AISSMS

COLLEGE OF ENGINEERING

Approved by AICTE, New Delhi, Recognized by Govt. of Maharashtra, Affiliated to Savitribai Phule Pune University and recognized 2(f) and 12(B) by UGC (Id.No. PU / PN/ Engg. / 093 (1992)

Accredited by NAAC with 'A+' Grade

DEPARTMENT OF COMPUTER ENGINEERING

YEAR 2020-21

TERM - 1

JOURNAL

DATA STRUCTURE LAB(FDS)

CLASS – SE COMPUTER ENGG.(SHIFT A)

NAME – PRATIK PINGALE

ROLL NO - 19CO056

BATCH – C

FACULTY – MR. SUMEDH DHENGRE.

★ INDEX★

SR No	Exp No.	Title of Experiment	Date of Prac	Date of Sub	Page No.
1	**A2**	Write a Python program to store marks scored in subject “Fundamental of Data Structure” by N students in the class. Write functions to compute following: a) The average score of class b) Highest score and lowest score of class c) Count of students who were absent for the test d) Display mark with highest frequency	15 July	22 July	4-12
2	**A5**	Write a Python program to compute following operations on String: a) To display word with the longest length b) To determines the frequency of occurrence of particular character in the string c) To check whether given string is palindrome or not d) To display index of first appearance of the substring e) To count the occurrences of each word in a given string	22 July	5 Aug	13-23
3	**A9**	Write a python program to compute following computation on matrix: a) Addition of two matrices b) Subtraction of two matrices c) Multiplication of two matrices d) Transpose of a matrix	5 Aug	19 Aug	24-32
4	**B11**	a) Write a python program to store roll numbers of student in array who attended training program in random order. Write function for searching whether particular student attended training program or not, using Linear search and Sentinel search. b) Write a python program to store roll numbers of student array who attended training program in sorted order. Write function for searching whether particular student attended training program or not, using Binary search and Fibonacci search.	19 Aug	26 Aug	33-41
5	**B14**	Write a python program to store first year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using a) Selection Sort b) Bubble sort and display top five scores.	26 Aug	9 Sept	42-48
6	**B16**	Write a python program to store first year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using quick sort and display top five scores.	9 Sept	16 Sept	49-54
7	**C19**	Department of Computer Engineering has student's club named 'Pinnacle Club'. Students of second, third and final year of department can be granted membership on request. Similarly one may cancel the membership of club. First node is reserved for	16 Sept	30 Sept	55-67

		president of club and last node is reserved for secretary of club. Write C++ program to maintain club member‘s information using singly linked list. Store student PRN and Name. Write functions to: a) Add and delete the members as well as president or even secretary. b) Compute total number of members of club c) Display members d) Two linked lists exists for two divisions. Concatenate two lists.			
8	**C21**	Write C++ program for storing appointment schedule for day. Appointments are booked randomly using linked list. Set start and end time and min and max duration for visit slot. Write functions for- a) Display free slots b) Book appointment c) Cancel appointment (check validity, time bounds, availability) d) Sort list based on time e) Sort list based on time using pointer manipulation	30 Sept	7 Oct	68-78
9	**D26**	In any language program mostly syntax error occurs due to unbalancing delimiter such as (),{},[]. Write C++ program using stack to check whether given expression is well parenthesized or not.	7 Oct	14 Oct	79-85
10	**D27**	Implement C++ program for expression conversion as infix to postfix and its evaluation using stack based on given conditions: 1. Operands and operator, both must be single character. 2. Input Postfix expression must be in a desired format. 3. Only '+', '-', '*' and '/ ' operators are expected.	14 Oct	28 Oct	86-94
11	**E29**	Queues are frequently used in computer programming, and a typical example is the creation of a job queue by an operating system. If the operating system does not use priorities, then the jobs are processed in the order they enter the system. Write C++ program for simulating job queue. Write functions to add job and delete job from queue.	28 Oct	4 Nov	95-104
12	**E31**	A double-ended queue (deque) is a linear list in which additions and deletions may be made at either end. Obtain a data representation mapping a deque into a one-dimensional array. Write C++ program to simulate deque with functions to add and delete elements from either end of the deque.	4 Nov	25 Nov	105-117
13	**E32**	Pizza parlor accepting maximum M orders. Orders are served in first come first served basis. Order once placed cannot be cancelled. Write C++ program to simulate the system using circular queue using array.	25 Nov	2 Dec	118-127

Name: Pratik Pingale
No: 19CO056

EXPERIMENT - A2.

AIM: Write a python program to store marks scored in subject FDS by N students. Write functions for:

a) Average class score
b) Highest and lowest scores
c) Total absent students
d) Display marks with highest frequency.

THEORY: Python:

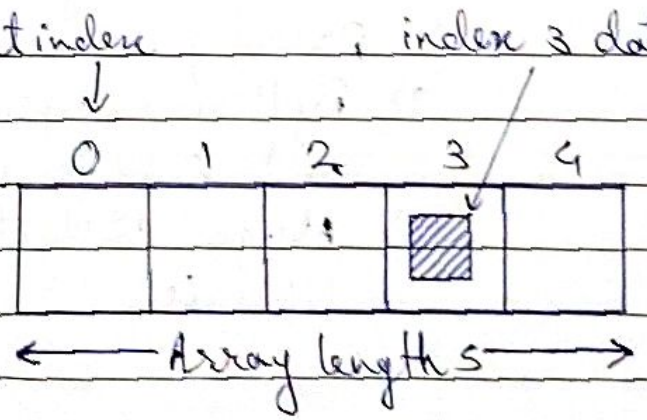
Python is a high purpose general level and very popular programming language. Python programming language is used in web development, Machine Learning along with all cutting edge technology in software technology. Python programming language is very well suited for Beginners as well as experienced programmers.

Features of python:

1) Widely used for high level multi-purpose tasks.
2) Follow Object oriented and Procedural paradigms
3) Python programs are generally redable and easy to write.
4) Huge collections of standard libraries.

→ Python array

1) An array is a collection of data stored at contiguous memory location.
2) It stores data of same kind together
3) This makes easier to access element by directly adding offset of base value.
4) Array name is the location of 1st element of array.
5) Array in python can be handled by either key word 'list' or a module 'array'
6) Array are useful and efficent when we had to manipulate a similar data type.
7) Auser can treat list as an array.
8) However a user can't restrain data types in list.
9) In array module all elements of array are of same type.



Type Code	C-Type	Python type	Byte
b	signed char	int	1
B	unsigned char	int	1
u	Py_UNICODE	unicode char	2
h	signed short	int	2
H	unsigned short	int	2
i	signed int	int	2
I	unsigned int	int	2
l	signed long	int	4
L	unsigned long	int	4
q	signed long long	int	8
Q	unsigned long long	int	8
f	float	float	4
d	double	float	8

Basic Operations :

1) Traverse :- Accessing one element one by one

2) Insertion :- Adding array to element at start, end or in between.

3) Delete :- Delete an element from start, end or its first occouring.

4) Search :- Find an element in an array with matches given value

5) Update :- Update the array with new value

6) Length :- Using builtin len() function to find array length.

• Example:

```
import array
arr = array.array('i', [1, 2, 3])
print('Given array is: ', end= '')
for i in range (3):
    print (a[i], end= '')
print().
```

Output: Given array is : 1 2 3

CONCLUSION: From this experiment we understood array implementation in python. Perform various array operation like calculating average of list, maximum, minimum or highest frequency in list.

```python
# DSL - ASSIGNMENT 1 - A2

class Test:
    def __init__(self):
        self.fds = []
        self.n = 0


    def getdata(self):
        self.n = int(input('\nEnter the class strength of SE Comp A: '))
        print('\n(Note: Enter -1 for absent students)\n')
        print('————————————————————————————————————————————————')
        print('Enter the marks scored in Fundamental of Data Structure: ')
        print('————————————————————————————————————————————————\n\n')

        for i in range(self.n):
            self.fds.append(int(input(f'Enter marks of roll no {i + 1}: ')))


    def putdata(self):
        print('\n\nTest Marks of Fundamental of Data Structure are as follows ... \n')
        print('********************************')
        print('| Roll No | DSA Marks |')
        print('********************************')

        for i in range(self.n):
            print(f'|\t{i + 1}\t|\t{self.fds[i]}\t|')
        print('————————————————————————————————\n\n')


    def avg(self):
        total = 0
        present = 0
        for i in self.fds:
            if i != -1:
                total += i
                present += 1

        print(f'Average marks of the class: {round(total / present, 3)}\n\n')


    def absstud(self):
        print('Students absent for Fundamental of Data Structure test are: \n')
        absent = 0
        for i in range(self.n):
            if self.fds[i] == -1:
                print(f'Roll No: {i + 1} absent')
                absent += 1

        print(f'\nTotal absent students are: {absent}\n\n')
```

```python
def maxmin(self):
    maxi, max_rollno, mini, min_rollno = 0, [], 10, []
    for i, val in enumerate(self.fds):
        if val ≠ -1:
            if maxi == val:
                max_rollno.append(i + 1)
            elif maxi < val:
                max_rollno = [i + 1]
                maxi = val

            if mini == val:
                min_rollno.append(i + 1)
            elif mini ≥ val:
                min_rollno = [i + 1]
                mini = val

    max_rollno, min_rollno = ', '.join(map(str, max_rollno)), ', '.join(map(str, min_rollno))

    print(f'Highest Test Score : Roll No : ({max_rollno}) with Marks = {maxi}\n')
    print(f'Lowest Test Score : Roll No : ({min_rollno}) with Marks = {mini}\n\n')


def frequency(self):
    max_marks = 10
    freq = [0] * (max_marks + 1)
    stud, marks, index = 0, 0, []

    for val in self.fds:
        if val ≠ -1:
            freq[val] += 1

    for i, val in enumerate(freq):
        if stud ≤ val:
            marks = i
            stud = val

    for i, val in enumerate(self.fds):
        if val == marks:
            index.append(i + 1)

    index = ', '.join(map(str, index))
    print(f'Maximum of {marks} marks are scored by {stud} students with Roll No: ({index})')
```

```python
def main():
    test = Test()
    test.getdata()
    test.putdata()
    test.avg()
    test.absstud()
    test.maxmin()
    test.frequency()

if __name__ == "__main__":
    main()
```

```
"""

———————————— OUTPUT ————————————

Enter the class strength of SE Comp A: 10

(Note: Enter -1 for absent students)

————————————————————————————————————————————————
Enter the marks scored in Fundamental of Data Structure:
————————————————————————————————————————————————


Enter marks of roll no 1: 2
Enter marks of roll no 2: 5
Enter marks of roll no 3: -1
Enter marks of roll no 4: 4
Enter marks of roll no 5: 2
Enter marks of roll no 6: 10
Enter marks of roll no 7: 6
Enter marks of roll no 8: 4
Enter marks of roll no 9: 10
Enter marks of roll no 10: 4


Test Marks of Fundamental of Data Structure are as follows...

*********************************
|    Roll No    |   DSA Marks   |
*********************************
|       1       |       2       |
|       2       |       5       |
|       3       |       -1      |
|       4       |       4       |
|       5       |       2       |
|       6       |       10      |
|       7       |       6       |
|       8       |       4       |
|       9       |       10      |
|       10      |       4       |
————————————————————————————————


Average marks of the class: 5.222


Students absent for Fundamental of Data Structure test are:

Roll No: 3 absent
```

```
Total absent students are: 1


Highest Test Score : Roll No : (6, 9) with Marks = 10

Lowest Test Score : Roll No : (1, 5) with Marks = 2


Maximum of 4 marks are scored by 3 students with Roll No: (4, 8, 10)

"""
```

Pralik Pingale 19C0056

Page No.
Date

EXPERIMENT - A5.

AIM: Write a python program to perform following operation on string.

a) Display word with longest length.
b) Charater frequency in a string.
c) Check palindrome.
d) Display accurance index of substring
e) Count accurange of word in string.

