CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Science and Engineering

CE259 - Programming in Python

Name : Rushik R. Rathod

ID : 20DCS103

Class : 4CSE2-B

1) Find whether a number is odd or even.

Source Code:

```
# odd or even

number = 8
if number%2==0:
    print(number)
    print("EVEN NUMBER")
else:
    print(number)
    print("ODD NUMBER")
```

Output:

```
PS D:\Python> python -u "d:\Python\oddOrEven.py"
8
EVEN NUMBER
PS D:\Python>
```

2) Find the largest among three numbers.

Source Code:

```
# the largest number and the smallest number

x = 10
y = 20
z = 30

if(x>y):
    if(x>z):
        print(str(x) + " is the largest number.")
    else:
        print(str(z) + " is the largest number.")
else:
    if(y>z):
        print(str(y) + " is the largest number.")
    else:
        print(str(z) + " is the largest number.")

if(x<y):
    if(x<z):
        print(str(x) + " is the smallest number.")
    else:
        print(str(z) + " is the smallest number.")
else:
    if(y<z):
        print(str(y) + " is the smallest number.")
    else:
        print(str(z) + " is the smallest number.")
```

Output:

```
PS D:\Python> python -u "d:\Python\largestSmallestNumber.py"
30 is the largest number.
10 is the smallest number.
PS D:\Python>
```

3) Find whether a number is positive or negative.

Source Code:

```
# negative number or positive number

x = 8
y = -11
z = 0

if(x>0):
    print(str(x) + " is a positive number.")
elif(x<0):
    print(str(x) + " is a negative number.")
else:
    print(str(x) + " is a zero/null.")

if(y>0):
    print(str(y) + " is a positive number.")
elif(y<0):
    print(str(y) + " is a negative number.")
else:
    print(str(y) + " is a zero/null.")

if(z>0):
    print(str(z) + " is a positive number.")
elif(z<0):
    print(str(z) + " is a negative number.")
else:
    print(str(z) + " is a zero/null.")
```

Output:

```
PS D:\Python> python -u "d:\Python\negativeOrPositive.py"
8 is a positive number.
-11 is a negative number.
0 is a zero/null.
PS D:\Python>
```

4) Find the result of a student.

Source Code:

```
id = "20DCS103"
name = "Rushik"
studentClass = "4CSE2 B"
maths = 98
computer = 96
physics = 99
chemistry = 95
english = 97
percentage = (maths + computer + physics + chemistry + english)/5
per = str(percentage)

print("ID         : " + id)
print("Name       : " + name)
print("Class      : " + studentClass)
print("Percentage : " + per)
print()

if percentage >= 90:
    print("Congratulations ! You are passed with AA grade !")
elif percentage >= 80:
    print("Congratulations ! You are passed with AB grade !")
elif percentage >= 70:
    print("Congratulations ! You are passed with BB grade !")
elif percentage >= 60:
    print("Congratulations ! You are passed with BC grade !")
elif percentage >= 50:
    print("Congratulations ! You are passed with CC grade !")
elif percentage >= 40:
    print("Congratulations ! You are passed with CD grade !")
else:
    print("Oops ! You failed the examination !")
print()
```

Output:

```
PS D:\Python> python -u "d:\Python\findPercentage.py"
ID         : 20DCS103
Name       : Rushik
Class      : 4CSE2 B
Percentage : 97.0

Congratulations ! You are passed with AA grade !

PS D:\Python>
```

5) Take the student's details as input and show the result.

