**EXPERIMENT 4**

**CODE:**

str = input("Enter string: ")

list = str.split(" ")

#print(list)

print("Menu: \n1. Word with longest length\n2. frequency of occurrence of given character\n3. Check Palindrome\n4. Index of first appearance of substring\n5. Count of each word\n6. Exit\n")

while(1):

print("Enter choice: ")

ch = int(input())

if ch == 1:

lenMax = 0

idx = 0

for i in list:

l = len(i)

#print(l)

if l > lenMax:

lenMax = l

idx = list.index(i)

print("Word with longest length:"),

print('"%s"'%list[idx])

elif ch == 2:

chr = input("Enter character:")

count = 0

for i in str:

if i == chr:

count += 1

print("frequency of occurrence of %s:"%chr),

print(count)

elif ch == 3:

revString = str[::-1]

if revString == str:

print("Palindrome")

else:

print("Not Palindrome")

elif ch == 4:

substr = input("Enter subtring :")

a = str.find(substr)

print("Index of first appearance of '%s':"%substr),

if a != -1:

print(a)

else:

print("substring '%s' not found"%substr)

elif ch == 5:

list\_unique = []

for i in list:

if i not in list\_unique:

list\_unique.append(i)

#print("e)")

for i in list\_unique:

a = str.count(i)

print("Count of word '%s':"%i,a)

#print(a)

elif ch == 6:

exit()

**OUTPUT:**

PS H:\Coding\FDS\expt> & C:/Users/Dell/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Code/assignment5.py

Enter string: Hello how are you

Menu:

1. Word with longest length

2. frequency of occurrence of given character

3. Check Palindrome

4. Index of first appearance of substring

5. Count of each word

6. Exit

Enter choice:

1

Word with longest length:

"Hello"

Enter choice:

2

Enter character:l

frequency of occurrence of l:

2

Enter choice:

3

Not Palindrome

Enter choice:

4

Enter subtring :are

Index of first appearance of 'are':

10

Enter choice:

5

Count of word 'Hello': 1

Count of word 'how': 1

Count of word 'are': 1

Count of word 'you': 1

Enter choice:

6

**EXPERIMENT 5**

**CODE:**

n = int(input("Enter size of matrix: "))

print("Input matrix 1 row-wise")

matrix1 = []

for i in range(n):

list = []

for j in range(n):

a = int(input())

list.append(a)

matrix1.append(list)

print("Input matrix 2 row-wise")

matrix2 = []

for i in range(n):

list = []

for j in range(n):

a = int(input())

list.append(a)

matrix2.append(list)

print("Which operation you want to perform: ")

print("1. Addition of two matrices")

print("2. Subtraction of two matrices")

print("3. Multiplication of two matrices")

print("4. Transpose of a matrix1")

print("5. Exit")

def display(matrix,n):

for i in range(n):

for j in range(n):

print(matrix[i][j],end = " ")

print()

def addition(matrix1,matrix2,n):

matrix = []

for i in range(n):

list = []

for j in range(n):

list.append(matrix1[i][j] + matrix2[i][j])

matrix.append(list)

return matrix

def subtraction(matrix1,matrix2,n):

matrix = []

for i in range(n):

list = []

for j in range(n):

list.append(matrix1[i][j] - matrix2[i][j])

matrix.append(list)

return matrix

def multiply(matrix1,matrix2,n):

matrix = []

for row in range(n):

list = []

for col in range(n):

s = 0

for i in range(n):

s += matrix1[row][i]\*matrix2[i][col]

list.append(s)

matrix.append(list)

return matrix

def transpose(matrix1,n):

matrix = []

for i in range(n):

list = []

for j in range(n):

list.append(matrix1[j][i])

matrix.append(list)

return matrix

while(1):

ch = int(input("Enter no. of operation you want to perform: "))

if ch == 1:

matrix = addition(matrix1,matrix2,n)

print("matrix 1 + matrix 2: ")

display(matrix,n)

elif ch == 2:

matrix = subtraction(matrix1,matrix2,n)

print("matrix 1 - matrix 2: ")

display(matrix,n)

elif ch == 3:

matrix = multiply(matrix1,matrix2,n)

print("matrix 1 \* matrix 2: ")

display(matrix,n)

elif ch == 4:

matrix = transpose(matrix1,n)

print("Transpose of matrix: ")

display(matrix,n)

elif ch == 5:

exit()

**OUTPUT:**

PS H:\Coding\FDS\expt> & C:/Users/Dell/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Code/assignment9.py

Enter size of matrix: 3

Input matrix 1 row-wise

1

2

3

4

5

6

7

8

9

Input matrix 2 row-wise

2

4

5

6

1

3

8

7

9

Which operation you want to perform:

1. Addition of two matrices

2. Subtraction of two matrices

3. Multiplication of two matrices

4. Transpose of a matrix

5. Exit

Enter no. of operation you want to perform: 1

matrix 1 + matrix 2:

3 6 8

10 6 9

15 15 18

Enter no. of operation you want to perform: 2

matrix 1 - matrix 2:

-1 -2 -2

-2 4 3

-1 1 0

Enter no. of operation you want to perform: 3

matrix 1 \* matrix 2:

38 27 38

86 63 89

134 99 140

Enter no. of operation you want to perform: 4

Transpose of matrix:

1 4 7

2 5 8

3 6 9

Enter no. of operation you want to perform: 5

**EXPERIMENT 6**

**CODE:**

print("\n\*\*\*\*\*\*\*\*PROGRAM FOR SPARSE MATRIX OPERATIONS\*\*\*\*\*\*\*\*\*\*")

def add(matrix1, matrix2):

matrix3 = []

n = matrix1[0][2]+1

matrix3.append([matrix1[0][0],matrix1[0][1],matrix1[0][2]])

i = 1; j = 1; k = 1

while i < n and j < n:

if matrix1[i][0] == matrix2[j][0]:

if matrix1[i][1] == matrix2[j][1]:

s = matrix1[i][2] + matrix2[j][2]

matrix3.append([matrix1[i][0],matrix1[i][1],s])

i += 1

j += 1

k += 1

else :

if matrix1[i][1] < matrix2[j][1] :

matrix3.append([matrix1[i][0],matrix1[i][1],matrix1[i][2]])

i += 1

k += 1

else :

matrix3.append([matrix2[j][0],matrix2[j][1],matrix2[j][2]])

j += 1

k += 1

else:

if(matrix1[i][0] < matrix2[j][0]):

matrix3.append([matrix1[i][0],matrix1[i][1],matrix1[i][2]])

k += 1

i += 1

else :

matrix3.append([matrix2[j][0],matrix2[j][1],matrix2[j][2]])

k += 1

j += 1

if i < n:

for x in range(i,n):

matrix3.append(matrix1[x])

k += 1

if j < n:

for x in range(j,n):

matrix3.append(matrix2[x])

k += 1

matrix3[0][2] = k-1

return matrix3

def simpleTranspose(matrix):

n = matrix[0][2]+1

matrix2 = []

matrix2.append([matrix[0][1],matrix[0][0],matrix[0][2]])

for i in range(matrix[0][1]):