THEORY: Python:

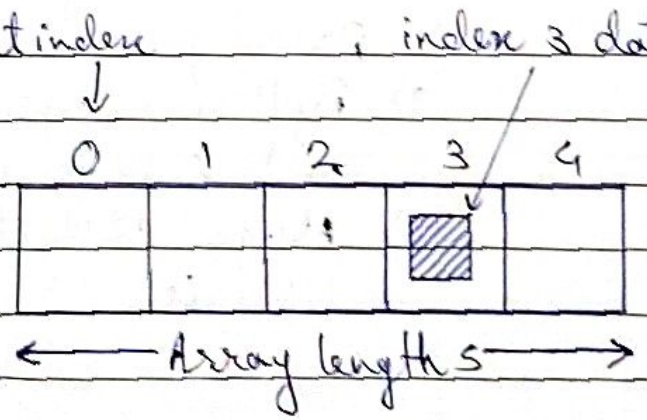
Python is a high purpose general level and very popular programming language. Python programming language is used in web development, Machine Learning along with all cutting edge technolagy in software technolo Python programming language is very well suited for Beginners as well as experienced programmers.

Features of python:

1) Widely used for high level multi-purpose tasks.
2) Follow Object oriented and Procedura paradigms
3) Python programs are generally redabl and easy to write.
4) Huge collections of standard libraries.

→ Python array

1) An array is a collection of data stored at contiguous memory location.
2) It stores data of same kind together
3) This makes easier to access element by directly adding offset of base value.
4) Array name is the location of 1st element of array.
5) Array in python can be handled by either key word 'list' or a module 'array'
6) Array are useful and efficent when we had to manipulate a similar data type.
7) Auser can treat list as an array.
8) However a user can't restrain data types in list.
9) In array module all elements of array are of same type.



Type Code	C-Type	Python type	Byte
b	signed char	int	1
B	unsigned char	int	1
u	Py_UNICODE	unicode char	2
h	signed short	int	2
H	unsigned short	int	2
i	signed int	int	2
I	unsigned int	int	2
l	signed long	int	4
L	unsigned long	int	4
q	signed long long	int	8
Q	unsigned long long	int	8
f	float	float	4
d	double	float	8

Basic Operations :

1) Traverse :- Accessing one element one by one

2) Insertion :- Adding array to element at start, end or in between.

3) Delete :- Delete an element from start, end or its first occouring.

4) Search :- Find an element in an array with matches given value

5) Update :- Update the array with new value

6) Length :- Using builtin len() function to find array length.

• Example:

```
import array
arr = array.array('i', [1, 2, 3])
print('Given array is: ', end= '')
for i in range (3):
    print (a[i], end= '').
print().
```

Output: Given array is : 1 2 3

• Strings in Python:

1) String literals in python are surrounded by either single quotation or double quotation mark.

'hella' == "hello"

2) Assign strings to variable

```
a = "hello"
print(a)
```

→ hella

Page No.
Date

5) Multiline string

```
a = """ This is multiline
        string . 2 Line"""
print (a).
```

→ This is multiline
string . 2 Line

4) Strings are same as array.

```
s = 'hello'
print ( s[1]).
```

→ e

5). Slicing

```
a = 'Hello, world'
print (a[2: 8]
```

→ llo, w

6) Length of string

```
a = 'Hello'
print (len(a))
```

→ 5

7) String methods :

```
s = 'Hello, world '
print (a.strip())
print (a. lower ())
print (a. upper ())
print (a. split(', '))
```

→ Hello, world # strips white steps
hello, world
HELLO, WORLD

['Hello', 'World'].

8) String Concatenation.

```
print('Hello' + 'World')
```

→ Hello World

9) Checking string

```
s = 'hello'
print('e' in s).
```

→ True

CONCLUSION: From this experiment we understood array implementation in python as well as string operation such as search or string advance. Also we learnt to calculate string length.

```python
# DSL - ASSIGNMENT 2 - A5

from pprint import pprint

def main():
    string = input('Enter string : ')
    stringfreq = dict()

    for i in string:
        stringfreq[i] = stringfreq.get(i, 0) + 1
    stringfreqword = dict()

    for i in string.split():
        stringfreqword[i] = stringfreqword.get(i, 0) + 1

    print(f'\nWord with maximum length : {max(string.split(), key=len)}')
    print(f'\nFrequency of each character :')
    pprint(stringfreq)

    pali = input('\nEnter a string to check Palindrome: ')
    print(f"Given string is {'not ' if pali != pali[::-1] else ''}a palindrome.\n")

    print(f"First occurrence of given substring is at index : {string.index(input('Enter
substring to be searched : '))}")

    print(f'\nFrequency of each word :')
    pprint(stringfreqword)


if __name__ == "__main__":
    main()
```

"""

──────────────── OUTPUT ────────────────

```
Enter string : Dali is a hunting goddess from the mythology of the Georgian people. The patron of
hoofed wild mountain animals, she was said to reward hunters who obeyed her taboos and to punish
violators. She was usually described as a beautiful nude woman with golden hair and glowing skin,
although she sometimes took on the form of her favored animals. Stories depict her taking human
lovers and jealously killing them, and later clashing with her rival Saint George. After the rise
of Christianity in Georgia, the stories told about Dali changed; Saint George was presented as
having the power to overrule her, and she began to be conflated with a malicious nature spirit
called the ali. As a patron of the hunt, she has been compared with Artemis of Greek mythology, a
hag in Scottish mythology called the glaistig, and a maiden from folklore who tames a unicorn.
Her associations with gold, seduction, and the morning star have led scholars to draw connections
with goddesses such as Aphrodite and Ishtar.

Word with maximum length : Christianity

Frequency of each character :
{' ': 168,
',': 10,
'.': 7,
';': 1,
'A': 4,
'C': 1,
'D': 2,
'G': 5,
'H': 1,
'I': 1,
'S': 5,
'T': 1,
'a': 74,
'b': 8,
'c': 17,
'd': 36,
'e': 82,
'f': 15,
'g': 25,
'h': 54,
'i': 59,
'j': 1,
'k': 6,
'l': 39,
'm': 20,
'n': 52,
'o': 72,
'p': 11,
'r': 47,
's': 55,
't': 64,
'u': 20,
```

```
'v': 7,
'w': 17,
'y': 10}

Enter a string to check Palindrome: malayalam
Given string is a palindrome.

Enter substring to be searched : hunt
First occurrence of given substring is at index : 10

Frequency of each word :
{'After': 1,
'Aphrodite': 1,
'Artemis': 1,
'As': 1,
'Christianity': 1,
'Dali': 2,
'George': 1,
'George.': 1,
'Georgia,': 1,
'Georgian': 1,
'Greek': 1,
'Her': 1,
'Ishtar.': 1,
'Saint': 2,
'Scottish': 1,
'She': 1,
'Stories': 1,
'The': 1,
'a': 7,
'about': 1,
'ali.': 1,
'although': 1,
'and': 8,
'animals,': 1,
'animals.': 1,
'as': 3,
'associations': 1,
'be': 1,
'beautiful': 1,
'been': 1,
'began': 1,
'called': 2,
'changed;': 1,
'clashing': 1,
'compared': 1,
'conflated': 1,
'connections': 1,
'depict': 1,
'described': 1,
'draw': 1,
```

'favored': 1,
'folklore': 1,
'form': 1,
'from': 2,
'glaistig,': 1,
'glowing': 1,
'goddess': 1,
'goddesses': 1,
'gold,': 1,
'golden': 1,
'hag': 1,
'hair': 1,
'has': 1,
'have': 1,
'having': 1,
'her': 4,
'her,': 1,
'hoofed': 1,
'human': 1,
'hunt,': 1,
'hunters': 1,
'hunting': 1,
'in': 2,
'is': 1,
'jealously': 1,
'killing': 1,
'later': 1,
'led': 1,
'lovers': 1,
'maiden': 1,
'malicious': 1,
'morning': 1,
'mountain': 1,
'mythology': 2,
'mythology,': 1,
'nature': 1,
'nude': 1,
'obeyed': 1,
'of': 6,
'on': 1,
'overrule': 1,
'patron': 2,
'people.': 1,
'power': 1,
'presented': 1,
'punish': 1,
'reward': 1,
'rise': 1,
'rival': 1,
'said': 1,
'scholars': 1,

```
'seduction,': 1,
'she': 4,
'skin,': 1,
'sometimes': 1,
'spirit': 1,
'star': 1,
'stories': 1,
'such': 1,
'taboos': 1,
'taking': 1,
'tames': 1,
'the': 10,
'them,': 1,
'to': 5,
'told': 1,
'took': 1,
'unicorn.': 1,
'usually': 1,
'violators.': 1,
'was': 3,
'who': 2,
'wild': 1,
'with': 6,
'woman': 1}

"""
```

Pratik Pingale 19CO056

Page No.
Date

Experiment A-9

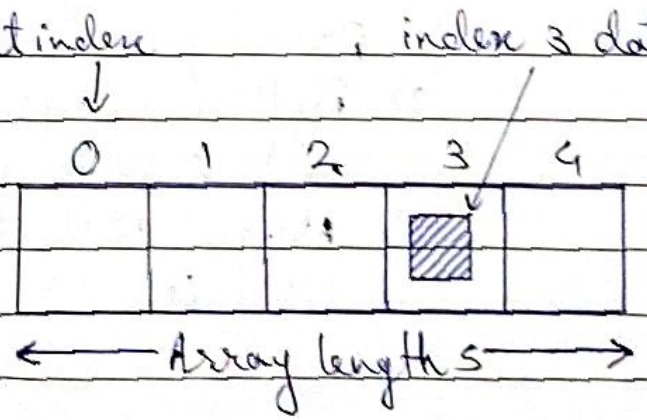
AIM: Write a python program to compute following operations on matrix
a) Addition
b) Subtraction
c) Multiplication
d) Transpose
Perform these operation on matrix

THEORY: Python:
Python is a high purpose general level and very popular programming language. Python programming language is used in web development, Machine Learning along with all cutting edge technology in software technol. Python programming language is very well suited for Beginners as well as experienced programmers.

Features of python:
1) Widely used for high level multi-purpose tasks.
2) Follow Object oriented and Procedure paradigms
3) Python programs are generally redab and easy to write.
4) Huge collections of standard libraries.

→ Python array

1) An array is a collection of data stored at contiguous memory location.
2) It stores data of same kind together
3) This makes easier to access element by directly adding offset of base value.
4) Array name is the location of 1st element of array.
5) Array in python can be handled by either key word 'list' or a module 'array'
6) Array are useful and efficent when we had to manipulate a similar data type.
7) Auser can treat list as an array.
8) However a user can't restrain data types in list.
9) In array module all elements of array are of same type.



Type Code	C-Type	Python type	Byte
b	signed char	int	1
B	unsigned char	int	1
u	Py_UNICODE	unicode char	2
h	signed short	int	2
H	unsigned short	int	2
i	signed int	int	2
I	unsigned int	int	2
l	signed long	int	4
L	unsigned long	int	4
q	signed long long	int	8
Q	unsigned long long	int	8
f	float	float	4
d	double	float	8

Basic Operations :

1) Traverse :- Accessing one element one by one

2) Insertion :- Adding array to element at start, end or in between.

3) Delete :- Delete an element from start, end or its first occouring.

4) Search :- Find an element in an array with matches given value

5) Update :- Update the array with new value

6) Length :- Using builtin len() function to find array length.

• Example:

```
import array
arr = array.array('i', [1, 2, 3])
print('Given array is: ', end= '')
for i in range(3):
    print(a[i], end= '').
print().
```

Output: Given array is : 1 2 3

• Two dimensional array:

1) It is an array with an array.
2) Element position is indicated with 2 ~~an~~ index at a time.
3) 2 D array represent row and column.