Source Code:

```
print("*****Enter The Details*****")
id = input("ID : ")
name = input("Enter your name : ")
studentClass = input("Enter class : ")

s1 = float(input("Maths : "))
s2 = float(input("Computer : "))
s3 = float(input("Physics : "))
s4 = float(input("Chemistry : "))
s5 = float(input("English : "))

print()
print("ID         : " + id)
print("Name       : " + name)
print("Class      : " + studentClass)

percentage = (s1 + s2 + s3 + s4 + s5)/5
per = str(percentage)

print("Percentage : " + per)
print()

if percentage >= 90:
    print("Congratulations ! You are passed with AA grade !")
elif percentage >= 80:
    print("Congratulations ! You are passed with AB grade !")
elif percentage >= 70:
    print("Congratulations ! You are passed with BB grade !")
elif percentage >= 60:
    print("Congratulations ! You are passed with BC grade !")
elif percentage >= 50:
    print("Congratulations ! You are passed with CC grade !")
elif percentage >= 40:
    print("Congratulations ! You are passed with CD grade !")
else:
    print("Oops ! You failed the examination !")
print()
```

Output:

```
PS D:\Python> python -u "d:\Python\studentPercentage.py"
*****Enter The Details*****
ID : 20DCS103
Enter your name : Rushik
Enter class : CSE2B
Maths : 95
Computer : 98
Physics : 99
Chemistry : 94
English : 92

ID         : 20DCS103
Name       : Rushik
Class      : CSE2B
Percentage : 95.6

Congratulations ! You are passed with AA grade !

PS D:\Python>
```

PRACTICAL: 1.1

Create a program that asks the user to enter their name and their age. Printout a message addressed to them that tells them the year that they will turn 100 years old.

Source Code:

```
name = input("Enter your name : ")

age = int(input("Enter your age : "))

year = (2021 - int(age)) + 100

print("You will turn 100 years in " + str(year))
```

Output:

```
========== RESTART: D:\Python\1.1.py ==========
Enter your name : Rushik
Enter your age : 20
You will turn 100 years in 2101
```

PRACTICAL: 1.2

Write a program to ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user. Hint: how does an even / odd number react differently when divided by 2?

Source Code:

```
# odd or even

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

if number%2==0:

    print(str(number) + " is an even number.")

else:

    print(str(number) + " is an odd number.")
```

Output:

```
========== RESTART: D:\Python\1.2.py ==========
Enter a number : 103
103 is an odd number.
```

PRACTICAL: 2.1

Write a program as mentioned below: Take a list, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], and write a program that print out all the elements of the list that are less than 5.

Source Code:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
i = 0
for a[i] in a:
    if a[i]<5:
        print(a[i])
```

Output:

```
========================== RESTART: D:\Python\2.1.py ==========================
1
1
2
3
```

PRACTICAL: 2.2

Create a program that asks the user for a number and then prints out a list of all the divisors of that number.

Source Code:

```
number = int(input("Enter a number : " ))
print("Divisors...")
for i in range(1, number):
    if number%i == 0:
        print(i)
```

Output:

```
========================== RESTART: D:\Python\2.2.py ==========================
Enter a number : 18
Divisors...
1
2
3
6
9
```

PRACTICAL: 3.1

Take two lists, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.

Source Code:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
print("List a : " + str(a))
print("List b : " + str(b))

# finding the common elements
temp = []
for i in range(0, 11):
    for j in range(0, 13):
        if a[i] == b[j]:
            temp.append(b[j])
print("Before removing duplicates : " + str(temp))

# removing duplicates
result = []
for i in temp:
    if i not in result:
        result.append(i)
print("After removing duplicates : " + str(result))
```

Output:

```
========== RESTART: D:\Python\3.1.py ==========
List a : [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
List b : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
Before removing duplicates : [1, 1, 2, 3, 5, 8, 13]
After removing duplicates : [1, 2, 3, 5, 8, 13]
```

PRACTICAL: 3.2

Write a program by asking the user for a string and print out whether this string is a palindrome or not. (A palindrome is a string that reads the same forwards and backwards.)