for j in range(1,n):

if matrix[j][1] == i:

list = [matrix[j][1],matrix[j][0],matrix[j][2]]

matrix2.append(list)

return matrix2

def fastTranspose(matrix):

n = matrix[0][2]+1

matrix2 = [[0]]\*(n)

matrix2[0] = [matrix[0][1],matrix[0][0],matrix[0][2]]

total = [0]\*(matrix[0][1])

index = [0]\*(len(total)+1)

index[0] = 1

for i in range(matrix[0][1]):

c = 0;

for j in range(1,n):

if matrix[j][1] == i:

c += 1

total[i] = c

for i in range(1,matrix[0][1]+1):

index[i] = index[i-1]+total[i-1]

for i in range(1,n):

loc = index[matrix[i][1]]

matrix2[loc] = [matrix[i][1],matrix[i][0],matrix[i][2]]

index[matrix[i][1]] += 1

return matrix2

def putData(matrix):

r = int(input("Enter no. of rows: "))

c = int(input("Enter no. of columns: "))

n = int(input("Enter no. of non-zero elements: "))

matrix.append([r,c,n])

print("Now give the input in the sparse matrix:(Row, Column, Element)")

for i in range(n):

list = []

row = int(input())

list.append(row)

col = int(input())

list.append(col)

elm = int(input())

list.append(elm)

matrix.append(list)

def display(matrix):

print("Row Column Element")

for i in matrix:

for j in i:

print("",j,end = " ")

print()

print("Menu:\n1. Simple Transpose\n2. Fast Transpose\n3. Addition of two sparse matrices\n4. Exit")

while(1):

print("Enter choice: ")

ch = int(input())

if ch == 1:

matrix = []

putData(matrix)

matrix2 = simpleTranspose(matrix)

print("Simple Transpose: ")

display(matrix2)

elif ch == 2:

matrix = []

putData(matrix)

matrix2 = fastTranspose(matrix)

print("Fast Transpose: ")

display(matrix2)

elif ch == 3:

matrix1 = []

matrix2 = []

putData(matrix1)

putData(matrix2)

matrix3 = add(matrix1,matrix2)

print("Addition of matrix1 and matrix2: ")

display(matrix3)

elif ch == 4:

print("----EXITTING-------")

exit()

**OUTPUT:**

PS H:\Coding\FDS\expt> & C:/Users/Dell/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Code/assignment10.py

\*\*\*\*\*\*\*\*PROGRAM FOR SPARSE MATRIX OPERATIONS\*\*\*\*\*\*\*\*\*\*

Menu:

1. Simple Transpose

2. Fast Transpose

3. Addition of two sparse matrices

4. Exit

Enter choice:

1

Enter no. of rows: 3

Enter no. of columns: 2

Enter no. of non-zero elements: 4

Now give the input in the sparse matrix:(Row, Column, Element)

0

1

3

1

0

2

1

1

5

2

0

8

Simple Transpose:

Row Column Element

2 3 4

0 1 2

0 2 8

1 0 3

1 1 5

Enter choice:

2

Enter no. of rows: 2

Enter no. of columns: 3

Enter no. of non-zero elements: 4

Now give the input in the sparse matrix:(Row, Column, Element)

0

1

3

0

2

5

1

0

9

1

2

8

Fast Transpose:

Row Column Element

3 2 4

0 1 9

1 0 3

2 0 5

2 1 8

Enter choice:

3

Enter no. of rows: 2

Enter no. of columns: 3

Enter no. of non-zero elements: 4

Now give the input in the sparse matrix:(Row, Column, Element)

0

0

5

0

1

6

1

0

8

1

1

5

Enter no. of rows: 2

Enter no. of columns: 3

Enter no. of non-zero elements: 4

Now give the input in the sparse matrix:(Row, Column, Element)

0

1

4

0

2

3

1

1

3

1

2

6

Addition of matrix1 and matrix2:

Row Column Element

2 3 6

0 0 5

0 1 10

0 2 3

1 0 8

1 1 8

1 2 6

Enter choice:

4

----EXITTING-------

**EXPERIMENT 7**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

struct Friend {

string name;

long long phone;

};

int compare(string a, string b) {

if(a[0] > b[0]) {

return 1;

}

if(a[0] < b[0]) {

return -1;

}

int l1 = a.length();

int l2 = b.length();

int l = min(l1,l2);

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

//cout << a[i] << " " << b[i] << endl;

if(a[i] > b[i]) {

return 1;

}

if(a[i] < b[i]) {

return -1;

}

}

if(l1 == l2) {

return 0;

}

return 0;

}

int binarySearchI(Friend\* arr, int t, string key) {

int low = 0;

int high = t - 1;

while(low <= high) {

int mid = (low + high)/2;

if(arr[mid].name == key) {

return mid;

}

else if(compare(key,arr[mid].name)==1) {

low = mid + 1;

}

else {

high = mid - 1;

}

}

return -1;

}

int binarySearchR(Friend\* arr, int low, int high, string key) {

if(low > high) {

return -1;

}

int mid = (low + high)/2;

int ans;

if(arr[mid].name == key) {

ans = mid;

}

else if(compare(key,arr[mid].name)==1) {

ans = binarySearchR(arr,mid+1,high,key);

}

else {

ans = binarySearchR(arr,low,mid-1,key);

}

return ans;

}

int fibonacciSearch(Friend\* arr, int t, string key) {

int a = 1, b = 0, f = 1;

while(f <= t) {

b = a;

a = f;

f = a + b;

}

int offset = -1;

while(f > 1) {

int i = min(offset + b, t-1);

if(compare(key,arr[i].name)==1) {

f = a;

a = b;

b = f - a;

offset = i;

}

else if(compare(key,arr[i].name) == -1) {

f = b;

a = a - b;

b = f - a;

}

else {

return i;

}

}

return -1;

}

void display(Friend\* arr, int t) {

cout << "Displaying List: \n";

cout << "------------------------------------------------------\n";

cout << " NAME" << " " << "MOBILE NO." << endl;

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

cout << i+1 << ". ";

cout << setw(10) << arr[i].name;

cout << " ";

cout << arr[i].phone << endl;

}

cout << "------------------------------------------------------\n";

}

void addFriend(Friend\* arr, int &t, string key, long long m) {

int ind = t;

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

if(compare(arr[i].name, key)==1) {

ind = i;

break;

}

}

for(int i = t-1; i >= ind; i--) {

arr[i+1] = arr[i];

}

arr[ind].name = key;

arr[ind].phone = m;

t ++;

cout << "Entry added successfully\n";

}

int main() {

Friend arr[100];

cout << "Enter total friends to enter: \n";

int t;

cin >> t;

cout << "Enter the friends in sorted order: \n";

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

cout << "Enter name: ";

cin >> arr[i].name;

cout << "Enter mobile no.: ";

cin >> arr[i].phone;