Matrix : [[1, 2, 3], → rows
[4, 5, 6],
[7, 8, 9]]

↑ Column

• Accessing element

1) 2 index are used to access element in 2-D array.
2) I st index represent the row number i.e the index of the row list of main array.
3) II nd index represent the column number, i.e. index of elendent ~~of~~ in row th element of main array.

```
M = [[1, 2, 3], [4, 5, 6], [7, 8, 9]].
print (M(2, 1))
```

→ 8.

```
# print array. M.
for i in M:
    for j in i:
        print( j , end= ' ')
```

→ 1, 2, 3, 4, 5, 6, 7, 8, 9.

CONCLUSION: In this experiment we learnt array implementation. We also learnt matrix operation like addition, subtraction, multiplication and transpose.

```python
# DSL - ASSIGNMENT 3 - A9

class Matrix:
    def __init__(self, row, col):
        self.row = row
        self.col = col
        self.matrix_A = [i[:] for i in [[0]*self.col]*self.row]
        self.matrix_B = [i[:] for i in [[0]*self.col]*self.row]
        self.matrix_C = [i[:] for i in [[0]*self.col]*self.row]


    def getinp(self, mat):
        for i in range(self.row):
            for j in range(self.col):
                mat[i][j] = int(input(f'Enter the element of row {i+1} and column {j+1}: '))

    def printmat(self, mat):
        for i in mat:
            print('\t'.join(map(str, i)))

    def getmatrix(self):
        print('\n\nEnter Matrix A data: ')
        self.getinp(self.matrix_A)

        print('\nEnter Matrix B data: ')
        self.getinp(self.matrix_B)

        print('\n\nMatrix A')
        self.printmat(self.matrix_A)

        print('\nMatrix B')
        self.printmat(self.matrix_B)


    def addition(self):
        for i in range(self.row):
            for j in range(self.col):
                self.matrix_C[i][j] = self.matrix_A[i][j] + self.matrix_B[i][j]

        print("\n\nAddition matrix: ")
        self.printmat(self.matrix_C)


    def subtraction(self):
        for i in range(self.row):
            for j in range(self.col):
                self.matrix_C[i][j] = self.matrix_A[i][j] - self.matrix_B[i][j]

        print("\n\nSubtraction matrix: ")
        self.printmat(self.matrix_C)
```

```python
    def multiply(self):
        for i in range(self.row):
            for j in range(self.col):
                self.matrix_C[i][j] = 0
                for k in range(self.row):
                    self.matrix_C[i][j] += self.matrix_A[i][k] * self.matrix_B[k][j]

        print("\n\nMultiplication matrix: ")
        self.printmat(self.matrix_C)


    def transpose(self):
        for i in range(self.row):
            for j in range(self.col):
                self.matrix_C[i][j] = self.matrix_A[j][i]

        print("\n\nTranspose of Matrix A: ")
        self.printmat(self.matrix_C)

        for i in range(self.row):
            for j in range(self.col):
                self.matrix_C[i][j] = self.matrix_B[j][i]

        print("\n\nTranspose of Matrix B: ")
        self.printmat(self.matrix_C)

def main():
    row = int(input('\nEnter number of rows: '))
    col = int(input('Enter number of columns: '))
    mat = Matrix(row, col)
    mat.getmatrix()
    mat.addition()
    mat.subtraction()
    mat.multiply()
    mat.transpose()

if __name__ == "__main__":
    main()
```

```
"""

──────────────── OUTPUT ─────────────────

Enter number of rows: 3
Enter number of columns: 3


Enter Matrix A data:
Enter the element of row 1 and column 1: 1
Enter the element of row 1 and column 2: 5
Enter the element of row 1 and column 3: 2
Enter the element of row 2 and column 1: 4
Enter the element of row 2 and column 2: 3
Enter the element of row 2 and column 3: 2
Enter the element of row 3 and column 1: 7
Enter the element of row 3 and column 2: 4
Enter the element of row 3 and column 3: 5

Enter Matrix B data:
Enter the element of row 1 and column 1: 5
Enter the element of row 1 and column 2: 4
Enter the element of row 1 and column 3: 1
Enter the element of row 2 and column 1: 2
Enter the element of row 2 and column 2: 7
Enter the element of row 2 and column 3: 4
Enter the element of row 3 and column 1: 3
Enter the element of row 3 and column 2: 6
Enter the element of row 3 and column 3: 1


Matrix A
1 5 2
4 3 2
7 4 5

Matrix B
5 4 1
2 7 4
3 6 1


Addition matrix:
6 9 3
6 10 6
10 10 6


Subtraction matrix:
-4 1 1
```

```
2 -4 -2
4 -2 4


Multiplication matrix:
21 51 23
32 49 18
58 86 28


Transpose of Matrix A:
1 4 7
5 3 4
2 2 5


Transpose of Matrix B:
5 2 3
4 7 6
1 4 1

"""
```

PRATIK PINGALE 19CO056

PAGE NO.
DATE:

EXPERIMENT - B 11

AIM: a) Write a Python program to store roll no. of student in array who attended training prog. in random order. Search a roll no using Linear and sentinal search.
b) Now take input in sorted order. Search a roll no. using Binary & fibonacci search.

THEORY: 1) Linear Search:

- Simple search algorith where each element of array is compared with the search element
- Searching starts from left to right.
- Returns True if element found else
- returns -1 (i.e. No element found).

- Code Example:

```
def linearsearch (arr, n, s):
    for i in range (n):
        if (arr[i] == s):
            return i
    return -1

print([1, 2, 3], 3 , 2).
```

→ 0

PAGE NO.
DATE:

- Time complexity of linear search is $O(n)$
- Due to linear time complexity, linear search is not widely used.

2) Sentinal search:

- Last array element is replaced by search element and linear search is performed on array.
- Followed by an if condition checking wether index is less than array length or ~~ei~~ last element is search element or not.
- No. of comparision in worst case is $(n+2)$

Code Example:

```
def sentinelsearch (arr, n, s):
    last = arr[n-1].
    arr[n-1], i = ~~val~~ s, 0
    while (arr[i] != s):
        i += 1
    arr[n-1] = last
    return ((i < n-1) or (val == arr[n-1]))

if sentinelsearch ([1, 2, 3, 4, 5], 5, 2):
    print ('Element found')
else:
    print ('Element not found').
```

→ Element found.

PAGE NO.
DATE:

3) Binary Search

- In binary search an array is divided in 2 sub array and the element is searched in them.
- This process goes on till length of sub array becomes 1.
- Search element is compared with the center element of the array $\left(\frac{n+1}{2}\right)^{th}$ element
- If above condition false then a comparision is done with the centre element
- If search elem > center elem. repeat process from right to center elem else left.

Example code:

```
def binarysearch (arr, val, n):
    s=0
    while s ≤ n:
        mid = (s+n)//2
        if arr[mid] == val:
            return 'Element found'
        elif arr[mid] < val:
            s = mid+1
        else:
            n = mid-1
print (binarysearch ([1, 2, 3, 4, 5], 4, 5)
```

→ Element found.

PAGE NO.
DATE:

4) Fibonacci search:

- This comparision based technique uses fibonacci search to find element
- Fibonacci number: $F(n) = F(n-1) + F(n-2)$
- $F(n-2) \approx 1/3 * F(n)$
  $F(n-1) \approx 2/3 * F(n)$.
- Algorithm:

a) $F(n)$, $F(n-1)$, $F(n-2)$ are calculated such that $F(n) < n$.

b) let the $F(n-2)^{th}$ element be x.

c) if val. $< x$ then move the 3 fibonacci number down by 2 step eleminating 2/3 of all

d) elif val $> x$ then move the 3 fibonacci number up by 1 step eleminating 1/3 of all.

e) else val == x return index.

f) Now only 1 elem remains so if $f(n-1)$ is 1 and val == arr[index]. return True.

- Fibonacci search is fast and efficient search algorithm but requires sorted array.

CONCLUSION: From this experiment we learnt proper implementation of linear, sentinel, binary and fibonacci search on sorted or unsorted array.

```python
# DSL - ASSIGNMENT 4 - B11




def linearSearch(arr, n):
    val = int(input('Enter the Roll no you want to search using linear search: '))

    for i in range(n):
        if arr[i] == val:
            print(f'Roll no {val} found at position {i + 1}')
            return

    print(f'Roll no {val} not found')
    return




def sentinelSearch(arr, n):
    val = int(input('\n\nEnter the Roll no you want to search using sentinal search: '))

    last = arr[n - 1]

    arr[n - 1] = val
    i = 0

    while (arr[i] != val):
        i += 1

    arr[n - 1] = last

    if ((i < n - 1) or (val == arr[n - 1])):
        print(f'Roll no {val} found at position {i + 1}')
        return
    else:
        print(f'Roll no {val} not found')
        return
```

```python
def binarySearch(arr, n):
    val = int(input('Enter the Roll no you want to search using binary search: '))

    s = 0
    while s ⩽ n:
        mid = (s + n) // 2
        if arr[mid] == val:
            print(f'Roll no {val} found at position {mid + 1}')
            return

        elif arr[mid] < val:
            s = mid + 1

        else:
            n = mid - 1

    print(f'Roll no {val} not found')
    return

def fibonacciSearch(arr, n):
    val = int(input('\n\nEnter the Roll no you want to search using fibonacci search: '))

    fib_2 = 0
    fib_1 = 1
    fib_N = fib_1 + fib_2

    if n == 0:
        return 0

    while fib_N < n:
        fib_2 = fib_1
        fib_1 = fib_N
        fib_N = fib_1 + fib_2
    index = -1

    while fib_N > 1:
        i = min(index + fib_2, (n - 1))
        if arr[i] < val:
            fib_N = fib_1
            fib_1 = fib_2
            fib_2 = fib_N - fib_1
            index = i
        elif arr[i] > val:
            fib_N = fib_2
            fib_1 = fib_1 - fib_2
            fib_2 = fib_N - fib_1
        else:
            print(f'Roll no {val} found at position {i + 1}')
            return
```

```python
    if (fib_1 and index < n - 1) and (arr[index + 1] == val):
        print(f'Roll no {val} found at position {index + 2}')
        return

    print(f'Roll no {val} not found')
    return


def putdata(arr, n):
    print('\n\nFollowing are the student\'s roll no who have attended training program... \n')
    print('********************************')
    print('|   Position    |    Roll No    |')
    print('********************************')

    for i in range(n):
        print(f'|\t{i + 1}\t|\t{arr[i]}\t|')
    print('————————————————————————————————\n\n')


def main():
    n = int(input('\nEnter the total strenght that have attended training program: '))
    arr = []

    for i in range(n):
        arr.append(int(input(f'Enter the roll numbers: ')))
    putdata(arr, n)

    linearSearch(arr, n)

    sentinelSearch(arr, n)

    arr.sort()
    print('\n\nSorting the roll numbers gives:')
    putdata(arr, n)

    binarySearch(arr, n - 1)

    fibonacciSearch(arr, n)


if __name__ == "__main__":
    main()
```

```
"""

──────────────── OUTPUT ────────────────

Enter the total strenght that have attended training program: 10

Enter the roll numbers: 12
Enter the roll numbers: 82
Enter the roll numbers: 73
Enter the roll numbers: 34
Enter the roll numbers: 14
Enter the roll numbers: 98
Enter the roll numbers: 89
Enter the roll numbers: 11
Enter the roll numbers: 45
Enter the roll numbers: 29


Following are the student's roll no who have attended training program...

*********************************
|   Position    |    Roll No    |
*********************************
|       1       |       12      |
|       2       |       82      |
|       3       |       73      |
|       4       |       34      |
|       5       |       14      |
|       6       |       98      |
|       7       |       89      |
|       8       |       11      |
|       9       |       45      |
|       10      |       29      |
─────────────────────────────────


Enter the Roll no you want to search using linear search: 11
Roll no 11 found at position 8


Enter the Roll no you want to search using sentinal search: 14
Roll no 14 found at position 5


Sorting the roll numbers gives:


Following are the student's roll no who have attended training program...
```

```
*********************************
|    Position     |     Roll No    |
*********************************
|       1         |       11       |
|       2         |       12       |
|       3         |       14       |
|       4         |       29       |
|       5         |       34       |
|       6         |       45       |
|       7         |       73       |
|       8         |       82       |
|       9         |       89       |
|       10        |       98       |
_________________________________


Enter the Roll no you want to search using binary search: 12
Roll no 12 found at position 2


Enter the Roll no you want to search using fibonacci search: 98
Roll no 98 found at position 10

"""
```

PRATIK PINGALE 19C0056

PAGE NO.
DATE:

EXPERIMENT - B14

AIM: Write python Program to store first year percentage of student in array. Sort the array using
a) selection Sort
b) Bubble Sort

THEORY: 1) Selection Sort:

- Repeatedly find minimum of the array, removes it and put at the start of the array.
- Maintain 2 sub array at a time.
  - one sorted at the beginning and
  - other unsorted at the end.
- In every interation start element for search is the element which comes after the sorted part of list.