Source Code:

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


def palindrome(myString):
    for i in range(0, int(len(myString)/2)):
        if myString[i] != myString[len(myString) - i - 1]:
            return False
        else:
            return True


result = palindrome(myString)


if result == 1:
    print("Entered string is palindrome.")
else:
    print("Entered string is NOT paindrome.")
```

Output:

```
========================== RESTART: D:\Python\3.2.py ==========================
Enter a string : MADAM
Entered string is palindrome.
```

PRACTICAL: 4.1

Write one line of Python that takes list a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100] and makes a new list that has only the even elements of this list in it.

Source Code:

```
a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

even_list = [a[i] for i in range(0, len(a)) if (a[i]%2) == 0]

print("Even elements of the list are : " + str(even_list))
```

Output:

```
============================ RESTART: D:\Python\4.1.py ============================
Even elements of the list are : [4, 16, 36, 64, 100]
```

PRACTICAL: 4.2

Write a program to make a two-player Rock-Paper-Scissors game.

(Hint: Ask for player plays (using input), compare them, print out a message of congratulations to the winner, and ask if the players want to start a new game)

Rules: Rock beats scissors, Scissors beats paper, Paper beats rock.

Source Code:

```
# rock, paper, scissors
player1 = input("Enter the name of player 1 : ")
player2 = input("Enter the name of player 2 : ")

print("ALL THE BEST! let's start the game...")

player1_input = input(player1 + ", enter your choice : ")
player2_input = input(player2 + ", enter your choice : ")

if player1_input == player2_input:
    print("Both players selected the same. It's a tie!")

elif player1_input == "rock":
    if player2_input == "scissors":
        print("Rock smashes scissors! " + player1 + " wins!")
    else:
        print("Paper covers rock! " + player2 + " wins!")

elif player1_input == "paper":
    if player2_input == "rock":
        print("Paper covers rock! " + player1 + " wins!")
    else:
        print("Scissors cut paper! " + player2 + " wins!")
```

```
elif player1_input == "scissors":
    if player2_input == "paper":
        print("Scissors cuts paper! " + player1 + " wins!")
    else:
        print("Rock smashes scissors! " + player2 + " wins!")
```

Output:

```
========== RESTART: D:\Python\4.2.py ==========
Enter the name of player 1 : Spiderman
Enter the name of player 2 : Batman
ALL THE BEST! let's start the game...
Spiderman, enter your choice : paper
Batman, enter your choice : rock
Paper covers rock! Spiderman winn!
```

PRACTICAL: 5.1

Write a program to generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right.

(Hint: remember to use the user input lessons from the very first practical)

Source Code:

```
import random


comp_number = random.randrange(1, 9)
flag = True


while(flag):
    user_number = int(input("Guess the number : "))


    if comp_number > user_number :
        print("Your guess is too low!")
        print("Guess again...")
        flag = True
    elif comp_number < user_number :
        print("Your guess is too high!")
        print("Guess again...")
        flag = True
    else:
        print("Congratulations! You guessed the right number !  : ")
        print("The random number was : " + str(comp_number))
        flag = False
```

Output:

```
========================== RESTART: D:\Python\5.1.py ==========================
Guess the number : 5
Your guess is too low!
Guess again...
Guess the number : 8
Your guess is too high!
Guess again...
Guess the number : 6
Your guess is too low!
Guess again...
Guess the number : 7
Congratulations! You guessed the right number !  :
The random number was : 7
```

PRACTICAL: 5.2

Take two lists, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes. Write this in one line of Python using at least one list comprehension.

Source Code:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]


print("List a : " + str(a))
print("List b : " + str(b))


# finding the common elements using list comprehension
common = [i for i in a if i in b]
print("Before removing duplicates : " + str(common))


# removing duplicates using list comprehension
result = []
[result.append(i) for i in common if i not in result]
print("After removing duplicates  : " + str(result))
```

Output:

```
========== RESTART: D:\Python\5.2.py ==========
List a : [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
List b : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
Before removing duplicates : [1, 1, 2, 3, 5, 8, 13]
After removing duplicates  : [1, 2, 3, 5, 8, 13]
```

PRACTICAL: 6.1

Write a program using a function to check whether the number is prime or not. (A prime number is a number that has no divisors.)