}

cout << "Menu:\nSearch your friend from list using:\n1. Binary Search iterative\n2. Binary search recursive\n3. Fibonacci Search\n4. Display List\n5. Exit\n\n";

int i; string key;

do {

cout << "Enter choice: ";

int n;

cin >> n;

cin.ignore();

switch(n) {

case 1:

cout << "Enter name of friend to search: ";

getline(cin, key);

i = binarySearchI(arr, t, key);

if(i != -1) {

cout << "Found using Iterative Binary Search at serial no. " << i+1 << ": " << endl;

cout << i+1 << ". ";

cout << setw(13) << "Name : " ;

cout << arr[i].name << endl;

cout << setw(16) << "Mobile No : ";

cout << arr[i].phone << endl;

}

else {

cout << "This friend is not in the list\nPlease enter his mobile no : ";

long long m;

cin >> m;

addFriend(arr,t,key,m);

}

cout << endl;

break;

case 2:

cout << "Enter name of friend to search: ";

getline(cin, key);

i = binarySearchR(arr, 0, t-1, key);

if(i != -1) {

cout << "Found using Recursive Binary Search at serial no. " << i+1 << ": " << endl;

cout << i+1 << ". ";

cout << setw(13) << "Name : " ;

cout << arr[i].name << endl;

cout << setw(16) << "Mobile No : ";

cout << arr[i].phone << endl;

}

else {

cout << "This friend is not in the list\nPlease enter his mobile no : ";

long long m;

cin >> m;

addFriend(arr,t,key,m);

}

cout << endl;

break;

case 3:

cout << "Enter name of friend to search: ";

getline(cin, key);

i = fibonacciSearch(arr, t, key);

if(i != -1) {

cout << "Found using Fibonacci Search at serial no. " << i+1 << ": " << endl;

cout << i+1 << ". ";

cout << setw(13) << "Name : " ;

cout << arr[i].name << endl;

cout << setw(16) << "Mobile No : ";

cout << arr[i].phone << endl;

}

else {

cout << "This friend is not in the list\nPlease enter his mobile no : ";

long long m;

cin >> m;

addFriend(arr,t,key,m);

}

cout << endl;

break;

case 4:

display(arr,t);

cout << endl;

break;

case 5:

cout << "--------------EXITING-----------------\n";

exit(0);

}

}while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-rrtu0fwx.hrg' '--stdout=Microsoft-MIEngine-Out-t15kape1.t2o' '--stderr=Microsoft-MIEngine-Error-ddr2gznh.rtx' '--pid=Microsoft-MIEngine-Pid-0wa5yilp.5id' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

Enter total friends to enter:

3

Enter the friends in sorted order:

Enter name: Avadhesh

Enter mobile no.: 8738834201

Enter name: Ramesh

Enter mobile no.: 9915697843

Enter name: Suresh

Enter mobile no.: 8966567239

Menu:

Search your friend from list using:

1. Binary Search iterative

2. Binary search recursive

3. Fibonacci Search

4. Display List

5. Exit

Enter choice: 4

Displaying List:

------------------------------------------------------

NAME MOBILE NO.

1. Avadhesh 8738834201

2. Ramesh 9915697843

3. Suresh 8966567239

------------------------------------------------------

Enter choice: 1

Enter name of friend to search: Avadhesh

Found using Iterative Binary Search at serial no. 1:

1. Name : Avadhesh

Mobile No : 8738834201

Enter choice: 2

Enter name of friend to search: Ramesh

Found using Recursive Binary Search at serial no. 2:

2. Name : Ramesh

Mobile No : 9915697843

Enter choice: 3

Enter name of friend to search: Suresh

Found using Fibonacci Search at serial no. 3:

3. Name : Suresh

Mobile No : 8966567239

Enter choice: 1

Enter name of friend to search: Dharmesh

This friend is not in the list

Please enter his mobile no : 7856562265

Entry added successfully

Enter choice: 4

Displaying List:

------------------------------------------------------

NAME MOBILE NO.

1. Avadhesh 8738834201

2. Dharmesh 7856562265

3. Ramesh 9915697843

4. Suresh 8966567239

------------------------------------------------------

Enter choice: 5

--------------EXITING-----------------

**EXPERIMENT 8**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

using namespace std::chrono;

void display(float\* arr, int n)

{

    cout << "-----------TOP 5 SCORES :--------------\n";

    cout << "     ";

    for(int i = n-1; i > n-1-5; i--)

    {

        cout << fixed << setprecision(2) << arr[i] << " ";

    }

    cout << endl;

}

long long insertionSort(float \*array, int size) {

    int i, key, j;

    long long cnt = 0;

    for (i = 1; i < size; i++)

    {

    key = array[i];

    j = i - 1;

    while (j >= 0 && array[j] > key)

        {

        cnt ++;

            array[j + 1] = array[j];

            j = j - 1;

        }

        array[j + 1] = key;

    }

    return cnt;

}

long long bubbleSort(float\* arr, int n)

{

    long long cnt = 0;

    for(int i = 0; i < n - 1; i++)

    {

        for(int j = 0; j < n - i - 1; j++)

        {

            cnt ++;

            if(arr[j] > arr[j+1])

                swap(arr[j],arr[j+1]);

        }

    }

    return cnt;

}

long long selectionSort(float \*array, int size) {

   int i, j, imin;

   long long cnt = 0;

   for(i = 0; i<size-1; i++) {

        imin = i;   //get index of minimum data

        for(j = i+1; j<size; j++) {

            cnt ++;

            if(array[j] < array[imin]) {

                imin = j;

            }

        }

        swap(array[i], array[imin]);

    }

   return cnt;

}

int main()

{

    auto start = high\_resolution\_clock::now();

    auto stop = high\_resolution\_clock::now();

    auto duration = duration\_cast<microseconds>(stop - start);

        cout << "\n--------PROGRAM TO SORT FIRST YEAR PERCENTAGE OF STUDENTS-------\n";

        cout << "Menu: \n";

        cout << "1. Selection Sort & Display top five scores\n";

        cout << "2. Bubble Sort & Display top five scores\n";

        cout << "3. Insertion sort & Display top five scores\n";

        cout << "4. Time taken for very large input size\n";

        cout << "5. Exit\n";

    do

    {

        cout << "\nENTER CHOICE - ";

        int ch;

        cin >> ch;

        switch(ch)

        {

            int n;

            float arr[100];

            case 1:

                cout << "Enter no. of students: ";

                cin >> n;

                cout << "Enter the percentages of students: \n";

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

                {

                    cin >> arr[i];

                }

                cout << "--------USING SELECTION SORT----------: \n";

                selectionSort(arr, n);

                display(arr,n);

                break;

            case 2:

                cout << "Enter no. of students: ";

                cin >> n;

                cout << "Enter the percentages of students: \n";

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

                {

                    cin >> arr[i];

                }

                cout << "--------USING BUBBLE SORT----------: \n";

                bubbleSort(arr, n);

                display(arr,n);

                break;

            case 3:

                cout << "Enter no. of students: ";

                cin >> n;

                cout << "Enter the percentages of students: \n";

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

                {

                    cin >> arr[i];

                }

                cout << "--------USING INSERTION SORT----------: \n";

                insertionSort(arr, n);

                display(arr,n);

                break;

            case 4: {

                cout << "\n-------TIME TAKEN FOR LARGE INPUT SIZE---------\n";

                cout << "\n------------BUBBLE SORT------------------\n";

                int num = 10000;

                float \*Arr = new float[num];

                long long a;

