```

for j in range(low, high):
    if array[j] <= pivot:
        i = i-1

    (array[i], array[j]) = (array[j], array[i])

(array[i + 1], array[high]) = (array[high], array[i + 1])

return i + 1

```

```

def quickSort(array, low, high):
    if low < high:
        pi = partition(array, low, high)
        quickSort(array, pi + 1, high)

```

```

data = [8, 7, 2, 1, 0, 9, 6]

```

```

print("Unsorted Array")

```

```

print(data)

```

```

size = len(data)

```

```

quickSort(data, 0, size - 1)

```

```

print('Sorted Array in Ascending Order:')

```

```

print(data)

```

27.Find duplicate elements in a string.

```

def duplicate_characters(string):
    chars = {}

    for char in string:
        if char not in chars:

```

```
        chars[char] = 1
    else:
        chars[char] += 1

duplicates = []
for char, count in chars.items():
    if count > 1:
        duplicates.append(char)

return duplicates

print(duplicate_characters("engineering"))
```

28. Python program to determine whether the given number is a Harshad Number.

```
num = 156;
rem = sum = 0;
n = num;
while(num > 0):
    rem = num%10;
    sum = sum + rem;
    num = num//10;
if(n%sum == 0):
    print(str(n) + " is a harshad number");
else:
    print(str(n) + " is not a harshad number");
```

29. Programs with list and tuples .

```
list_ = ["Python", "Lists", "Tuples", "Differences"]  
tuple_ = ("Python", "Lists", "Tuples", "Differences")  
list_[3] = "Mutable"  
print( list_ )  
  
try:  
    tuple_[3] = "Immutable"  
    print( tuple_ )  
except TypeError:  
    print( "Tuples cannot be modified because they are immutable" )
```

30. Implement dictionary.

```
my_information = {'name': 'Dionysia', 'age': 28, 'location': 'Athens'}  
print(my_information)  
print(type(my_information))
```

31. Python program to determine whether the given number is a Harshad Number.

```
num = 156;  
rem = sum = 0;  
n = num;  
while(num > 0):  
    rem = num%10;  
    sum = sum + rem;
```

```
num = num//10;

if(n%sum == 0):

    print(str(n) + " is a harshad number");

else:

    print(str(n) + " is not a harshad number");
```

32.Python program to create a doubly linked list from a ternary tree.

```
class Node:

    def __init__(self,data):

        self.data = data;

        self.left = None;

        self.middle = None;

        self.right = None;

class TernaryTreeToDLL:

    def __init__(self):

        self.root = None;

        self.head = None;

        self.tail = None;

    def convertTernaryToDLL(self, node):

        if(node == None):

            return;
```

```
left = node.left;

middle = node.middle;

right = node.right;

if(self.head == None):

    self.head = self.tail = node;

    node.middle = None;

    self.head.left = None;

    self.tail.right = None;

else:

    self.tail.right = node;

    node.left = self.tail;

    node.middle = None;

    self.tail = node;

    self.tail.right = None;

self.convertTernaryToDLL(left);

self.convertTernaryToDLL(middle);

self.convertTernaryToDLL(right);

def displayDLL(self):

    current = self.head;

    if(self.head == None):

        print("List is empty");

        return;
```

```

        print("Nodes of generated doubly linked list: ");

        while(current != None):

            print(current.data),

            current = current.right;


tree = TernaryTreeToDLL();

tree.root = Node(5);

tree.root.left = Node(10);

tree.root.middle = Node(12);

tree.root.right = Node(15);

tree.root.left.left = Node(20);

tree.root.left.middle = Node(40);

tree.root.left.right = Node(50);

tree.root.middle.left = Node(24);

tree.root.middle.middle = Node(36);

tree.root.middle.right = Node(48);

tree.root.right.left = Node(30);

tree.root.right.middle = Node(45);

tree.root.right.right = Node(60);

tree.convertTernaryToDLL(tree.root);

tree.displayDLL();

```

33.compare two arrays and return the common elements.

```
import numpy as np

ar1 = np.array([0, 1, 2, 3, 4])

ar2 = [1, 3, 4]

print(np.intersect1d(ar1, ar2))
```

34. Write a python Program to find whether a string or number is palindrome or not.

```
def isPalindrome(s):

    return s == s[::-1]

s = "malayalam"

ans = isPalindrome(s)

if ans:

    print("Yes")

else:

    print("No")
```

35. Implement more than one interface in a single class.

```
from abc import ABC, abstractmethod

class Interface1(ABC):

    def method1(self):

        pass

class Interface2(ABC):

    def method2(self):

        pass
```



```
class MyClass(Interface1, Interface2):  
    def method1(self):  
        print("Implementation of method1")  
    def method2(self):  
        print("Implementation of method2")
```

36. Python program to determine whether the given number is a Harshad Number.

```
num = 156;  
rem = sum = 0;  
n = num;  
while(num > 0):  
    rem = num%10;  
    sum = sum + rem;  
    num = num//10;  
if(n%sum == 0):  
    print(str(n) + " is a harshad number");  
else:  
    print(str(n) + " is not a harshad number");
```