- Code example:

```
def selection sort (arr, n):
    for i in range(n):
        m = i
        for j in range (i+1, n):
            if arr[m] > arr[j]:
                m = j
        arr[i], arr[m] = arr[m], arr[i].
```

PAGE NO.
DATE:

a = [3, 4, 1, 5, 2].
selectionsort (a)
print (a)

→ [1, 2, 3, 4, 5].

- Time complexity of selection sort is $O(n^2)$.

- [1, 4, 3, 5, 2] 1st iteration
- [1, 2, 3, 5, 4]. 2nd iteration
- [1, 2, 3, 5, 4] 3rd iteration
- [1, 2, 3, 4, 5] 4th iteration.

2) Bubble sort:

- Simplest sorting algorithm.
- Repeatedly swaps adjacent elements if ~~as~~ they are in wrong order.
- Bubble sort becomes fatally inefficient as the array length grows.
- Time complexity of bubble sort is $O(n*n)$

1) 4213 → 2413 → 2143 → 2134
2) 2134 → 1234 → 1234 → 1234
3) 1234 → 1234 → 1234 → 1234

∴ list sorted

PAGE NO.
DATE:

Code Example:

```
def bubblesort (arr, n):
    for i in range(n):
        for j in range (0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr
                                   [j].

a = [5, 2, 1, 4, 3].
bubblesort (a).
print (a).
```

→ [1, 2, 3, 4, 5].

CONCLUSION: Hence we can now implement bubble, selection ~~so~~ sort using python on a array.

```python
# DSL - ASSIGNMENT 5 - B14

def bubbleSort(arr, n):
    index = list(range(1, n + 1))

    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j+1] :
                arr[j], arr[j+1] = arr[j+1], arr[j]
                index[j], index[j+1] = index[j+1], index[j]

    return index


def selectionSort(arr, n):
    index = list(range(1, n + 1))

    for i in range(n):
        min_idx = i
        for j in range(i + 1, n):
            if arr[min_idx] > arr[j]:
                min_idx = j

        arr[i], arr[min_idx] = arr[min_idx], arr[i]
        index[i], index[min_idx] = index[min_idx], index[i]

    return index


def putdata(arr, n):
    print('\n\nFollowing are the percentages of all the students ... \n')
    print('**********************************')
    print('|    Roll No    |   Percentage   |')
    print('**********************************')

    for i in range(n):
        print(f'|\t{i + 1}\t|\t{arr[i]}%\t |')
    print('————————————————————————————————\n\n')


def putsorteddata(arr, n):
    print('\n**********************************')
    print('|    Roll No    |   Percentage   |')
    print('**********************************')

    for i in range(n):
        print(f'|\t{arr[i][0]}\t|\t{arr[i][1]}%\t |')
    print('————————————————————————————————\n\n')
```

```python
def main():
    n = int(input('\nEnter number of student in 1st year: '))
    arr = []

    for i in range(n):
        arr.append(int(input(f'Enter percentage of roll no {i + 1}: ')))

    putdata(arr, n)
    arr2 = arr[:]
    n = len(arr)

    print('Sorting all the percentages using Bubble Sort:')
    index = bubbleSort(arr, n)
    putsorteddata(list(zip(index, arr)), n)

    print('Sorting all the percentages using Selection Sort:')
    index = selectionSort(arr2, n)
    putsorteddata(list(zip(index, arr2)), n)


if __name__ == "__main__":
    main()
```

```
"""

──────────────── OUTPUT ────────────────

Following are the percentages of all the students...

**********************************
|     Roll No     |   Percentage   |
**********************************
|        1        |       85%      |
|        2        |       82%      |
|        3        |       77%      |
|        4        |       91%      |
|        5        |       89%      |
|        6        |       45%      |
|        7        |       62%      |
|        8        |       73%      |
|        9        |       99%      |
|        10       |       95%      |
──────────────────────────────────


Sorting all the percentages using Bubble Sort:

**********************************
|     Roll No     |   Percentage   |
**********************************
|        6        |       45%      |
|        7        |       62%      |
|        8        |       73%      |
|        3        |       77%      |
|        2        |       82%      |
|        1        |       85%      |
|        5        |       89%      |
|        4        |       91%      |
|        10       |       95%      |
|        9        |       99%      |
──────────────────────────────────
```

```
Sorting all the percentages using Selection Sort:

**********************************
|     Roll No     |    Percentage    |
**********************************
|        6        |        45%       |
|        7        |        62%       |
|        8        |        73%       |
|        3        |        77%       |
|        2        |        82%       |
|        1        |        85%       |
|        5        |        89%       |
|        4        |        91%       |
|        10       |        95%       |
|        9        |        99%       |
____________________________________

"""
```

PRATIK PINGALE 19CO056

PAGE NO.
DATE:

EXPERIMENT - B-16

AIM : Write a python program to store 1st year percentage of student in array. Write a quick sort function using floating points in ascending order.

THEORY : Quick Sort →

1) Average time complexity of quick sort is O(n logn).
2) Quick sort is a divide and conquer type algorithm
3) Quick sort is not a stable as it exchanges non-adjacent element.
4) It picks an element as pivot and divides the given array in 2 parts.
5) Pivot can be either at, start, end, medium or at random position.
6) Partition( ) function is the most important function
7) All the array element greater than pivot are moved to the right of the pivot
8) All the array element ~~greater~~ less than pivot are moved to the left of the pivot.
9) Now quick sort is again applied to ~~the~~ both the partitions.

PAGE NO.
DATE:

• Algorithm: (Pseudocode).

```
quicksort(arr, low, high):
    if (low < high):
        # pi is partitioning index
        pi = partition(arr, low high)
        quicksort(arr, low, pi-1).
        quicksort(arr, pi+1, high).
```

- Calculate partition index whenever low is less than high
- Call ~~parti~~ quicksort for part of array which is less than pivot.
- Call quicksort for part of array which is greater than pivot.

• Pseudocode (Partition)

```
partition(arr[], low, high)
    pivot = arr[high].
    i = low - 1
    for j in range(low, high)
        if (arr[j] < pivot)
            i += 1
            swap arr[i] and arr[j]
    swap arr[i+1] and arr[high]
    return(i+1)
```

PAGE NO.
DATE:

- Set pivot as last array element
- Initialize i as low - 1 (smaller elem)
- Vary j from low to high
- For each j check if jth elem is less than pivot or not.
- If true then swap arr[i] and arr[j].
- Increment i by 1 i.e index of smaller element
- Finally swap the pivot i.e arr[high] with arr[i+1] i.e its correct position

CONCLUSION: Hence we can now implement quick sort on an unsorted array and sort it ascendingly order.

```python
# DSL - ASSIGNMENT 6 - B16


def partition(arr, index, low, high):
    i = (low - 1)
    pivot = arr[high]

    for j in range(low , high):
        if arr[j] < pivot:
            i = i + 1
            arr[i], arr[j] = arr[j], arr[i]
            index[i], index[j] = index[j], index[i]

    arr[i + 1], arr[high] = arr[high], arr[i + 1]
    index[i + 1], index[high] = index[high], index[i + 1]

    return i + 1

def quickSort(arr, index, low, high):

    if low < high:
        pi = partition(arr, index, low, high)
        quickSort(arr, index, low, pi - 1)
        quickSort(arr, index, pi + 1, high)

    return index


def putdata(arr, n):
    print('\n\nFollowing are the percentages of all the students ... \n')
    print('**********************************')
    print('|    Roll No    | Percentage |')
    print('**********************************')

    for i in range(n):
        print(f'|\t{i + 1}\t|\t{arr[i]}%\t |')
    print('————————————————————————————————\n\n')


def putsorteddata(arr, n):
    print('\n**********************************')
    print('|    Roll No    | Percentage |')
    print('**********************************')

    for i in range(n):
        print(f'|\t{arr[i][0]}\t|\t{arr[i][1]}%\t |')
    print('————————————————————————————————\n\n')
```

```python
def main():
    n = int(input('\nEnter number of student in 1st year: '))
    arr = []

    for i in range(n):
        arr.append(int(input(f'Enter percentage of roll no {i + 1}: ')))

    putdata(arr, n)
    n = len(arr)
    index = list(range(1, n + 1))

    print('Sorting all the percentages using Quick Sort:')
    index = quickSort(arr, index, 0, n - 1)
    putsorteddata(list(zip(index, arr)), n)


if __name__ == "__main__":
    main()
```

```
"""

──────────────── OUTPUT ────────────────

Following are the percentages of all the students...

**********************************
|    Roll No    |   Percentage   |
**********************************
|       1       |      85.6%     |
|       2       |      82.8%     |
|       3       |      77.7%     |
|       4       |      91.9%     |
|       5       |      89.08%    |
|       6       |      45.88%    |
|       7       |      62.4%     |
|       8       |      73.5%     |
|       9       |      99.02%    |
|       10      |      95.6%     |
__________________________________


Sorting all the percentages using Quick Sort:

**********************************
|    Roll No    |   Percentage   |
**********************************
|       6       |     45.88%     |
|       7       |     62.4%      |
|       8       |     73.5%      |
|       3       |     77.7%      |
|       2       |     82.8%      |
|       1       |     85.6%      |
|       5       |     89.08%     |
|       4       |     91.9%      |
|       10      |     95.6%      |
|       9       |     99.02%     |
__________________________________

"""
```

PRATIK PINGALE 19CO056

PAGE NO.
DATE:

EXPERIMENT - C19

AIM: Department of Computer Engineering has student's club named 'Pinnacle Club'. Student of 2nd, 3rd and final year of department can be granted membership on permission. Similarly one can cancel the membership of club. First node is reserve for President of the club and last node for secratary of club. Write C++ program to maintain club member's information using singly linked list. Store student PRN and name. Write functions to:

a) Add and delete president or secratary are members.
b) Compute total number of club members.
c) Display memper of club.
d) Concatenate 2 club list.

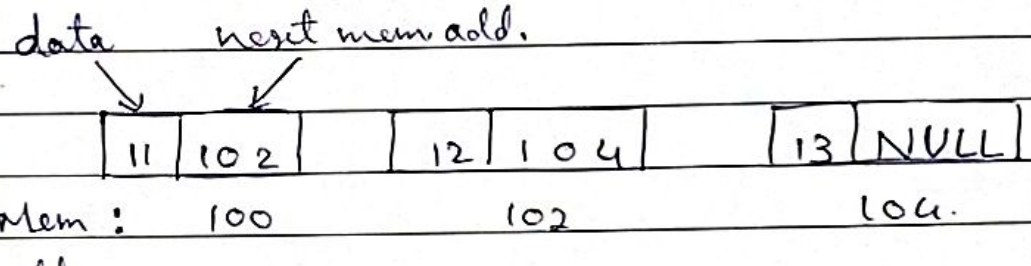
THEORY: • Singly linked list:

1) Also called singly or linear linked list
2) In singly linked list, each node contains 2 things: data and next.
3) Data contains the node data to be stored
4) Next links the current node to the next node

PAGE NO.
DATE:

5) It is a one way traversing list, i.e. ~~wha~~ we can traverse list in one direction only. and can't in backgsward direction.