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = bubbleSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------10,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                num = 100000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = bubbleSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------1,00,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                num = 1000000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = bubbleSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------10,00,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                cout << "\n------------SELECTION SORT------------------\n";

                num = 10000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = selectionSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------10,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                num = 100000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = selectionSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------1,00,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                num = 1000000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = selectionSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------10,00,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                cout << "\n------------INSERTION SORT------------------\n";

                num = 10000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = insertionSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------10,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                num = 100000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = insertionSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------1,00,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                num = 1000000;

                Arr = new float[num];

                for(int i = 0; i < num; i++)

                {

                    Arr[i] = rand();

                }

                start = high\_resolution\_clock::now();

                a = insertionSort(Arr,num);

                stop = high\_resolution\_clock::now();

                cout << "------10,00,000 numbers------\n";

                duration = duration\_cast<microseconds>(stop - start);

                cout << "No. of comparisons: " << a << endl;

                cout << fixed << "Time Taken: " << setprecision(2) << duration.count() / 1000000.00 << " seconds" << endl;

                cout << endl;

                break;

            }

            case 5:

                cout << "\n------------EXITING-------------\n";

                exit(0);

        }

    }

    while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-hgqlf3i1.zsa' '--stdout=Microsoft-MIEngine-Out-ls5n3r1b.cd4' '--stderr=Microsoft-MIEngine-Error-wjy0en1v.m3d' '--pid=Microsoft-MIEngine-Pid-q0k5wcp1.jca' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

--------PROGRAM TO SORT FIRST YEAR PERCENTAGE OF STUDENTS-------

Menu:

1. Selection Sort & Display top five scores

2. Bubble Sort & Display top five scores

3. Insertion sort & Display top five scores

4. Time taken for very large input size

5. Exit

ENTER CHOICE - 1

Enter no. of students: 10

Enter the percentages of students:

32.56 49.70 99.11 77.65 89.00 91.56 78.64 85.43 97.61 90.50

--------USING SELECTION SORT----------:

-----------TOP 5 SCORES :--------------

99.11 97.61 91.56 90.50 89.00

ENTER CHOICE - 2

Enter no. of students: 10

Enter the percentages of students:

98.10 95.43 97.01 99.95 88.60 56.66 67.21 69.67 71.25 88.73

--------USING BUBBLE SORT----------:

-----------TOP 5 SCORES :--------------

99.95 98.10 97.01 95.43 88.73

ENTER CHOICE - 3

Enter no. of students: 10

Enter the percentages of students:

43.44 94.56 96.11 92.56 65.66 67.65 99.10 98.10 81.68 82.35

--------USING INSERTION SORT----------:

-----------TOP 5 SCORES :--------------

99.00 98.00 96.00 94.00 92.00

ENTER CHOICE - 4

-------TIME TAKEN FOR LARGE INPUT SIZE---------

------------BUBBLE SORT------------------

------10,000 numbers------

No. of comparisons: 49995000

Time Taken: 0.72 seconds

------1,00,000 numbers------

No. of comparisons: 4999950000

Time Taken: 64.63 seconds

------10,00,000 numbers------

No. of comparisons: 499999500000

Time Taken: 6572.87 seconds

------------SELECTION SORT------------------

------10,000 numbers------

No. of comparisons: 49995000

Time Taken: 0.16 seconds

------1,00,000 numbers------

No. of comparisons: 4999950000

Time Taken: 16.24 seconds

------10,00,000 numbers------

No. of comparisons: 499999500000

Time Taken: 1648.36 seconds

------------INSERTION SORT------------------

------10,000 numbers------

No. of comparisons: 24640804

Time Taken: 0.10 seconds

No. of comparisons: 2494601951

Time Taken: 9.79 seconds

No. of comparisons: 2,52,55,01,55,990

Time Taken: 958.441 seconds

ENTER CHOICE - 5

------------EXITING-------------

**EXPERIMENT 9**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

using namespace std::chrono;

void display(float\* arr, int n)

{

cout << "-----------TOP 5 SCORES :--------------\n";

cout << " ";

for(int i = n-1; i > n-1-5; i--)

{

cout << fixed << setprecision(2) << arr[i] << " ";

}

cout << endl;

}

long long shellSort(float arr[], int n)

{

long long cnt = 0;

for (int gap = n/2; gap > 0; gap /= 2)

{

for (int i = gap; i < n; i += 1)

{

float temp = arr[i];

int j;

for (j = i; j >= gap && arr[j - gap] > temp; j -= gap) {

cnt ++;

arr[j] = arr[j - gap];

}

arr[j] = temp;

}

}

return cnt;

}

long long quickSort(float\* arr, int first, int last)

{

if(first > last)

{

return 1;

}

int p = first;

int i = first;

int j = last;

long long cnt = 0;

while(i < j)

{

while(arr[i] <= arr[p])

{

cnt ++;

if(i == last)

{

break;

}

i ++;

}

while(arr[j] >= arr[p])

{

cnt++;

if(j == first)

{

break;

}

j --;

}

if(i < j)

{

swap(arr[i],arr[j]);

}

}

swap(arr[p],arr[j]);

long long c1 = quickSort(arr, first , j - 1);

long long c2 = quickSort(arr, j + 1, last);

return cnt + c1 + c2;

}

int main()

{

auto start = high\_resolution\_clock::now();

auto stop = high\_resolution\_clock::now();

auto duration = duration\_cast<microseconds>(stop - start);

cout << "\n--------PROGRAM TO SORT FIRST YEAR PERCENTAGE OF STUDENTS-------\n";

cout << "Menu: \n";

cout << "1. Quick Sort & Display top five scores\n";

cout << "2. Shell Sort & Display top five scores\n";

cout << "3. Time taken for very large input size\n";

cout << "4. Exit\n";

do

{

cout << "\nENTER CHOICE - ";

int ch;

cin >> ch;

switch(ch)

{

int n;

float arr[100];

case 1:

cout << "Enter no. of students: ";

cin >> n;

cout << "Enter the percentages of students: \n";

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

{

cin >> arr[i];

}

cout << "--------USING QUICK SORT----------: \n";

quickSort(arr,0,n-1);

display(arr,n);

break;

case 2:

//int arr[100];

cout << "Enter no. of students: ";

cin >> n;

cout << "Enter the percentages of students: \n";

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

{

cin >> arr[i];

}

cout << "--------USING SHELL SORT----------: \n";

shellSort(arr, n);

display(arr,n);

break;

case 3: {

cout << "\n-------TIME TAKEN FOR LARGE INPUT SIZE---------\n";

cout << "\n------------QUICK SORT------------------\n";

int num = 10000;

float \*Arr = new float[num];

long long a;

for(int i = 0; i < num; i++)

{

Arr[i] = rand();

}

start = high\_resolution\_clock::now();

a = quickSort(Arr,0,num-1);

stop = high\_resolution\_clock::now();

cout << "------10,000 numbers------\n";

duration = duration\_cast<microseconds>(stop - start);

cout << "No. of comparisons: " << a << endl;

cout << fixed << "Time Taken: " << setprecision(4) << duration.count() / 1000000.00 << " seconds" << endl;

cout << endl;

num = 100000;