37. Implement a program for encapsulation.

```
class Base:  
    def __init__(self):
```

```

        self._a = 2

class Derived(Base):

    def __init__(self):

        Base.__init__(self)

        print("Calling protected member of base class: ",

              self._a)

        self._a = 3

        print("Calling modified protected member outside class: ",

              self._a)

obj1 = Derived()

obj2 = Base()

print("Accessing protected member of obj1: ", obj1._a)

print("Accessing protected member of obj2: ", obj2._a)

```

38.Print all pronic numbers between 1 and 100.

```

def isPronicNumber(num):

    flag = False;

    for j in range(1, num+1):

        if((j*(j+1)) == num):

            flag = True;

            break;

    return flag;

print("Pronic numbers between 1 and 100: ");

```

```
for i in range(1, 101):  
    if(isPronicNumber(i)):  
        print(i),  
        print(" "),
```

39.convert string to char and vice versa.

To char;

```
my_string = "hello"  
char_list = list(my_string)  
print(char_list)
```

To string;_____

```
char_list = ['h', 'e', 'l', 'l', 'o']  
my_string = ''.join(char_list)  
print(my_string)
```

40. Iterate the LinkedHashMap values.

```
from collections import OrderedDict  
  
my_dict = OrderedDict()  
  
my_dict['key1'] = 'value1'  
  
my_dict['key2'] = 'value2'  
  
my_dict['key3'] = 'value3'  
  
for value in my_dict.values():  
    print(value)
```

42) Print all pronic numbers between 1 and 100.

```
for i in range(1, 101):
```

```
    if i*(i+1) <= 100:
```

```
        print(i*(i+1))
```

```
    else:
```

```
        break
```

43) Implement a program to handle more than one exception

```
try:
```

```
    num = int(input("Enter a number: "))
```

```
    result = 10 / num
```

```
    print("Result:", result)
```

```
except ZeroDivisionError:
```

```
    print("Cannot divide by zero!")
```

```
except ValueError:
```

```
    print("Please enter a valid integer!")
```

```
except Exception as e:
```

```
    print("An error occurred:", e)
```

```
finally:
```

```
    print("Program finished")
```

44. Python program to create a doubly linked list from a ternary tree.

```
class Node:
```

```
    def __init__(self, data):
```

```
        self.data = data;
```

```
self.left = None;

self.middle = None;

self.right = None;
```

```
class TernaryTreeToDLL:
```

```
    def __init__(self):

        self.root = None;

        self.head = None;

        self.tail = None;
```

```
    def convertTernaryToDLL(self, node):
```

```
        if(node == None):

            return;

        left = node.left;

        middle = node.middle;

        right = node.right;

        if(self.head == None):

            self.head = self.tail = node;

            node.middle = None;

            self.head.left = None;

            self.tail.right = None;

        else:
```

```

        self.tail.right = node;

        node.left = self.tail;

        node.middle = None;

        self.tail = node;

        self.tail.right = None;

    self.convertTernaryToDLL(left);

    self.convertTernaryToDLL(middle);

    self.convertTernaryToDLL(right);

def displayDLL(self):

    current = self.head;

    if(self.head == None):

        print("List is empty");

        return;

    print("Nodes of generated doubly linked list: ");

    while(current != None):

        print(current.data),

        current = current.right;

tree = TernaryTreeToDLL();

tree.root = Node(5);

tree.root.left = Node(10);

tree.root.middle = Node(12);

```

```
tree.root.right = Node(15);
tree.root.left.left = Node(20);
tree.root.left.middle = Node(40);
tree.root.left.right = Node(50);
tree.root.middle.left = Node(24);
tree.root.middle.middle = Node(36);
tree.root.middle.right = Node(48);
tree.root.right.left = Node(30);
tree.root.right.middle = Node(45);
tree.root.right.right = Node(60);
tree.convertTernaryToDLL(tree.root);
tree.displayDLL();
```

45.Convert arraylist into string

```
my_list = ['apple', 'banana', 'orange', 'pear']
delimiter = ','
result = delimiter.join(my_list)
print(result)
```

46) Python program to determine whether the given number is a Harshad Number

```
num = int(input("Enter a number: "))
```

```
sum_of_digits = sum(int(digit) for digit in str(num))
```

```
if num % sum_of_digits == 0:
```

```
    print(num, "is a Harshad number")
```

```
else:
```

```
    print(num, "is not a Harshad number")
```

47) Convert a set to stream.

```
my_set = {'apple', 'banana', 'orange', 'pear'}
```

```
stream = (item for item in my_set)
```

```
for item in stream:
```

```
    print(item)
```