6) First node is called head node and last node is sometime reffered as tail node.



7) Operation on Singly Linked List.

- Traversing : To access each element of the linked list

- Counting : Counts frequency of a given key.

- Searching : Data of a node is compared with the data user gives as input.

- Inserting : To insert an element either at beginning, at end or in between the list

PAGE NO.
DATE:

5) Deleting : Delete head node or tail node or a node with a given data or else a node at a certain position.

6) Reversing: Reversing a list such that head node becomes tail node and tail node becomes head node.

7) Sorting : Sorting a linked list using various sorting algorithms such as bubble sort, quick sort or selection sort.

• Reversing a linked list (Algorithm)

```
def reverse(self):
# Initialize prev(previous node) & curr
# (current node)
    prev = None
    curr = self.head.
# Starting from head, break its link with
# next and join with prev.
    while curr:
        next = curr.next
        # Attach prev to current's ~~head~~ next
        curr.next = prev
```

PAGE NO.
DATE:

```
# set prev to current and
# current to next i.e move up
# the linked list
prev = curr
curr = next

# Change the tail to head at last
self.head = prev.
```

CONCLUSION: Hence we learnt to implement singly linked list in c++ and performed various operation on it like traversal or deleting or reversing.

```cpp
/* DSL - EXPERIMENT 7 - C19 */

#include<iostream>
#include<string.h>
using namespace std;

class Member {
    public:
        int prn;
        char name[20];
        Member *next;
};

class Club {
    Member *root;
    Member *curr;
    Member *prev;

    public:
        Club() {
            root = NULL;
            curr = NULL;
            prev = NULL;
        }

        Member *GetRoot() {
            return (root);
        }

        void AddMembers() {

            while (true) {
                curr = new Member();
                curr->next = NULL;
                cout << "\nEnter PRN No: ";
                cin >> curr->prn;

                if(curr->prn == 0) break;

                cout << "\nEnter Member Name: ";
                cin >> curr->name;

                if (root == NULL) {
                    root = curr;
                    prev = curr;
                }
                else {
                    prev->next = curr;
                    prev = curr;
                }
            }
```

```
}

void DisplayMembers() {
    curr = root;

    while (curr ≠ NULL) {
        printf("(%d) %s ⟶ ", curr→prn, curr→name);
        curr = curr→next;
    }
    cout<<"NULL";
}

void CountMembers() {
    int c = 0;
    curr = root;

    while(curr ≠ NULL) {
        c++;
        curr = curr→next;
    }
    cout<<c;
}

void DisplayMembersReverse(Member *r) {
    if(r == NULL) return;
    DisplayMembersReverse(r→next);
    printf("(%d) %s ⟶ ", r→prn, r→name);
}

void DeleteMembers() {
    int prn;
    cout << "\nEnter PRN No. To Delete: ";
    cin >> prn;
    curr = root;
    prev = NULL;

    while (curr ≠ NULL) {
        if(prn == curr→prn) {

            // Root node
            if (curr == root){
                root = root→next;
                delete curr;
                break;
            }

            // Last node
            else if (curr→next==NULL) {
                prev→next = NULL;
                delete curr;
                break;
```

```
            }

            // Other nodes
            else {
                prev->next= curr->next;
                delete curr;
                break;
            }
        }
        prev = curr;
        curr = curr->next;
    }
}

Club operator + (Club c2) {
    Member *r1;
    Member *r2;
    Member *r3;
    Club c3;

    r1 = root;
    r2 = c2.root;

    while (r1≠NULL) {
        c3.curr = new Member();
        c3.curr->prn = r1->prn;
        strcpy(c3.curr->name,r1->name);
        c3.curr->next = NULL;
        if(c3.root == NULL) {
            c3.root = c3.curr;
            c3.prev = c3.curr;
        }
        else {
            c3.prev->next = c3.curr;
            c3.prev = c3.curr;
        }

        r1 = r1->next;
    }

    while (r2≠NULL) {
        c3.curr = new Member();
        c3.curr->prn = r2->prn;
        strcpy(c3.curr->name,r2->name);
        c3.curr->next = NULL;
        if(c3.root == NULL) {
            c3.root = c3.curr;
            c3.prev = c3.curr;
        }
        else {
            c3.prev->next = c3.curr;
```

```
                c3.prev = c3.curr;
            }
            r2 = r2->next;
        }

        return (c3);
    }
};

int main() {
    Club c1,c2,c3;
    int op = -1;

    while(op!=0) {
        cout << "\n\n1.Add Members";
        cout << "\n2.Display Members";
        cout << "\n3.Display Reverse";
        cout << "\n4.Count Members";
        cout << "\n5.Delete Members";
        cout << "\n6.Concatenate List Using Operator Overloading";
        cout << "\n0.Exit";
        cout << "\n\n*** Select Option *** -> ";
        cin >> op;

        switch(op) {
            case 1:
                cout << "\n\n*** Add To Club 1 *** \n";
                c1.AddMembers();
                cout << "\n\n*** Add To Club 2 *** \n";
                c2.AddMembers();
                break;

            case 2:
                cout << "\n\n*** Display Members - Club 1 *** \n\n";
                c1.DisplayMembers();
                cout<<"\n";
                cout << "\n*** Display Members - Club 2 *** \n\n";
                c2.DisplayMembers();
                cout<<"\n";
                break;

            case 3:
                cout << "\n\n*** Display Members Reverse - Club 1 *** \n\n";
                c1.DisplayMembersReverse(c1.GetRoot());
                cout << "\n\n*** Display Members Reverse - Club 2 *** \n\n";
                c2.DisplayMembersReverse(c2.GetRoot());
                cout<<"\n";
                break;

            case 4:
                cout<<"\n\nTotal Members Of Club 1 Are ";//<<c1.CountMembers();
```

```
            c1.CountMembers();
            cout<<"\n\nTotal Members Of Club 2 Are ";//<<c2.CountMembers();
            c2.CountMembers();
            cout<<"\n";
            break;

        case 5:
            cout << "\n\n*** Delete Members - Club 1 ***\n";
            c1.DeleteMembers();
            cout << "\n\n*** Delete Members - Club 2 ***\n";
            c2.DeleteMembers();
            cout<<"\n";
            break;

        case 6:
            cout << "\n\n*** Concatenating - Club 1 & Club 2***\n";
            c3 = c1 + c2;
            cout << "\n*** Display Members - Club 3 ***\n\n";
            c3.DisplayMembers();
            cout<<"\n";
            break;

        case 0:
            break;
        }
    }

    return 0;
}
```

```
/*

──────────────────── OUTPUT ────────────────────


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 1


*** Add To Club 1 ***

Enter PRN No: 123

Enter Member Name: Pratik

Enter PRN No: 456

Enter Member Name: Alex

Enter PRN No: 789

Enter Member Name: Jojo

Enter PRN No: 0


*** Add To Club 2 ***

Enter PRN No: 147

Enter Member Name: Shama

Enter PRN No: 258

Enter Member Name: Riya

Enter PRN No: 369

Enter Member Name: Yuki

Enter PRN No: 0
```

```
1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 2


*** Display Members - Club 1 ***

(123) Pratik ⟶ (456) Alex ⟶ (789) Jojo ⟶ NULL

*** Display Members - Club 2 ***

(147) Shama ⟶ (258) Riya ⟶ (369) Yuki ⟶ NULL


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 3


*** Display Members Reverse - Club 1 ***

(789) Jojo ⟶ (456) Alex ⟶ (123) Pratik ⟶

*** Display Members Reverse - Club 2 ***

(369) Yuki ⟶ (258) Riya ⟶ (147) Shama ⟶


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 4
```

```
Total Members Of Club 1 Are 3

Total Members Of Club 2 Are 3


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 5


*** Delete Members - Club 1 ***

Enter PRN No. To Delete: 456


*** Delete Members - Club 2 ***

Enter PRN No. To Delete: 369


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 2


*** Display Members - Club 1 ***

(123) Pratik ⟶ (789) Jojo ⟶ NULL

*** Display Members - Club 2 ***

(147) Shama ⟶ (258) Riya ⟶ NULL


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
```

```
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 4


Total Members Of Club 1 Are 2

Total Members Of Club 2 Are 2


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 6


*** Concatenating - Club 1 & Club 2***

*** Display Members - Club 3 ***

(123) Pratik ⟶ (789) Jojo ⟶ (147) Shama ⟶ (258) Riya ⟶ NULL


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 0

*/
```

PRATIK PINGALE 19CODS6

PAGE NO.
DATE:

EXPERIMENT - C21

AIM: Write a C++ program for storing appointment schedule for a day. Appointments are booked randomly using linked list. Set start and end time & min and max duration for slot. Write function for:-

a) Display free slot
b) Book appointment
c) Cancel appointment
d) Sort list based on time using pointer

Theory: Doubly linked list:-

1) Doubly linked list is a type of linked list in which can a node contains 3 thing: data, prev., next.

2) Data: Data contains the data that the node stores.

3). Next: Next links current node to the next node by storing its address in it.

4) Prev: Prev or previous stores address of previous node in it.

PAGE NO.
DATE:

5) In doubly linked list we can traverse in both directions as it has next as well as previous attribute.

- Representation of doubly linked list

(node)

Prev	data	next
NULL	10	102
100	20	104
102	30	NULL

Mem: 100 102 104.
add

- Previous of head node and next of tail node contains NULL.

- Advantages of doubly linked list:

1) A doubly linked list's node has 1 extra attribute compared to singly linked list i.e previous.

2) Previous stores a memory address of the previous node.

3) We can traverse a doubly linked list in both directions i.e Both forward

PAGE NO.
DATE:

and backward directions.

4) Deleting nodes in doubly linked list is made effective than singly linked list if pointer to the node to be deleted is given.

5)

- Operations on Doubly linked list

1) Traversing
2) Counting
3) Searching
4) Inserting at head
5) Inserting at tail.
6) Inserting at any other point
7) Deleting head node
8) Deleting tail node
9) Deleting any inbetween node.
10) Reversing
11) Sorting

- Sorting a doubly linked list

→ Algorithm.

1) Initialize int i = 0, struct node * sort;
2) Set sort = head.
3) while sort → next != NULL.
    check if sort → time > sort → next → time

PAGE NO.
DATE:

swap (sort , sort → next).