Arr = new float[num];

for(int i = 0; i < num; i++)

{

Arr[i] = rand();

}

start = high\_resolution\_clock::now();

a = quickSort(Arr,0,num-1);

stop = high\_resolution\_clock::now();

cout << "------1,00,000 numbers------\n";

duration = duration\_cast<microseconds>(stop - start);

cout << "No. of comparisons: " << a << endl;

cout << fixed << "Time Taken: " << setprecision(4) << duration.count() / 1000000.00 << " seconds" << endl;

cout << endl;

num = 1000000;

Arr = new float[num];

for(int i = 0; i < num; i++)

{

Arr[i] = rand();

}

start = high\_resolution\_clock::now();

a = quickSort(Arr,0,num-1);

stop = high\_resolution\_clock::now();

cout << "------10,00,000 numbers------\n";

duration = duration\_cast<microseconds>(stop - start);

cout << "No. of comparisons: " << a << endl;

cout << fixed << "Time Taken: " << setprecision(4) << duration.count() / 1000000.00 << " seconds" << endl;

cout << endl;

cout << "\n------------SHELL SORT------------------\n";

num = 10000;

Arr = new float[num];

for(int i = 0; i < num; i++)

{

Arr[i] = rand();

}

start = high\_resolution\_clock::now();

a = shellSort(Arr,num);

stop = high\_resolution\_clock::now();

cout << "------10,000 numbers------\n";

duration = duration\_cast<microseconds>(stop - start);

cout << "No. of comparisons: " << a << endl;

cout << fixed << "Time Taken: " << setprecision(4) << duration.count() / 1000000.00 << " seconds" << endl;

cout << endl;

num = 100000;

Arr = new float[num];

for(int i = 0; i < num; i++)

{

Arr[i] = rand();

}

start = high\_resolution\_clock::now();

a = shellSort(Arr,num);

stop = high\_resolution\_clock::now();

cout << "------1,00,000 numbers------\n";

duration = duration\_cast<microseconds>(stop - start);

cout << "No. of comparisons: " << a << endl;

cout << fixed << "Time Taken: " << setprecision(4) << duration.count() / 1000000.00 << " seconds" << endl;

cout << endl;

num = 1000000;

Arr = new float[num];

for(int i = 0; i < num; i++)

{

Arr[i] = rand();

}

start = high\_resolution\_clock::now();

a = shellSort(Arr,num);

stop = high\_resolution\_clock::now();

cout << "------10,00,000 numbers------\n";

duration = duration\_cast<microseconds>(stop - start);

cout << "No. of comparisons: " << a << endl;

cout << fixed << "Time Taken: " << setprecision(4) << duration.count() / 1000000.00 << " seconds" << endl;

cout << endl;

break;

}

case 4:

cout << "\n------------EXITING-------------\n";

exit(0);

}

}

while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-z3t21vrn.kwj' '--stdout=Microsoft-MIEngine-Out-haxrjauq.jrk' '--stderr=Microsoft-MIEngine-Error-szl01gnq.b1f' '--pid=Microsoft-MIEngine-Pid-k3rsmm5u.l0d' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

--------PROGRAM TO SORT FIRST YEAR PERCENTAGE OF STUDENTS-------

Menu:

1. Quick Sort & Display top five scores

2. Shell Sort & Display top five scores

3. Time taken for very large input size

4. Exit

ENTER CHOICE - 1

Enter no. of students: 10

Enter the percentages of students:

32.56 49.70 99.11 77.65 89.00 91.56 78.64 85.43 97.61 90.50

--------USING QUICK SORT----------:

-----------TOP 5 SCORES :--------------

99.11 97.61 91.56 90.50 89.00

ENTER CHOICE - 2

Enter no. of students: 10

Enter the percentages of students:

98.10 95.43 97.01 99.95 88.60 56.66 67.21 69.67 71.25 88.73

--------USING SHELL SORT----------:

-----------TOP 5 SCORES :--------------

99.95 98.10 97.01 95.43 88.73

ENTER CHOICE - 3

-------TIME TAKEN FOR LARGE INPUT SIZE---------

------------QUICK SORT------------------

------10,000 numbers------

No. of comparisons: 192922

Time Taken: 0.0032 seconds

------1,00,000 numbers------

No. of comparisons: 2223125

Time Taken: 0.0335 seconds

------10,00,000 numbers------

No. of comparisons: 38473465

Time Taken: 0.3169 seconds

------------SHELL SORT------------------

------10,000 numbers------

No. of comparisons: 149783

Time Taken: 0.0031 seconds

------1,00,000 numbers------

No. of comparisons: 2844840

Time Taken: 0.0501 seconds

------10,00,000 numbers------

No. of comparisons: 47264436

Time Taken: 0.5511 seconds

ENTER CHOICE - 4

------------EXITING-------------

**EXPERIMENT 10**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

struct Node {

    string PRN;

    int roll;

    string name;

    Node \*next;

};

class LinkedList {

    int t;

    Node \*pr;

    Node \*sr;

    public:

    LinkedList() {

        pr = new Node;

        sr = new Node;

        pr -> PRN = "-";

        pr -> next = sr;

        sr -> PRN = "-";

        sr -> next = NULL;

        t = 0;

    }

    void addP() {

        string PRN; string name;

        cout << "Enter PRN: ";

        cin.ignore();

        getline(cin, PRN);

        cout << "Enter name: ";

        getline(cin, name);

        pr -> PRN = PRN;

        pr -> name = name;

        t ++;

    }

    void addS() {

        string PRN; string name;

        cout << "Enter PRN: ";

        cin.ignore();

        getline(cin, PRN);

        cout << "Enter Name: ";

        getline(cin, name);

        sr -> PRN = PRN;

        sr -> name = name;

        t ++;

    }

    void addM() {

        string PRN; string name;

        cout << "Enter PRN: ";

        cin.ignore();

        getline(cin, PRN);

        cout << "Enter Name: ";

        getline(cin, name);

        Node \*ptr = new Node;

        ptr -> PRN = PRN;

        ptr -> name = name;

        Node \*tmp = pr;

        while(tmp -> next != sr) {

            tmp = tmp -> next;

        }

        tmp -> next = ptr;

        ptr -> next = sr;

        t ++;

    }

    void deleteM() {

        cout << "Enter PRN of member to remove: ";

        string PRN;

        cin.ignore();

        getline(cin, PRN);

        Node \*tmp = pr;

        while(tmp -> next != sr) {

            if(tmp -> next ->PRN == PRN) {

                break;

            }

            tmp = tmp -> next;

        }

        cout << "MEMBER " << tmp -> next -> name << " REMOVED\n";

        tmp -> next = tmp -> next -> next;

        t --;

    }

    void deleteP() {

        pr -> PRN = -1;

        t --;

    }

    void deleteS() {

        sr -> PRN = -1;

        t --;