48) Python program to create a doubly linked list from a ternary tree

```
class Node:
```

```
    def __init__(self,data):
```

```
        self.data = data;
```

```
        self.left = None;
```

```
        self.middle = None;
```

```
        self.right = None;
```

```
class TernaryTreeToDLL:
```

```
    def __init__(self):
```

```
        self.root = None;
```

```
        self.head = None;
```



```
self.tail = None;
```

```
def convertTernaryToDLL(self, node):
```

```
    if(node == None):
```

```
        return;
```

```
    left = node.left;
```

```
    middle = node.middle;
```

```
    right = node.right;
```

```
    if(self.head == None):
```

```
        self.head = self.tail = node;
```

```
        node.middle = None;
```

```
        self.head.left = None;
```

```
        self.tail.right = None;
```

```
    else:
```

```
        self.tail.right = node;
```

```
        node.left = self.tail;
```

```
        node.middle = None;
```

```
        self.tail = node;
```

```
        self.tail.right = None;
```

```
    self.convertTernaryToDLL(left);
```

```
    self.convertTernaryToDLL(middle);
```

```
    self.convertTernaryToDLL(right);
```

```
def displayDLL(self):  
    current = self.head;  
  
    if(self.head == None):  
        print("List is empty");  
        return;  
  
    print("Nodes of generated doubly linked list: ");  
  
    while(current != None):  
        print(current.data),  
        current = current.right;
```

```
tree = TernaryTreeToDLL();  
tree.root = Node(5);  
tree.root.left = Node(10);  
tree.root.middle = Node(12);  
tree.root.right = Node(15);  
tree.root.left.left = Node(20);  
tree.root.left.middle = Node(40);  
tree.root.left.right = Node(50);  
tree.root.middle.left = Node(24);  
tree.root.middle.middle = Node(36);  
tree.root.middle.right = Node(48);  
tree.root.right.left = Node(30);
```

```
tree.root.right.middle = Node(45);  
tree.root.right.right = Node(60);  
tree.convertTernaryToDLL(tree.root);  
tree.displayDLL();
```

49) Write a program in python to check whether number is palindrom or not using recursive method?

```
def is_palindrome(num):  
    if num // 10 == 0:  
        return True  
    else:  
        first_digit = num % 10  
        last_digit = num // (10 ** (len(str(num)) - 1))  
        if first_digit == last_digit:  
            remaining_num = (num - (last_digit * (10 ** (len(str(num)) - 1)))) // 10  
            return is_palindrome(remaining_num)  
        else:  
            return False  
  
num = int(input("Enter a number: "))  
if is_palindrome(num):  
    print(num, "is a palindrome")  
else:
```

```
print(num, "is not a palindrome")
```

50) Swap two numbers without using third variable.

```
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
print("Before swapping: a =", a, ", b =", b)

a = a + b
b = a - b
a = a - b

print("After swapping: a =", a, ", b =", b)
```

52) Write a program to print all the prime numbers between two numbers Between 1 and 10 between 20 to 30.

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

def print_primes(start, end):
    for num in range(start, end+1):
        if is_prime(num):
```

```
        print(num)

print("Prime numbers between 1 and 10:")

print_primes(1, 10)

print("Prime numbers between 20 and 30:")

print_primes(20, 30)
```

53) Write a program to check the string is palindrome or not.

```
strings = ["Madam", "wow", "cycle"]

for string in strings:

    string = string.lower().replace(" ", "")

    if string == string[::-1]:

        print(string, "is a palindrome")

    else:

        print(string, "is not a palindrome")
```

54) 5. Write a program to print pattern?

Input= 4:

```
n = 4

for i in range(n):

    for j in range(n-i):

        print("*", end="")

    print()
```

Input=5:

```
n = 5
```

```
for i in range(n):
```

```
    for j in range(n-i):
```

```
        print("*", end="")
```

```
    print()
```

55) Write a program to check the vowels in the string.

```
string = "Codoid innovations"
```

```
vowels = 'aeiouAEIOU'
```

```
for char in string:
```

```
    if char in vowels:
```

```
        print(char, "is a vowel")
```

56) Remove the duplicate elements in the array without using builtin function.

```
[ 5,4,10,20,4,6,10,39,4,39]
```

```
arr = [5, 4, 10, 20, 4, 6, 10, 39, 4, 39]
```

```
new_arr = []
```

```
for num in arr:
```

```
    if num not in new_arr:
```

```
        new_arr.append(num)
```

```
print("Original array:", arr)
```

```
print("Array with duplicates removed:", new_arr)
```

57) Find the largest number in the array (without using pre define functions)

```
n = int(input("Enter the number of elements in the array: "))

arr = []

for i in range(n):

    element = int(input("Enter element {}: ".format(i+1)))

    arr.append(element)

    max_num = arr[0]

for num in arr:

    if num > max_num:

        max_num = num

print("The largest number in the array is:", max_num)
```

58) Change the vowel characters to "S".

```
string = input("Enter a string: ")

vowels = "aeiouAEIOU"

new_string = ""

for char in string:

    if char in vowels:

        new_string += "S"

    else:

        new_string += char
```

```
print("New string:", new_string)
```