4) ~~In~~ Step up sort i.e sort = sort → next

5) Repeat step (2) n times (i.e array size)

CONCLUSION: We learnt to implement doubly linked list and performed various operation on it like sorting the list, inserting, deleting and displaying the list.

```
/* DSL - EXPERIMENT 8 - C21 */

#include<iostream>
#include<cstring>
using namespace std;

struct node {
    string name;
    int time;
    struct node *next;
    struct node *prev;
};

class App {
    struct node *head,*p,*temp,*q;
    int size = 0;

    public:
        App() {
            head = NULL;
            p = NULL;
            temp = NULL;
        }

        void bookappment() {

            // If head isn't initialized yet
            if (p == NULL) {
                p = new (struct node);
                cout << "Enter your name: ";
                cin >> p->name;
                cout << "Enter slot time: ";
                cin >> p->time;
                p->next = NULL;
                p->prev = NULL;
                head = p;
            }

            // Every next node other than head
            else {
                temp = new (struct node);
                cout << "Enter your name: ";
                cin >> temp->name;
                cout << "Enter slot time: ";
                cin >> temp->time;
                if (p->next != NULL) p = p->next;
                temp->prev = p;
                p->next = temp;
                temp->next = NULL;
            }
```

```
    size += 1;
}

void freeslot() {
    int h = 12, x;
    cout << "\n";

    while (h <= 18) {
        p = head;
        x = 0;

        while(p≠NULL) {
            if (p→time == h) x = 1;
            p = p→next;
        }

        if (x == 0) cout << "Time slot " << h << " is free\n";
        h++;
    }
}

void cancelappment()
{
    cout << "\nEnter your name to cancel your appointment: ";
    string n;
    cin >> n;
    p = head;

    while (p ≠ NULL) {
        if (p→name == n) {

            // deleting head node
            if (p == head) {
                head = p→next;
                if (head ≠ NULL) head→prev = NULL;
            }

            // delete last node
            else if (p→next == NULL) {
                temp = p→prev;
                if (temp ≠ NULL) temp→next = NULL;
            }

            // delete random position
            else {
                p→next→prev = p→prev;
                p→prev→next = p→next;
            }

            size -= 1;
            cout << "Appointment cancelled successfully!! \n";
```

```
                free(p);
                break;
            }

            p = p->next;
        }

        if (p == NULL) cout << "No name found. Try entering proper name\n";
    }

    void display() {
        int i, j, t;
        string s;
        struct node *sort;

        for (i = 0; i < size; i++) {
            sort = head;

            while (sort->next != NULL) {
                if (sort->time < sort->next->time) {

                    t = sort->time;
                    sort->time = sort->next->time;
                    sort->next->time = t;

                    s = sort->name;
                    sort->name = sort->next->name;
                    sort->next->name = s;
                }

                sort = sort->next;
            }
        }

        cout << "\n";

        while(sort != NULL) {
            cout << sort->name << " has an appointment at " << sort->time << " PM\n";
            sort = sort->prev;
        }
    }
};
int main() {
    App d;
    int ch;
    char opt;

    do {
        cout << "\n******** DOCTOR APPOINTMENT *******\n";
        cout << "1. Book appointment\n2. Freeslots\n3. Cancel Appointment\n4. Display
appointment list\n";
```

```
        cout << "\nEnter your choice: ";
        cin >> ch;
        switch (ch) {
            case 1:
                cout << "\nEnter slots from 12PM to 18PM and each slots of one hour\n";
                d.bookappment();
                break;
            case 2:
                d.freeslot();
                break;
            case 3:
                d.cancelappment();
                break;
            case 4:
                d.display();
                break;
        }

        cout << "\nDo you want to continue? [Y/n] ";
        cin >> opt;
    } while (opt == 'y' || opt == 'Y');

    return 0;
}
```

```
/*

──────────────── OUTPUT ────────────────

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 1

Enter slots from 12PM to 18PM and each slots of one hour
Enter your name: Ram
Enter slot time: 13

Do you want to continue? [Y/n] y

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 1

Enter slots from 12PM to 18PM and each slots of one hour
Enter your name: Karan
Enter slot time: 17

Do you want to continue? [Y/n] y

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 1

Enter slots from 12PM to 18PM and each slots of one hour
Enter your name: Sita
Enter slot time: 15

Do you want to continue? [Y/n] y

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
```

```
4. Display appointment list

Enter your choice: 2

Time slot 12 is free
Time slot 14 is free
Time slot 16 is free
Time slot 18 is free

Do you want to continue? [Y/n] y

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 4

Ram has an appointment at 13 PM
Sita has an appointment at 15 PM
Karan has an appointment at 17 PM

Do you want to continue? [Y/n] y

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 3

Enter your name to cancel your appointment: Karan
Appointment cancelled successfully!!

Do you want to continue? [Y/n] y

******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 3

Enter your name to cancel your appointment: Sita
Appointment cancelled successfully!!

Do you want to continue? [Y/n] y
```

```
******** DOCTOR APPOINTMENT *******
1. Book appointment
2. Freeslots
3. Cancel Appointment
4. Display appointment list

Enter your choice: 4

Ram has an appointment at 13 PM

Do you want to continue? [Y/n] n

*/
```

PRATIK PINGLALE 19CO056

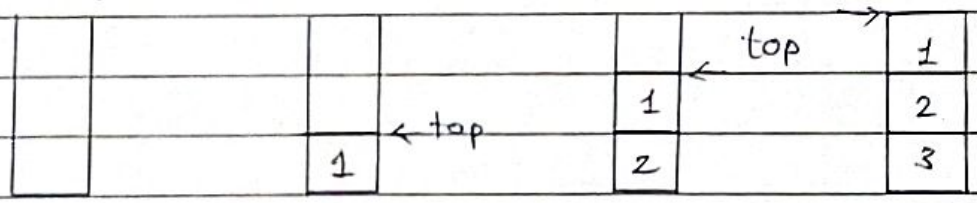
PAGE NO.
DATE:

EXPERIMENT - D-26

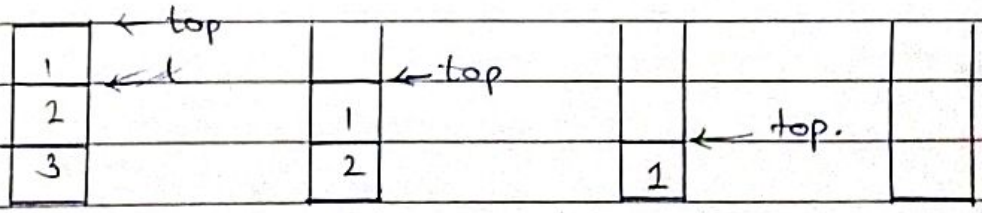
AIM : Write a C++ program using stacks to check whether a given expression is well parenthesized or not.

THEORY : Stack:

1) Stack Data Structure is based on Last In First Out (LIFO) Concept.
2) Stack is an ordered list of same type of element.
3) In linear list like stack insertion and deletion occurs only at one end of the list.
4) When elements are added to stack it grows at one end.
5) Similarly when an element is popped, it is popped at one end only.
6) Following depicts how insertion and popping happens in stack :



Insertion



Deletion

PAGE NO.
DATE:

• Operations on Stack

1) Initialize :
Stack can be initialized using array in both C++ and python.

2) isEmpty():
Check wheter a stack is empty or not

→ isEmpty() :
return top == -1

3) isFull():
Determines wheather a stack is full or not.

→ isFull() :
return top == MAX - 1

- Where MAX is array size.

4) push ():
Push function pushes an element in the stack.

→ push (data):
stack[++top] = data

5) pop () :
Removes top element of the stack

→ pop () :
return stack[--top].

PAGE NO.
DATE:

- Algorithm to verify wheather an infix expression is well parenthesized or not.

1) Read the expression
2) Loop through expr and let x be the character
3) if x is '(' or '{' or '['.
   push x into a stack
4) if x is ')' or '}' or ']'
   if stack top is '(' or '{' or '['
      pop from stack.
   else
      print " invalid expression "

CONCLUSION: Therefore from this experiment proper ~~element~~ implementation of stack as well as we learnt to check whether a expression is valid or not.

```cpp
/* DSL - EXPERIMENT 9 - D26 */

#include <iostream>
using namespace std;

#define MAX 25


// Class Stack
class Stack {
    private:
        char *data;                         // Dynamic allocation or else 'char data[MAX]'
        int top;

    public:
        Stack() {                           // Constructor initialization
            data = new char[MAX];           // Allocate 'data' with 'new char[MAX]'
            top = -1;
        }
        bool isEmpty();                     // Prototypes
        bool isFull();
        bool push(char x);
        char pop();
        char peek();
        void display();
};


// Class Parenthesis
class Parenthesis {
    char expn[MAX];
    Stack stack;

    public:
        void read();
        void checkexpn();
};


// Stack class function defintions
bool Stack::isEmpty() {
    return (top == -1);
}


bool Stack::isFull() {
    return (top == (MAX - 1));
}
```

```
bool Stack::push(char x) {
    if (isFull()) {
        cout << "Stack Overflow\n";
        return false;
    }
    else {
        data[++top] = x;
        cout << x << " : Pushed into stack\n";
        return true;
    }
}


char Stack::pop() {
    char x;
    if (isEmpty()) {
        cout << "Stack Underflow\n";
        return 0;
    }
    else{
        x = data[top--];
        cout << x << " : Popped from stack\n";
        return x;
    }
}


char Stack::peek() {
    if (isEmpty()) {
        cout << "Stack is Empty\n";
        return 0;
    }
    else
        return data[top];
}

void Stack::display() {
    cout << "\nCurrent stack is: ";
    for (int i = 0; i <= top; i++) {
        cout << data[i] << " ⟶ ";
    }
    cout << "NULL";
}

// Parenthesis class function defintions
void Parenthesis::read() {
    cout << "\nEnter the expression: "; cin >> expn;
    cout << "\n";
}
```

```
void Parenthesis::checkexpn() {
    int i,flag = 0;
    char ch;

    for (i = 0; expn[i] ≠ '\0'; i++) {
        if (expn[i] == '{' || expn[i] == '[' || expn[i] == '(')
            stack.push(expn[i]);

        if(expn[i] == '}' || expn[i] == ']' || expn[i] == ')') {
            if (!stack.isEmpty()) {
                ch = stack.pop();

                if (expn[i] == '}' && ch ≠ '{') {
                    flag = 1;
                    break;
                }

                if (expn[i] == ']' && ch ≠ '[') {
                    flag = 1;
                    break;
                }

                if (expn[i] == ')' && ch ≠ '(') {
                    flag = 1;
                    break;
                }
            }
        }
    }

    stack.display();

    if (flag == 0 && stack.isEmpty())
        cout<<"\n\nExpression is in well Paranthesis";
    else
        cout<<"\n\nExpression is not in well Paranthesis";
}


int main() {
    Parenthesis expr;
    expr.read();
    expr.checkexpn();
    return 0;
}
```

```
/*

──────────────────── OUTPUT ────────────────────

Enter the expression: [(a+b)/{c*(d+e)}]

[ : Pushed into stack
( : Pushed into stack
( : Popped from stack
{ : Pushed into stack
( : Pushed into stack
( : Popped from stack
{ : Popped from stack
[ : Popped from stack

Current stack is: NULL

Expression is in well Paranthesis


Enter the expression: [(a+b)/{c*(d+e

[ : Pushed into stack
( : Pushed into stack
( : Popped from stack
{ : Pushed into stack
( : Pushed into stack

Current stack is: [ ⟶ { ⟶ ( ⟶ NULL

Expression is not in well Paranthesis

*/
```

PRATIK PINGALE 19C0056

PAGE NO.
DATE:

EXPERIMENT - D-27

AIM: Write a c++ program to convert an infix expression to a postfix expression using stack using following conditions:

a) Operand and operator must be single character

b) Input infix expression must be in desired format

c) Only +, -, * and / operators are allowed.

THEORY: Expression notation:

1) Expression are composed of operands and operators

2) Operands are either number (0 - 9) or alphabets

3) Operators are of 3 types: unary and binary.

4) Expression consist of arithmetic operator which are binary operator

5) arithmetic operators are +, -, *, /, %.

Example : 3 + ((4+2) * 6 - 5) / 2

is a expression.

PAGE NO.
DATE:

• Infix expression :
- In this type of expression the arrangement of operators and operands are as follows

Infix expr = (Operand 1)(Operator)(operand 2).

Example: (a+b) , a + (b * c) ,
(a+ b/c) * (d-e).

- Parenthesis can be used in infix expr as they are at higher priority than operators.
- Infix expression is most commonly used expression form.

• Postfix expression :
- In this type of expression the arrangement of operator and operand are as follows.

Postfix expr = (Operand 1)(operand 2)(Operator)

Example: a b + , a b c * + ,
a b c / + d e - * ,

- Parenthesis are not used in postfix expression.