    }

    void display() {

        int cnt = 1;

        cout << "\n------------------DISPLAYING PINNACLE CLUB----------------\n";

        cout << "--------------PRESIDENT----------------\n";

        if(pr -> PRN == "-") {

            cout << "NOT APPOINTED\n";

        }

        else {

            cout << cnt << ". " << "PRN: " << pr -> PRN << endl;

            cout << "   " "NAME: " << pr -> name << endl;

            cnt ++;

        }

        cout << "---------------------------------------\n";

        Node \*tmp = pr -> next;

        while(tmp != sr) {

            cout << cnt << ". " << "PRN: " << tmp -> PRN << endl;

            cout << "   " << "NAME: " << tmp -> name << endl;

            cout << "-----------------------------------------\n";

            tmp = tmp -> next;

            cnt ++;

        }

        cout << "--------------SECRETARY----------------\n";

        if(sr -> PRN == "-") {

            cout << "NOT APPOINTED\n";

        }

        else {

            cout << cnt << ". " << "PRN: " << sr -> PRN << endl;

            cout << "   " "NAME: " << sr -> name << endl;

            cnt ++;

        }

        cout << "---------------------------------------\n";

        cout << "TOTAL MEMBERS: " << t << endl;

        cout << "---------------------------------------\n";

    }

    bool isEmpty() {

        if(pr -> next == sr) {

            return 1;

        }

        return 0;

    }

    void displayReverse() {

        cout << "\n--------------DISPLAYING LIST IN REVERSE ORDER USING RECURSION--------------\n";

        int cnt = 1;

        rec(pr,cnt);

        cout << "\n";

    }

    int rec(Node \*ptr, int cnt) {

        if(ptr == sr) {

            cout << "--------------SECRETARY----------------\n";

            if(sr -> PRN == "-") {

                cout << "NOT APPOINTED\n";

            }

            else {

                cout << cnt << ". " << "PRN: " << sr -> PRN << endl;

                cout << "   " "NAME: " << sr -> name << endl;

                cnt ++;

            }

            cout << "---------------------------------------\n";

            return cnt;

        }

        int c = rec(ptr -> next,cnt);

        if(ptr != pr) {

            cout << c << ". " << "PRN: " << ptr -> PRN << endl;

            cout << "   " << "NAME: " << ptr -> name << endl;

            cout << "-----------------------------------------\n";

            ptr = ptr -> next;

            c ++;

        }

        else {

            cout << "--------------PRESIDENT----------------\n";

            if(pr -> PRN == "-") {

                cout << "NOT APPOINTED\n";

            }

            else {

                cout << c << ". " << "PRN: " << pr -> PRN << endl;

                cout << "   " "NAME: " << pr -> name << endl;

                c ++;

            }

            cout << "---------------------------------------\n";

            cout << "TOTAL MEMBERS: " << t << endl;

            cout << "---------------------------------------\n";

        }

        return c;

    }

    int total() {

        return t;

    }

};

int main() {

    cout << "----------WELCOME TO PINNACLE CLUB---------------\n";

    LinkedList l;

    cout << "MENU: \n";

    cout << "1. Add President\n2. Add Secretary\n3. Add member\n4. Delete President / Secretary / member\n5. Compute total no. of members\n6. Display members\n7. Display List in reverse order\n8. Concatenate two LL\n9. Exit\n";

    do {

        cout << "Enter no. of operation: ";

        int n;

        cin >> n;

        switch(n) {

            case 1:

                l.addP();

                break;

            case 2:

                l.addS();

                break;

            case 3:

                l.addM();

                break;

            case 4:

                cout << "1. President\n2. Secretary\n3. Member\n";

                cout << "Who you want to delete: ";

                int ch;

                cin >> ch;

                switch(ch) {

                    case 1:

                        l.deleteP();

                        break;

                    case 2:

                        l.deleteS();

                        break;

                    case 3:

                        l.deleteM();

                        break;

                    default: break;

                }

                break;

            case 5:

                cout << "TOTAL MEMBERS: " << l.total() << endl;

                break;

            case 6:

                l.display();

                break;

            case 7:

                l.displayReverse();

                break;

            case 8: {

                cout << "\n--------------------------CONCATENATING TWO LINKED LISTS---------------------------\n";

                cout << "\nWe will take input of list Roll No.s of students of Divison A and Divsion B and then concatenate the two lists\n\n";

                cout << "Enter Roll No.s of Div. A students:(Enter -1 to terminate) \n";

                int x = 0;

                Node \*head1 = NULL, \*head2 = NULL;

                Node \*temp;

                while(x != -1) {

                    cin >> x;

                    if(x == -1) {

                        break;

                    }

                    temp = head1;

                    Node \*ptr = new Node;

                    ptr -> roll = x;

                    ptr -> next = NULL;

                    if(head1 == NULL) {

                        head1 = ptr;

                    }

                    else {

                        while(temp -> next != NULL) {

                            temp = temp -> next;

                        }

                        temp -> next = ptr;

                    }

                }

                cout << "\nEnter Roll No.s of Div. B students:(Enter -1 to terminate) \n";

                x = 0;

                while(x != -1) {

                    cin >> x;

                    if(x == -1) {

                        break;

                    }

                    temp = head2;

                    Node \*ptr = new Node;

                    ptr -> roll = x;

                    ptr -> next = NULL;

                    if(head2 == NULL) {

                        head2 = ptr;

                    }

                    else {

                        while(temp -> next != NULL) {

                            temp = temp -> next;

                        }

                        temp -> next = ptr;

                    }

                }

                Node \*tmp = head1;

                cout << "\nDIVISION A: ";

                while(tmp -> next != NULL) {

                    cout << tmp -> roll << " -> ";

                    tmp = tmp -> next;

                }

                cout << tmp -> roll << endl;

                tmp = head2;

                cout << "DIVISION B: ";

                while(tmp -> next != NULL) {

                    cout << tmp -> roll << " -> ";

                    tmp = tmp -> next;

                }

                cout << tmp -> roll << endl;

                tmp = head1;

                while(tmp -> next != NULL) {

                    tmp = tmp -> next;

                }

                tmp -> next = head2;

                cout << "\nCONCATENATED: ";

                tmp = head1;

                while(tmp -> next != NULL) {

                    cout << tmp -> roll << " -> ";

                    tmp = tmp -> next;

                }

                cout << tmp -> roll << endl;

                cout << "------------------------------------------------------------------------\n";

                break;

            }

            case 9:

                cout << "-------------EXITING--------------\n";

                exit(0);

        }

    }

    while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-asy1zccm.xlc' '--stdout=Microsoft-MIEngine-Out-hejydfkb.5vw' '--stderr=Microsoft-MIEngine-Error-tunsu2ao.c0z' '--pid=Microsoft-MIEngine-Pid-its4yhvy.l4s' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

----------WELCOME TO PINNACLE CLUB---------------

MENU:

1. Add President

2. Add Secretary

3. Add member

4. Delete President / Secretary / member

5. Compute total no. of members

6. Display members

7. Display List in reverse order

8. Concatenate two LL

9. Exit

Enter no. of operation: 1

Enter PRN: 72285632G

Enter name: YASWANT SINGH

Enter no. of operation: 2

Enter PRN: 72285659J

Enter Name: AJAY SINGH

Enter no. of operation: 3

Enter PRN: 72285697M

Enter Name: ANMOL SINGH

Enter no. of operation: 3

Enter PRN: 72285802H

Enter Name: KEVIN VARGHESE

Enter no. of operation: 3

Enter PRN: 72285816H

Enter Name: LOKENDRA SINGH

Enter no. of operation: 3

Enter PRN: 72285854L

Enter Name: PANKAJ YADAV

Enter no. of operation: 3

Enter PRN: 72285925C

Enter Name: SAURABH KUMAR

Enter no. of operation: 3

Enter PRN: 72285997L

Enter Name: YASHVIR YADAV

Enter no. of operation: 3

Enter PRN: 72285989K

Enter Name: VIKRAM RENIWAL

Enter no. of operation: 6

------------------DISPLAYING PINNACLE CLUB----------------

--------------PRESIDENT----------------

1. PRN: 72285632G

NAME: YASWANT SINGH

---------------------------------------

2. PRN: 72285697M

NAME: ANMOL SINGH

-----------------------------------------

3. PRN: 72285802H

NAME: KEVIN VARGHESE

-----------------------------------------

4. PRN: 72285816H

NAME: LOKENDRA SINGH

-----------------------------------------

5. PRN: 72285854L

NAME: PANKAJ YADAV

-----------------------------------------

6. PRN: 72285925C

NAME: SAURABH KUMAR

-----------------------------------------

7. PRN: 72285997L

NAME: YASHVIR YADAV

-----------------------------------------

8. PRN: 72285989K

NAME: VIKRAM RENIWAL

-----------------------------------------

--------------SECRETARY----------------

9. PRN: 72285659J

NAME: AJAY SINGH

---------------------------------------

TOTAL MEMBERS: 9

---------------------------------------

Enter no. of operation: 7

--------------DISPLAYING LIST IN REVERSE ORDER USING RECURSION--------------

--------------SECRETARY----------------

1. PRN: 72285659J

NAME: AJAY SINGH

---------------------------------------

2. PRN: 72285989K

NAME: VIKRAM RENIWAL

-----------------------------------------

3. PRN: 72285997L

NAME: YASHVIR YADAV

-----------------------------------------

4. PRN: 72285925C

NAME: SAURABH KUMAR

-----------------------------------------

5. PRN: 72285854L

NAME: PANKAJ YADAV

-----------------------------------------

6. PRN: 72285816H

NAME: LOKENDRA SINGH

-----------------------------------------

7. PRN: 72285802H

NAME: KEVIN VARGHESE

-----------------------------------------

8. PRN: 72285697M

NAME: ANMOL SINGH

-----------------------------------------

--------------PRESIDENT----------------

9. PRN: 72285632G

NAME: YASWANT SINGH

---------------------------------------

TOTAL MEMBERS: 9

---------------------------------------

Enter no. of operation: 4

1. President

2. Secretary

3. Member

Who you want to delete: 3

Enter PRN of member to remove: 72285854L

MEMBER PANKAJ YADAV REMOVED

Enter no. of operation: 6

------------------DISPLAYING PINNACLE CLUB----------------

--------------PRESIDENT----------------

1. PRN: 72285632G

NAME: YASWANT SINGH

---------------------------------------

2. PRN: 72285697M

NAME: ANMOL SINGH

-----------------------------------------

3. PRN: 72285802H

NAME: KEVIN VARGHESE

-----------------------------------------

4. PRN: 72285816H

NAME: LOKENDRA SINGH

-----------------------------------------

5. PRN: 72285925C

NAME: SAURABH KUMAR

-----------------------------------------

6. PRN: 72285997L

NAME: YASHVIR YADAV

-----------------------------------------

7. PRN: 72285989K

NAME: VIKRAM RENIWAL

-----------------------------------------

--------------SECRETARY----------------

8. PRN: 72285659J

NAME: AJAY SINGH

---------------------------------------

TOTAL MEMBERS: 8

---------------------------------------

Enter no. of operation: 8

--------------------------CONCATENATING TWO LINKED LISTS---------------------------

We will take input of list Roll No.s of students of Divison A and Divsion B and then concatenate the two lists

Enter Roll No.s of Div. A students:(Enter -1 to terminate)

3201 3210 3212 3214 3229 3237 3252 3257 -1

Enter Roll No.s of Div. B students:(Enter -1 to terminate)

7202 7204 7232 7249 7256 -1

DIVISION A: 3201 -> 3210 -> 3212 -> 3214 -> 3229 -> 3237 -> 3252 -> 3257

DIVISION B: 7202 -> 7204 -> 7232 -> 7249 -> 7256

CONCATENATED: 3201 -> 3210 -> 3212 -> 3214 -> 3229 -> 3237 -> 3252 -> 3257 -> 7202 -> 7204 -> 7232 -> 7249 -> 7256

------------------------------------------------------------------------

Enter no. of operation: 9

-------------EXITING--------------

**EXPERIMENT 11**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

class binary{

    struct node{

        char d;

        node \*next;

        node \*prev;

    };

    int t;

    node \*head;

    public:

    binary() {

        t = 0;

        head = NULL;

    }

    node \*getlast() {

        node \*tmp = head;

        while(tmp -> next != NULL) {

            tmp = tmp -> next;

        }

        return tmp;

    }

    node \*getHead() {

        return head;

    }

    int len() {

        return t;

    }

    void input() {

        string s;

        cin >> s;

        for(auto i:s) {

            node \*ptr = new node;

            ptr -> d = i;

            ptr -> next = NULL;

            ptr -> prev = NULL;

            if(head == NULL) {

                head = ptr;

                t ++;

                continue;

            }

            node \*tmp = head;

            while(tmp -> next != NULL) {

                tmp = tmp -> next;

            }

            tmp -> next = ptr; ptr -> prev = tmp;

            t ++;

        }

    }

    string onesComplement() {

        string t = "";

        node \*tmp = head;

        while(tmp != NULL) {

            if(tmp -> d == '0') {

                tmp -> d = '1';

            }

            else {

                tmp -> d = '0';

            }

            t += tmp -> d;

            tmp = tmp -> next;

        }

        return t;

    }

    string twosComplement() {

        string t = "";

        node \*tmp = head;

        while(tmp -> next != NULL) {

            if(tmp -> d == '0') {

                tmp -> d = '1';

            }

            else {

                tmp -> d = '0';

            }

            tmp = tmp -> next;

        }

        bool c = 0;

        if(tmp -> d == '0') {

            t += '0';

            c = 1;

        }

        if(tmp -> d == '1') {

            t += '1';

        }

        tmp = tmp -> prev;

        while(tmp != NULL) {

            if(c and tmp -> d == '0') {

                tmp -> d = '1';

                c = 0;

            }

            else if(c and tmp -> d == '1') {

                tmp -> d = '0';

            }

            t += tmp -> d;

            tmp = tmp -> prev;

        }

        if(c) {

            t += '1';

        }

        reverse(t.begin(),t.end());

        return t;