Source Code (without using a function):

```
# prime number without using a function
n = int(input("Enter a number : "))
count = 0

for i in range(1, n):
    if(n % i) == 0:
        count = count + 1

if count == 1:
    print(str(n) + " is a prime number.")
else:
    print(str(n) + " is not a prime number.")
```

Source Code (using a function):

```
# function definition to check prime number
def prime(n):
    count = 0
    for i in range(1, n):
        if(n % i) == 0:
            count = int(count) + 1

    if count == 1:
        print(str(n) + " is a prime number.")
    else:
        print(str(n) + " is not a prime number.")
```

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


# function calling
prime(num)
```

Output:

```
============================ RESTART: D:\Python\6.1.py ============================
Enter a number : 37
37 is a prime number.
```

PRACTICAL: 6.2

Write a program that takes a list of numbers (for example, a = [5, 10, 15, 20, 25]) and makes a new list of only the first and last elements of the given list. For practice, write this code inside a function.

Source Code (without using a function):

```
a = [5, 10, 15, 20, 25]
print("The list is : " + str(a))
print("First element : " + str(a[0]))
print("Last element  : " + str(a[-1]))
```

Source Code (using a function):

```
# function definition
def firstAndLast():
    print("First element : " + str(a[0]))
    print("Last element  : " + str(a[-1]))

# driver code
a = [5, 10, 15, 20, 25]
print(a)

# function calling
firstAndLast()
```

Output:

```
============================ RESTART: D:\Python\6.2.py ============================
The list is : [5, 10, 15, 20, 25]
First element : 5
Last element  : 25
```

PRACTICAL: 7.1

Write a program that asks the user how many Fibonacci numbers to generate and then generates them. Take this opportunity to think about how you can use functions. Make sure to ask the user to enter the number of numbers in the sequence to generate. (Hint: The Fibonacci sequence is a sequence of numbers where the next number in the sequence is the sum of the previous two numbers in the sequence. The sequence looks like this: 0, 1, 1, 2, 3, 5, 8, 13,…)

Source Code:

```
# function definition
def fib(num):
    t1 = 0
    t2 = 1
    nextTerm = 0

    for i in range(1, n+1):
        if i == 1:
            print(t1)
            continue
        elif i == 2:
            print(t2)
            continue
        else:
            nextTerm = t1 + t2
            t1 = t2
            t2 = nextTerm
            print(nextTerm)

# driver code
n = int(input("Enter the number of fibonacci sequence number : "))
# function calling
fib(n)
```

Output:

```
========================== RESTART: D:\Python\7.1.py ==========================
Enter the number of fibonacci sequence number : 8
0
1
1
2
3
5
8
13
```

PRACTICAL: 7.2

Write a program (function!) that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates.

Source Code:

```
# defining a function
def fun(n):

    print("Enter list elements : ")
    old_list = []
    for i in range(0, n):
        ele = input()
        old_list.append(ele)
    print("Enterd list : " + str(old_list))

    # using list comprehension
    new_list = []
    [new_list.append(i) for i in old_list if i not in new_list]
    print("New list after removing duplicates : " + str(new_list))



# driver code
num = int(input("Enter the length of a list : "))
fun(num)       #fuction calling
```

Output:

```
============================ RESTART: D:\Python\7.2.py ============================
Enter the length of a list : 5
Enter list elements :
1
1
8
9
5
Enterd list : ['1', '1', '8', '9', '5']
New list after removing duplicates : ['1', '8', '9', '5']
```

PRACTICAL: 8.1

Write a password generator in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your run-time code in a main method.

Source Code:

```
# defining a function
def rev(sen):
    words = sen.split(" ")
    words = list(reversed(words))
    print(" ".join(words))



#driver code
sen = str(input("Enter a string : "))
rev(sen)    # calling a function
```

Output:

```
========================== RESTART: D:\Python\8.1.py ==========================
Enter a string : My name is Rushik
Rushik is name My
```

PRACTICAL: 8.2

Write a password generator in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your run-time code in a main method.

Source Code:

```
import random
import array

password_length = 12

digits = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']

lowercase = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k',
'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y',
'z']

uppercase = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
'M', 'N', 'O', 'p', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y',
'Z']

symbols = ['@', '#', '$', '%', '=', ':', '?', '.', '/', '|', '~',
'>', '*', '(', ')', '<']

random_digit = random.choice(digits)
random_upper = random.choice(uppercase)
random_lower = random.choice(lowercase)
random_symbol = random.choice(symbols)

temp_pass = random_digit + random_upper + random_lower +
random_symbol

# combines all the character arrays above to form one array
combined_list = digits + lowercase + uppercase + symbols

for x in range(password_length - 4):
    temp_pass = temp_pass + random.choice(combined_list)
    temp_pass_list = array.array('u', temp_pass)
    random.shuffle(temp_pass_list)
```

```
# traverse the temporary password array and append the chars to form
the password
password = ""
for x in temp_pass_list:
        password = password + x
# print out password
print("Generated password : " + str(password))
```

Output:

```
========== RESTART: D:\Python\8.2.py ==========
Generated password : IN|p?Opzk913
>>>
```

PRACTICAL: 9.1

Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.

Source Code:

```
from ast import arg

class Circle:
    def __init__(self, r):
        self.r = r

    def area(self):
        ar = 3.14*self.r*self.r
        return ("{0:.2f}".format(ar))

    def perimeter(self):
        peri = 2*3.14*self.r
        return ("{0:.2f}".format(peri))

# driver code
r = 25
print("Radius    : " + str(r))
obj = Circle(r)
print("Area      : " + str(obj.area()))
print("Perimeter : " + str(obj.perimeter()))
```

Output:

```
========== RESTART: D:\Python\9.1.py ==========
Radius    : 25
Area      : 1962.50
Perimeter : 157.00
>>>
```

PRACTICAL: 9.2

Write a function that takes an ordered list of numbers (a list where the elements are in order from smallest to largest) and another number. The function decides whether or not the given number is inside the list and returns (then prints) an appropriate Boolean.

Source Code:

```
# function definition
def process(list1, n):
    if n in list1:
        print("True")
    else:
        print("False")

# driver code
list1 = []
list1 = input("Enter an ordered list : ")

print("List entered : " + str(list1))
n = input("Enter a number to check : ")
print("Entered number : " + str(n))

# calling function
process(list1, n)
```

Output:

```
========== RESTART: D:\Python\9.2.py ==========
Enter an ordered list : 1 3 5 7 9
List entered : 1 3 5 7 9
Enter a number to check : 8
Entered number : 8
False
>>>
```

PRACTICAL: 10.1

Given a .txt file that has a list of a bunch of names, count how many of each name there are in the file, and print out the results to the screen.

Source Code:

```
file = open("20dcs103.txt", "rt")
words = file.read().lower()

print(f"Press 1 to find a word occurence in {file.name}")
print(f"Press 2 to find a letter occurence in {file.name}")

choice = int(input("Enter your choice : "))

if choice == 1:
    user_word = input("Enter your word: ")
    print(f"Your word '{user_word}' occurs {words.count(user_word)}
times in {file.name}")

elif choice == 2:
    count = 0
    user_letter = input("Enter your letter : ")
    for i in words:
        if i == user_letter:
            count = count + 1

    print(
        f"Your letter '{user_letter}' occurs '{count}' times in
'{file.name}'")

else:
    print("Invalid choice !")
```

Output:

```
========================== RESTART: D:\Python\10.1.py ==========================
Press 1 to find a word occurence in 20dcs103.txt
Press 2 to find a letter occurence in 20dcs103.txt
Enter your choice : 2
Enter your letter : r
Your letter 'r' occurs '2' times in '20dcs103.txt'
```

PRACTICAL: 10.2

Write a program to implement different Data structures using Python.

❖ **Linked List**

Source code:

```
class Node:
    def __init__(self, new_data):
        self.data = new_data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None

    def insert_at_first(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node

    def insert_at_last(self, new_data):
        new_node = Node(new_data)

        if self.head is None:
            self.head = new_node
            return
        else:
            t = self.head
            while t.next:
                t = t.next
            t.next = new_node

    def deleteNode(self, position):
        if self.head is None:
            return
        temp = self.head

        if position == 0:
            self.head = temp.next
            temp = None
            return
```

```
        for i in range(position - 1):
            temp = temp.next
            if temp is None:
                break

        if temp is None:
            return
        if temp.next is None:
            return
        next = temp.next.next
        temp.next = None
        temp.next = next

    def search_key(self, key):

        t = self.head
        while t is not None:
            if t.data == key:
                return True
            t = t.next

        return False

    def display(self):
        p = self.head
        while p is not None:
            print(p.data, end=" ")
            p = p.next


# driver code
if __name__ == '__main__':

    l1 = LinkedList()
    l1.insert_at_last(1)
    l1.insert_at_last(2)
    l1.insert_at_last(3)
    l1.insert_at_last(4)
    l1.insert_at_last(5)

    print("--- Original List ---")
    l1.display()
```

```
    # inserting at first
    print()
    n1 = int(input("Enter the element for insertion at beginning :
"))
    l1.insert_at_first(n1)
    l1.display()

    # inserting at last
    print()
    n2 = int(input("Enter the element for insertion at last : "))
    l1.insert_at_last(n2)
    l1.display()

    # searching for a value
    print()
    key = int(
        input("Enter a number to check whether it is present in the
list or not : "))

    if l1.search_key(key):
        print("Key found!")
    else:
        print("Key is not found.")

    # deleting a value
    pos = int(input("Enter a position to delete a node : "))
    l1.deleteNode(pos)
    l1.display()
```

Output:

```
========== RESTART: D:\Python\10.2 LinkedList.py ==========
--- Original List ---
1 2 3 4 5
Enter the element for insertion at beginning : 10
10 1 2 3 4 5
Enter the element for insertion at last : 20
10 1 2 3 4 5 20
Enter a number to check whether it is present in the list or not : 5
Key found!
Enter a position to delete a node : 6
10 1 2 3 4 5
```

❖ **Stack**

Source code:

```
class Stack:

    def __init__(self, size):
        self.arr = [None] * size
        self.capacity = size
        self.top = -1

    def push(self, val):
        if self.isFull():
            print("Stack overflow!")
            return
        else:
            print(f'Inserting : {val}')
            self.top = self.top + 1
            self.arr[self.top] = val

    def pop(self):
        if self.isEmpty():
            print("Stack underflow!")
            return
        else:
            print(f'Removing : {self.peek()}')
            top = self.arr[self.top]
            self.top = self.top - 1
            return top

    def peek(self):
        if self.isEmpty():
            return
        return self.arr[self.top]

    def size(self):
        return self.top + 1

    def isEmpty(self):
        return self.size() == 0
    def isFull(self):
        return self.size() == self.capacity
```

```
if __name__ == '__main__':

    st = Stack(5)
    st.push(1)
    st.push(3)
    st.push(5)
    st.push(7)
    st.push(9)
    print('Top element is : ', st.peek())
    print('The stack size is : ', st.size())
    st.pop()
    st.pop()
    st.pop()
    print('Top element is : ', st.peek())
    print('The stack size is : ', st.size())
```

Output:

```
======================== RESTART: D:\Python\10.2 Stack.py ========================
Inserting : 1
Inserting : 3
Inserting : 5
Inserting : 7
Inserting : 9
Top element is :  9
The stack size is :  5
Removing : 9
Removing : 7
Removing : 5
Top element is :  3
The stack size is :  2
```

❖ Queue

Source code:

```
class Queue:

    def __init__(self):
        self.queue = []

    def enQueue(self, item):
        self.queue.append(item)

    def deQueue(self):
        if len(self.queue) < 1:
            return None
        return self.queue.pop(0)

    def display(self):
        print("Queue : " + str(self.queue))

    def size(self):
        return len(self.queue)


# driver code
q = Queue()

q.enQueue(1)
q.enQueue(3)
q.enQueue(5)
q.enQueue(7)
q.enQueue(9)

q.display()

q.deQueue()
print("--- After deQueue() ---")
q.display()
```

Output:

```
======================== RESTART: D:\Python\10.2 Queue.py ========================
Queue : [1, 3, 5, 7, 9]
--- After deQueue() ---
Queue : [3, 5, 7, 9]
```

❖ **Binary Tree**

Source code:

```
class Node:
    def __init__(self, key):
        self.left = None
        self.right = None
        self.val = key

    def traverse_PreOrder(self):
        print(self.val, end=' ')
        if self.left:
            self.left.traverse_PreOrder()
        if self.right:
            self.right.traverse_PreOrder()

    def traverse_InOrder(self):
        if self.left:
            self.left.traverse_InOrder()
        print(self.val, end=' ')
        if self.right:
            self.right.traverse_InOrder()

    def traverse_PostOrder(self):
        if self.left:
            self.left.traverse_PostOrder()
        if self.right:
            self.right.traverse_PostOrder()
        print(self.val, end=' ')



root = Node(1)
root.left = Node(3)
root.right = Node(5)

root.left.left = Node(7)

print("Pre-order traversal  : ", end="")
root.traverse_PreOrder()
print()
```

```
print("In-order traversal   : ", end="")
root.traverse_InOrder()

print()
print("Post-order traversal : ", end="")
root.traverse_PostOrder()
```

Output:

```
==================== RESTART: D:\Python\10.2 Binary Tree.py ====================
Pre-order traversal  : 1 3 7 5
In-order traversal   : 7 3 1 5
Post-order traversal : 7 3 5 1
```

PRACTICAL: 11.1

Develop programs to understand the working of exception handling with the user guessing a number until he/she gets it right.

Source Code:

```
import random

# creating error classes
class ValueTooSmallError(Exception):
    pass

class ValueTooLargeError(Exception):
    pass

# driver code
number = random.randrange(1,11)
while True:
    try:
        i = int(input("Guess a number : "))

        if (i < int(number)):
                raise ValueTooSmallError
        elif (i > int(number)) :
                raise ValueTooLargeError
        break

    except ValueTooSmallError:
            print("Entered value is too small!")
            print()
    except ValueTooLargeError:
            print("Entered value is too large!")
            print()

print("You guessed the right number!")
```

Output:

```
========================== RESTART: D:\Python\11.1.py ==========================
Guess a number : 10
Entered value is too large!

Guess a number : 9
Entered value is too large!

Guess a number : 8
Entered value is too large!

Guess a number : 7
Entered value is too large!

Guess a number : 6
Entered value is too large!

Guess a number : 5
Entered value is too large!

Guess a number : 4
Entered value is too large!

Guess a number : 3
Entered value is too large!

Guess a number : 2
You guessed the right number!
```

PRACTICAL: 11.2

Develop programs to understand the working of exception handling with the user entering age and check whether he/she is eligible to vote or not.

Source Code:

```
import random

# creating error classes
class AgeTooSmall(Exception):
    pass

# driver code
try:
    i = int(input("Enter your age : "))
    if (i <= 18):
        raise AgeTooSmall

except AgeTooSmall:
    print("Sorry! You are not eligible to vote for the country.")
    print()

print("Yes! You are eligible to vote!")
```

Output:

```
========================== RESTART: D:\Python\11.2.py ==========================
Enter your age : 20
Yes! You are eligible to vote!
```

PRACTICAL: 11.3

Develop programs to understand the working of exception handling with the user entering balance and the amount to withdraw. Additionally, check whether he/she is eligible to withdraw the desirable amount or not.

Source Code:

```
# creating error classes

class InvalidEntry(Exception):
    pass

class BalanceIsNotEnough(Exception):
    pass

# driver code
try:
    b = int(input("Enter your balance : "))
    w = int(input("Enter the amount you want to withdraw : "))

    if(b < 0):
        raise InvalidEntry
    elif (w > b):
        raise BalanceIsNotEnough

except InvalidEntry:
    print("Enter a valid number bro !")

except BalanceIsNotEnough:
    print("Oops! Your balance is not enough")

print("Amount withdrawn!")
```

Output:

```
========================== RESTART: D:\Python\11.3.py ==========================
Enter your balance : 200000
Enter the amount you want to withdraw : 10000
Amount withdrawn!
```

PRACTICAL: 12.1

Create two 2D Numpy arrays with random numbers and concatenate them using the Numpy library. After Concatenation, reshape the resulting Numpy array such that the number of rows and columns is reversed.

Source Code:

```
import numpy as np

arr1 = np.array([[1, 3, 5], [7, 9, 11]])
arr2 = np.array([[2, 4, 6], [8, 10, 12]])

print("--- Array 1 ---")
print(arr1)
print()
print("--- Array 2 ---")
print(arr2)
print()
print('--- After concatenating two arrays ---')
concatArr = np.array(np.concatenate((arr1, arr2), axis=0))
print(concatArr)

print("\nIntitial Shape", concatArr.shape)

reshapeArr = np.array(np.reshape(concatArr, (3, 4)))
print('Reshaping array by interchanging rows and columns, new shape
: ', reshapeArr.shape)
print('--- After reshaping the array ---\n', reshapeArr)
```

Output:

```
========================== RESTART: D:\Python\12.1.py ==========================
--- Array 1 ---
[[ 1  3  5]
 [ 7  9 11]]

--- Array 2 ---
[[ 2  4  6]
 [ 8 10 12]]

--- After concatenating two arrays ---
[[ 1  3  5]
 [ 7  9 11]
 [ 2  4  6]
 [ 8 10 12]]

Intitial Shape (4, 3)
Reshaping array by interchanging rows and columns, new shape :  (3, 4)
--- After reshaping the array ---
 [[ 1  3  5  7]
 [ 9 11  2  4]
 [ 6  8 10 12]]
```

PRACTICAL: 12.2

Create a Pandas series from a Python List. Find out the mean, median, mode, range and standard deviation of the series.

Source Code:

```
import pandas as pd

arr = [[1, 3, 5, 7], [2, 4, 6, 8], [9, 11, 13, 15],
       [10, 12, 14, 16], [17, 19, 21, 23], [18, 20, 22, 24]]

df = pd.DataFrame(arr)

print("Details about DataFrame\n", df)
print("\nMean of data\n", df.mean())

print("\nMedian of data\n", df.median())

print("\nMode of data\n", df.mode())

print("\nStandard Deviation of data\n", df.std())

print("\nRange\n", df.max()-df.min())
```

Output:

```
========================== RESTART: D:\Python\12.2.py ==========================
Details about DataFrame
     0   1   2   3
0    1   3   5   7
1    2   4   6   8
2    9  11  13  15
3   10  12  14  16
4   17  19  21  23
5   18  20  22  24

Mean of data
 0     9.5
1     11.5
2     13.5
3     15.5
dtype: float64

Median of data
 0     9.5
1     11.5
2     13.5
3     15.5
dtype: float64

Mode of data
     0   1   2   3
0    1   3   5   7
1    2   4   6   8
2    9  11  13  15
3   10  12  14  16
4   17  19  21  23
5   18  20  22  24

Standard Deviation of data
 0    7.17635
1     7.17635
2     7.17635
3     7.17635
dtype: float64


Range
 0    17
1     17
2     17
3     17
dtype: int64
```

Thank you !