PAGE NO.
DATE:

• Algorithm to convert infix to postfix expression:

1) Scan the infix expression from left to right
2) If the scanned character is an operand, output it.
3) Else
   a) The precedence of scanned opt is ≥ precendence of opt at stack top, push it
   b) Else pop all operator whose prec ≥ precedence of scanned and output it. Push scanned to stack
4) If '(' occur, push it
5) If ')' occur, pop and output until '(' is encountered.
6) Repeat 2-6 till all the operators / operands in the infix expression is scanned successfully
7) Print the output in each of the above cases.
8) Pop the stack and output it till it becomes empty.
9) Stop.

PAGE NO.
DATE:

CONCLUSION : In this experiment we learnt to implement stack and at the same time learned to convert and infix expression to a postfix expression.

```cpp
/* DSL - EXPERIMENT 10 - D27 */

#include <iostream>
using namespace std;


#define MAX 50


// Class Stack
class Stack {
    private:
        char *data;                         // Dynamic allocation or else 'char data[MAX]'
        int top;

    public:
        Stack() {                           // Constructor initialization
            data = new char[MAX];           // Allocate 'data' with 'new char[MAX]'
            top = -1;
        }
        bool isEmpty();                     // Prototypes
        bool isFull();
        bool push(char x);
        char pop();
        char peek();
};


// Stack class function defintions
bool Stack::isEmpty() {
    return (top == -1);
}


bool Stack::isFull() {
    return (top == (MAX - 1));
}


bool Stack::push(char x) {
    if (isFull()) {
        cout << "Stack Overflow\n";
        return false;
    }
    else {
        data[++top] = x;
        return true;
    }
}
```

```
char Stack::pop() {
    char x;
    if (isEmpty()) {
        cout << "Stack Underflow\n";
        return 0;
    }
    else {
        x = data[top--];
        return x;
    }
}


char Stack::peek() {
    if (isEmpty()) {
        cout << "Stack is Empty\n";
        return 0;
    }
    else
        return data[top];
}


//Function to return precedence of operators
int prec(char c) {
    if(c == '^')
        return 3;
    else if(c == '*' || c == '/')
        return 2;
    else if(c == '+' || c == '-')
        return 1;
    else
        return -1;
}


bool isOperand(char c) {
    return (('a' ≤ c && c ≤ 'z') || ('A' ≤ c && c ≤ 'Z') || ('0' ≤ c && c ≤ '9'));
}


// The main function to convert infix expression to postfix expression
string infixToPostfix(string s) {
    Stack stack;
    string postfix;

    for(int i = 0; i < s.length(); i++) {

        // If the scanned character is an operand, add it to output string.
        if (isOperand(s[i]))
            postfix += s[i];
```

```
        // If the scanned character is an '(', push it to the stack.
        else if (s[i] == '(')
            stack.push('(');

        // If the scanned character is an ')',
        // pop and to output string from the stack until an '(' is encountered.
        else if (s[i] == ')') {
            while (!stack.isEmpty() && stack.peek() ≠ '(')
                postfix += stack.pop();

            if (stack.peek() == '(')
                stack.pop();
        }

        //If an operator is scanned
        else {
            while (!stack.isEmpty() && prec(s[i]) ≤ prec(stack.peek()))
                postfix += stack.pop();

            stack.push(s[i]);
        }

    }

    // Pop all the remaining elements from the stack
    while(!stack.isEmpty())
        postfix += stack.pop();

    return postfix;
}


// Evaluate Postfix expression
int evalPostfix(string pfexp) {
    Stack stack;

    for (char c : pfexp) {

        // If operand then push to stack
        if (isdigit(c))
            stack.push(c - '0');

        // If operator then pop 2 digits from sack,
        // use operator and push value to stack
        else {
            int val1 = stack.pop();
            int val2 = stack.pop();

            switch (c) {
                case '+':
```

```
                stack.push(val2 + val1);
                break;
            case '-':
                stack.push(val2 - val1);
                break;
            case '*':
                stack.push(val2 * val1);
                break;
            case '/':
                stack.push(val2 / val1);
                break;
        }
    }
}

    return stack.pop();
}


//Driver program to test above functions
int main() {
    string exp;
    string pfexp;
    cout << "\nEnter an Infix Expression : "; cin >> exp;
    pfexp = infixToPostfix(exp);
    cout << "\nPostfix Expression is given by : " << pfexp << endl;
    cout << "\nThis Postfix expression is evaluated as : " << evalPostfix(pfexp) << endl;
    return 0;
}
```

```
/*

──────────────── OUTPUT ────────────────

Enter an Infix Expression : 1+2-(4*(8/2-4))/3+6

Postfix Expression is given by : 12+482/4-*3/-6+

This Postfix expression is evaluated as : 9

*/
```

PRATIK PINGALE 19CO056

PAGE NO.
DATE:

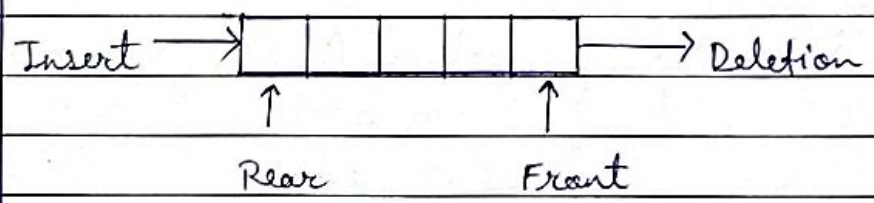
EXPERIMENT - E - 29

AI: M

Write a c++ program for simulating job queue. Write function to add job and delete job from queue

THEORY:

1) It is special kind of list.
2) In Queue, an item is added from one end of the list (rear)
2) Items are deleted from one end of the list (front).



• Application:

1) In Operating sytem. queue is necessary as OS has to schedule processes.
2) Queue for breadth first traversal.
3) Queue are also used in CPU for spooling

A queue is a client processes. waiting to recieve the service from server process.

PAGE NO.
DATE:

- Operations on queue

1) Initialize ( ) : Initialize a queue by setting the value of rear & front as -1

2) Enqueue ( ) : Insert at rear end of array an element

3) Requeue ( ) : Deletes front element and returns the same

4) empty ( ) : A boolean function that returns true or 1 if deque is empty else false or 0.

5) full ( ) : A boolean function that returns true or 1 if deque is full else false or 0.

6) print ( ) : Printing the queue elements.

- Enque ( ) :

1) In empty deque

R = F = 0

deq [F] = x // insert element x.

2) In predefined queue.

R += 1

PAGE NO.
DATE:

deq[R] = x

· dequeue ().
1) return data[F]
R = F = -1

CONCLUSION: Hence, by implementing ~~those~~ queue, we understood its concept and also various operations on queue using arrays. on job queue.

```cpp
/* DSL - EXPERIMENT 11 - E29 */

#include<iostream>
using namespace std;

#define MAX 5


class Job {
    int ID;
    friend class Queue;        //Queue can access private members of class Job

    public:
        void getdata() {
            cout << "\nEnter Job ID: "; cin >> ID;
        }
        void putdata() {
            cout << ID;
        }

};


class Queue {
    int front, rear;
    Job queue[MAX];

    public:
        Queue() {
            front = -1;
            rear = -1;
        }
        bool isEmpty();
        bool isFull();
        void insert();
        void remove();
        void display();
};


bool Queue::isEmpty() {
    return (front == (rear + 1) || rear == -1);
}


bool Queue::isFull() {
    return (rear == (MAX - 1));
}
```

```
void Queue::insert() {
    Job j;

    if (isFull())
        cout << "\nQueue is Full";

    else {
        j.getdata();

        // Empty queue
        if (isEmpty()) {
            front++;
            rear++;

            queue[rear] = j;
        }
        else {
            int i = rear;

            queue[i + 1] = j;
            rear++;
        }
        cout << "\nJob Added To Queue" << endl;
    }
}


void Queue::remove() {
    if (isEmpty())
        cout<<"\nQueue is Empty";

    else {
        for (int i = 0; i <= rear; i++)
            queue[i] = queue[i + 1];
        front++;
        rear--;
        cout<<"\nJob Processed From Queue"<<endl;
    }
}


void Queue::display() {
    if (isEmpty())
        cout<<"\nQueue is Empty.";
    else
        for(int i = 0; i <= rear; i++) {
            cout << "\n" << i + 1 << ") ";
            queue[i].putdata();
        }

}
```

```
int main() {
    int ch;
    Queue q;

    do {
        cout<<"\n\n**** MENU ****\n"
            <<"1. Insert job\n"
            <<"2. Display jobs\n"
            <<"3. Remove job\n"
            <<"0. Exit\n"
            <<"Choice: "; cin>>ch;

        switch(ch) {
            case 1:
                q.insert();
                break;

            case 2:
                cout << "\nJob ID:";
                q.display();
                break;

            case 3:
                q.remove();
                break;
        }
    } while(ch != 0);

    return 0;
}
```

```
/*

──────────────── OUTPUT ────────────────

**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Enter Job ID: 8546

Job Added To Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Enter Job ID: 452

Job Added To Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Enter Job ID: 33524

Job Added To Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Enter Job ID: 2457

Job Added To Queue
```

```
**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Enter Job ID: 4196

Job Added To Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Queue is Full

**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 2

Job ID:
1) 8546
2) 452
3) 33524
4) 2457
5) 4196

**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 3

Job Processed From Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 2
```

```
Job ID:
1) 452
2) 33524
3) 2457
4) 4196

**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 3

Job Processed From Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 2

Job ID:
1) 33524
2) 2457
3) 4196

**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 1

Enter Job ID: 832

Job Added To Queue


**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 2

Job ID:
1) 33524
2) 2457
```

```
3) 4196
4) 832

**** MENU ****
1. Insert job
2. Display jobs
3. Remove job
0. Exit
Choice: 0

*/
```

PRATIK PINGALE 19C0056

PAGE NO.
DATE:

EXPERIMENT - E-31

AIM: A double-ended queue is a special type of list in which addition and deletion of element can be done on both sides. Write a C++ program to stimulate deque with functions to add and delete elements on both side of array.

THEORY: Deque :-

1) A deque simply means a double ended queue
2) A deque can be used in place of both stack and queue.
3) In deque, insertion and deletion can be carried out at either front or rear end.
4) Dequeue are classified as:
   a) Input restricted deque
   b) output restricted deque.

• Input restricted deque →
1) Insert at rear
2) Delete at rear
3) Delete at front

• output restricted deque →
1) Delete at front
2) Insert at front

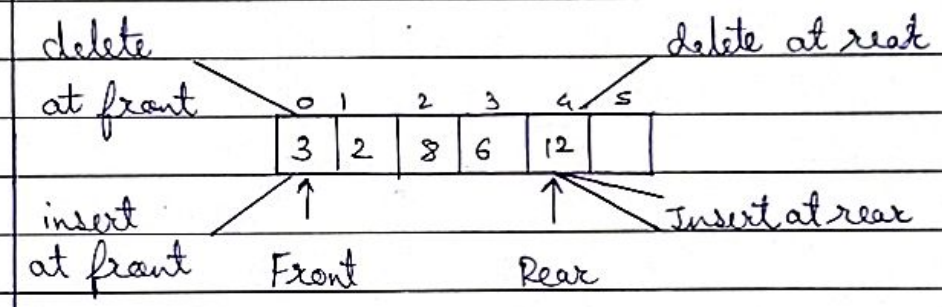
PAGE NO.
DATE:

3) Insert at rear.

- Deque implementation:

1) Using circular array.
2) Using singly linked list
3) Using singly circular linked list
4) Using doubly linked list
5) Using doubly circular linked list.

- Representation of deque.



- Operation on deque →

1) empty() → check whether deque is empty

2) full() → Whether deque is full or not

3) initialize() → Make queue empty and initialize R = F = -1

PAGE NO.
DATE:

4) enqueueR(): add element at rear. O(1).

5) enqueue F(): add element at front O(1).

6) dequeueR(): delete element from rear O(1).

7) dequeueF(): delete element from front O(1).

CONCLUSION

Therefore, we learnt to implement deque using linked list in C++ and performed various operations like enqueueR, enqueueF, dequeueR, dequeueF.

```
#include <iostream>
using namespace std;

#define MAX 10

// class representing a node
struct Node {
    int data;
    struct Node *next;
    struct Node *prev;

    Node () {
        next = NULL;
        prev = NULL;
    }
};


// Deuqe class
class Deque {
    struct Node *head, *tail, *temp;
    int size, front, rear;

    public:
        Deque ();
        bool isFull();
        bool isEmpty();
        voia pushfront(int);
        voia pushrear(int);
        int popfront();
        int poprear();
        voia display();

};


Deque::Deque () {
    head = tail = temp = NULL;
    size = 0;
    front = rear = -1;
}


bool Deque::isFull() {
    return ((front + rear == MAX - 1) || (front == MAX - 1) || (rear == MAX - 1));
}


bool Deque::isEmpty() {
    return ((front + rear == -1) || (head == NULL && tail == NULL));
}
```

```
void Deque::pushfront(int x) {
    if (isFull()) {
        cout << "\nDeque is Full. OVERFLOW!!!!";
        return;
    }

    // First node
    if (front == -1) {
        head = new (struct Node);
        head->data = x;
        tail = head;
        front++;
        rear++;
    }

    // Other node
    else {
        temp = new (struct Node);
        temp->data = x;
        temp->prev = tail;
        tail->next = temp;
        tail = temp;
        front++;
    }

    size++;
}


void Deque::pushrear(int x) {
    if (isFull()) {
        cout << "\nDeque is Full. OVERFLOW!!!!";
        return;
    }

    // First node
    if (front == -1) {
        head = new (struct Node);
        head->data = x;
        tail = head;
        front++;
        rear++;
    }

    // Other node
    else {
        rear++;
        temp = new (struct Node);
        temp->data = x;
        temp->next = head;
        head->prev = temp;
```

```
        head = temp;
    }

    size++;
}


int Deque::popfront() {
    int data;

    if (isEmpty()) {
        cout << "\nDeque is empty. UNDERFLOW!!!!";
        return -1;
    }
    data = tail->data;

    // Last node
    if (size == 1) {
        head = tail = NULL;
    }

    // Other node
    else {
        tail = tail->prev;
        tail->next = NULL;
    }

    front--;

    size--;
    return data;
}


int Deque::poprear() {
    int data;

    if (isEmpty()) {
        cout << "\nDeque is empty. UNDERFLOW!!!!";
        return -1;
    }
    data = head->data;

    // Last node
    if (size == 1) {
        head = tail = NULL;
    }

    // Other node
    else {
        head = head->next;
```

```
        head->prev = NULL;
    }

    rear--;

    size--;
    return data;
}


void Deque::display() {
    temp = head;
    cout << '\n';
    while (temp) {
        cout << temp->data << " ⟶ ";
        temp = temp->next;
    }
    cout << "\b\b\b\b    ";
}


int main() {
    Deque deque;
    int ch;

    while (1) {
        cout<< "\n\n1. Insert at front\n"
            << "2. Insert at rear\n"
            << "3. Pop from front\n"
            << "4. Pop from rear\n"
            << "5. Display Deque\n"
            << "6. Exit\n\n"
            << "Choose your option <1-6> : ";
        cin >>ch;

        switch(ch) {
            case 1:
                cout <<"Enter the Element: ";
                cin >> ch;
                deque.pushfront(ch);
                break;
            case 2:
                cout <<"Enter the Element: ";
                cin >> ch;
                deque.pushrear(ch);
                break;
            case 3:
                cout << "\nPopped element is : " << deque.popfront();
                break;
            case 4:
                cout << "\nPopped element is : " << deque.poprear();
```

```
            break;
        case 5:
            cout << "\nDeque is :\n";
            deque.display();
            break;
        case 6:
            exit(1);
        }
    }
}
```

```
/*

──────────────────── OUTPUT ────────────────────

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 1


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 2


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 3


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 4


1. Insert at front
2. Insert at rear
3. Pop from front
```

```
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 5


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 6


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 7


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 8


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :
```

```
8 ⟶ 6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3 ⟶ 5 ⟶ 7

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 3

Popped element is : 7

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

8 ⟶ 6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3 ⟶ 5

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 4

Popped element is : 8

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3 ⟶ 5
```

```
1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 3

Popped element is : 5

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 4

Popped element is : 6

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

4 ⟶ 2 ⟶ 1 ⟶ 3

1. Insert at front
2. Insert at rear
3. Pop from front
```

```
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 6

*/
```

PRATIK PINGALE 19CO056

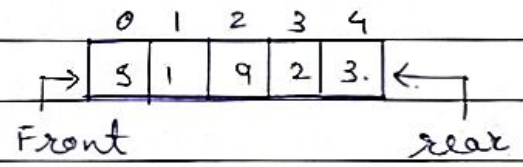
PAGE NO.
DATE:

EXPERIMENT - E-32

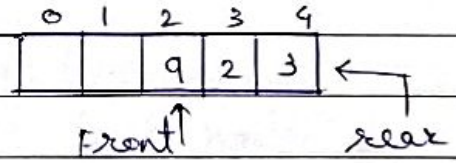
AIM: Pizza parlor accepting maximum Marders. Orders are served in first come first served basis. Order once placed cannot be cancelled. Write C++ program to stimulate the system using circular queue using array.

THEORY: Circular queue →

1) Make a queue using an array has a drawback
2). Even though if an array seems to be full, there might be free spaces in between.
3) Consider an array with 5 element in it.
   rear = 4 } indicates queue is full.
   front = 0 }
4) After 2 successive deletion:
   rear = 4 } queue seems to be full.
   front = 2 }
5).



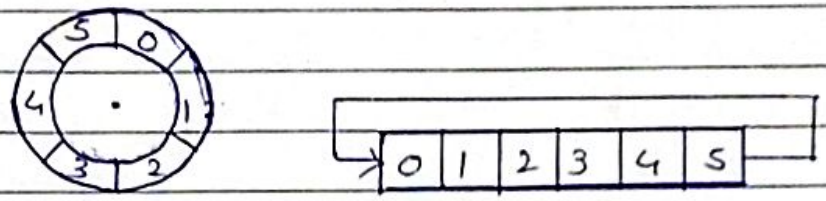
after deletion.



6) This can be solved by wrapping rear of the array along with the front.

PAGE NO.
DATE:

7) The array can now be represented as



- Implementation of circular queue.
  1) For a circular array to work with array, last element of the array should be linked with first element

  2) The movement of element in a circular looks like
     0, 1, 2, 3, 4, 0, 1, 2, 3, 4, 0 . . . .

     $i = (i+1)\%5$
     → Generates the circular index sequence.

- Circular queue using linked list:

  head : head node
  tail : tail node

  tail→next = head.
  head→prev = tail

PAGE NO.
DATE:

CONCLUSION: Therefore we can now implement a circular queue using linked list and performed various operation like adding are deleting element.

```cpp
/* DSL - EXPERIMENT 13 - E32 */

#include <iostream>
using namespace std;


#define MAX 5


string arr[3] = {"Veg Soya Pizza", "Veg butter Pizza", "Chicken Pizza"};

class Pizza {
    int order[MAX];
    int front,rear;

    public:
        Pizza() {
            front = rear = -1;
        }

        int isFull() {
            return ((front == 0) && (rear == (MAX - 1)) || (front == (rear + 1) % MAX));
        }

        int isEmpty() {
            return (front == -1);
        }

        void accept_order(int);
        void make_payment(int);
        void order_in_queue();
};


void Pizza::accept_order(int item) {

    if (isFull())
        cout<<"\nVery Sorry !!!! No more orders....\n";

    else {
        if (front == -1)
            front = rear = 0;

        else
            rear = (rear + 1) % MAX;

        order[rear] = item;
    }
}
```

```
void Pizza::make_payment(int n) {

    int item;
    char ans;

    if (isEmpty())
        cout<<"\nSorry !!! There are no pending orders....\n";

    else {
        cout<<"\nDeliverd orders as follows... \n";

        for(int i = 0; i < n; i++) {

            item = order[front];

            if(front == rear)
                front = rear = -1;

            else
                front = (front + 1) % MAX;

            cout << arr[item - 1] << " , ";
        }

        cout << "\n\nTotal amount to pay = " << n * 100;
        cout << "\nThank you visit Again....\n";
    }
}


void Pizza::order_in_queue() {
    int i;

    if (isEmpty())
        cout << "\nSorry !! There is no pending order... \n";

    else {
        i = front;
        cout << "\nPending Order as follows.. \n";

        while (i != rear) {
            cout << arr[order[i] - 1] << " , ";
            i = (i + 1) % MAX;
        }

        cout << arr[order[i] - 1] << "\n";
    }
}
```

```
int main()
{
    Pizza pizza;
    int ch,k,n;

    do {
        cout<< "\n\n\t***** Welcome To Pizza Parlor *******\n"
            << "\n1. Accept order\n2. Make_payment\n3. Pending Orders\n4. Exit"
            << "\nEnter your choice: ";
        cin >> ch;

        switch(ch) {
            case 1:
                cout<< "\nWhich Pizza do u like most....\n"
                    << "\n1. Veg Soya Pizza\n2. Veg butter Pizza\n3. Chicken Pizza"
                    << "\nPlease enter your order: ";
                cin >> k;
                pizza.accept_order(k);
                break;

            case 2:
                cout << "\nHow many Pizza do you want? ";
                cin >> n;
                pizza.make_payment(n);
                break;


            case 3:
                cout << "\nFollowing orders are in queue to deliver....as follows.. \n";
                pizza.order_in_queue();
                break;
        }
    } while (ch != 4);

    return 0;
}
```

```
/*

──────────────────── OUTPUT ────────────────────


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 1

Which Pizza do u like most....

1. Veg Soya Pizza
2. Veg butter Pizza
3. Chicken Pizza
Please enter your order: 1


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 1

Which Pizza do u like most....

1. Veg Soya Pizza
2. Veg butter Pizza
3. Chicken Pizza
Please enter your order: 2


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 1

Which Pizza do u like most....

1. Veg Soya Pizza
2. Veg butter Pizza
3. Chicken Pizza
Please enter your order: 3
```

```
        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 1

Which Pizza do u like most....

1. Veg Soya Pizza
2. Veg butter Pizza
3. Chicken Pizza
Please enter your order: 1


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 1

Which Pizza do u like most....

1. Veg Soya Pizza
2. Veg butter Pizza
3. Chicken Pizza
Please enter your order: 2


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 1

Which Pizza do u like most....

1. Veg Soya Pizza
2. Veg butter Pizza
3. Chicken Pizza
Please enter your order: 3

Very Sorry !!!! No more orders....
```

```
        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 3

Following orders are in queue to deliver....as follows..

Pending Order as follows..
Veg Soya Pizza , Veg butter Pizza , Chicken Pizza , Veg Soya Pizza , Veg butter Pizza


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 2

How many Pizza do you want? 3

Deliverd orders as follows...
Veg Soya Pizza , Veg butter Pizza , Chicken Pizza

Total amount to pay = 300
Thank you visit Again....


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 3

Following orders are in queue to deliver....as follows..

Pending Order as follows..
Veg Soya Pizza , Veg butter Pizza


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
```

```
Enter your choice: 2

How many Pizza do you want? 2

Deliverd orders as follows…
Veg Soya Pizza , Veg butter Pizza

Total amount to pay = 200
Thank you visit Again....


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 3

Following orders are in queue to deliver....as follows..

Sorry !! There is no pending order…


        ***** Welcome To Pizza Parlor *******

1. Accept order
2. Make_payment
3. Pending Orders
4. Exit
Enter your choice: 4

*/
```