    }

    string add(binary b) {

        string t = "";

        int n1 = this -> len();

        int n2 = b.len();

        if(n1 > n2) {

            for(int i = 0; i < n1 - n2; i++) {

                node \*ptr = new node;

                ptr -> d = '0';

                ptr -> next = NULL;

                ptr -> prev = NULL;

                if(head == NULL) {

                    head = ptr;

                    continue;

                }

                ptr -> next = head;

            }

        }

        if(n2 > n1) {

            for(int i = 0; i < n2 - n1; i++) {

                node \*ptr = new node;

                ptr -> d = '0';

                ptr -> next = NULL;

                ptr -> prev = NULL;

                if(b.getHead() == NULL) {

                    head = ptr;

                    continue;

                }

                ptr -> next = b.getHead();

            }

        }

        bool c = 0;

        node \*l1 = this -> getlast();

        node \*l2 = b.getlast();

        while(l1 != NULL) {

            if(l1 -> d == '0' and l2 -> d == '0') {

                if(!c) {

                    t += '0';

                }

                else {

                    t += '1';

                    c = 0;

                }

            }

            else if(l1 -> d == '1' and l2 -> d == '0' or l1 -> d == '0' and l2 -> d == '1') {

                if(c) {

                    t += '0';

                }

                else {

                    t += '1';

                }

            }

            else {

                if(c) {

                    t += '1';

                }

                else {

                    t += '0';

                    c = 1;

                }

            }

            l1 = l1 -> prev;

            l2 = l2 -> prev;

        }

        if(c) {

            t += '1';

        }

        reverse(t.begin(),t.end());

        return t;

    }

};

int main() {

    cout << "\n------------PROGRAM FOR OPERATION ON BINARY NUMBERS USING DOUBLY LINKED LIST-------------\n";

    cout << "MENU: \n1. 1's Complement\n2. 2's Complement\n3. Addition of two binary numbers\n4. Exit\n";

    do {

        cout << "\nEnter choice: ";

        int n;

        cin >> n;

        switch(n) {

            case 1: {

                cout << "Enter the binary number: ";

                binary b;

                b.input();

                cout << "1's COMPLEMENT: ";

                cout << b.onesComplement();

                cout << endl;

                break;

            }

            case 2: {

                cout << "Enter the binary number: ";

                binary b;

                b.input();

                cout << "2's COMPLEMENT: ";

                cout << b.twosComplement();

                cout << endl;

                break;

            }

            case 3: {

                binary b1, b2;

                cout << "Enter first binary number: ";

                b1.input();

                cout << "Enter second binary number: ";

                b2.input();

                cout << "SUM: ";

                cout << b1.add(b2) << endl;

                break;

            }

            case 4:

                cout << "\n---------EXITING---------\n";

                exit(0);

        }

    }

    while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-2qcjqzxv.n0y' '--stdout=Microsoft-MIEngine-Out-hlutxtfw.u4m' '--stderr=Microsoft-MIEngine-Error-hfhrty0q.p14' '--pid=Microsoft-MIEngine-Pid-0kh0evgv.xu3' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

------------PROGRAM FOR OPERATION ON BINARY NUMBERS USING DOUBLY LINKED LIST-------------

MENU:

1. 1's Complement

2. 2's Complement

3. Addition of two binary numbers

4. Exit

Enter choice: 1

Enter the binary number: 10011010

1's COMPLEMENT: 01100101

Enter choice: 2

Enter the binary number: 10101011

2's COMPLEMENT: 01010101

Enter choice: 3

Enter first binary number: 101100

Enter second binary number: 100111

SUM: 1010011

Enter choice: 4

---------EXITING---------

**EXPERIMENT 12**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

#define size 100

template<class T>

class Stack{

    T arr[100];

    public:

    int t;

    Stack() {

        t = -1;

    }

    void push(T x) {

        if(t == size-1) {

            cout << "Stack Overflow\n";

        }

        else {

            t ++;

            arr[t] = x;

        }

    }

    int pop() {

        if(t < 0) {

            //cout << "Stack Underflow\n";

            return -1;

        }

        T x = arr[t];

        t --;

        return x;

    }

    void display() {

        for(int i = 0; i <= t; i++) {

            cout << arr[i] << " ";

        }

        cout << endl;

    }

    bool isEmpty() {

        if(t < 0) {

            return 1;

        }

        return 0;

    }

};

bool checkParentheses(string s) {

    Stack<char> stk;

    for(auto i:s) {

        if(i == '(' or i == '[' or i == '{') {

            stk.push(i);

        }

        if(i == ')') {

            int x = stk.pop();

            if(x == -1 or x != '(') {

                return 0;

            }

        }

        else if(i == ']') {

            int x = stk.pop();

            if(x == -1 or x != '[') {

                return 0;

            }

        }

        else if(i == '}') {

            int x = stk.pop();

            if(x == -1 or x != '{') {

                return 0;

            }

        }

    }

    if(!stk.isEmpty()) {

        return 0;

    }

    return 1;

}

int main() {

    cout << "\n--------------PROGRAM TO CHECK BALANCED PARENTHESES-----------------\n";

    cout << "MENU: \n1. Check Balanced Parentheses\n2. Exit";

    do {

        cout << "\nEnter choice: ";

        int n;

        cin >> n;

        switch(n) {

            case 1: {

                cout << "Enter the expression: ";

                string s;

                cin >> s;

                if(checkParentheses(s)) {

                    cout << "BALANCED\n";

                }

                else {

                    cout << "NOT BALANCED\n";

                }

                break;

            }

            case 2:

                cout << "\n----------------EXITING---------------\n";

                exit(0);

        }

    }

    while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-tkeocchw.gnv' '--stdout=Microsoft-MIEngine-Out-z5wr1epw.wge' '--stderr=Microsoft-MIEngine-Error-racbtba4.dwz' '--pid=Microsoft-MIEngine-Pid-au2l22wa.rlj' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

--------------PROGRAM TO CHECK BALANCED PARENTHESES-----------------

MENU:

1. Check Balanced Parentheses

2. Exit

Enter choice: 1

Enter the expression: {[A+B/(D-E)]}

BALANCED

Enter choice: 1

Enter the expression: {[()]}

BALANCED

Enter choice: 1

Enter the expression: }{

NOT BALANCED

Enter choice: 1

Enter the expression: {[}

NOT BALANCED

Enter choice: 1

Enter the expression: {[)()]}

NOT BALANCED

Enter choice:

2

----------------EXITING---------------

**EXPERIMENT 13**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

#define size 100

template<class T>

class Stack{

    T arr[size];

    public:

    int t;

    Stack() {

        t = -1;

    }

    T top() {

        return arr[t];

    }

    void push(T x) {

        if(t == size-1) {

            cout << "Stack Overflow\n";

        }

        else {

            t ++;

            arr[t] = x;

        }

    }

    T pop() {

        if(t < 0) {

            cout << "Stack Underflow\n";

            return -1;

        }

        T x = arr[t];

        t --;

        return x;

    }

    void display() {

        for(int i = 0; i <= t; i++) {

            cout << arr[i] << " ";

        }

        cout << endl;

    }

    bool isEmpty() {

        if(t < 0) {

            return 1;

        }

        return 0;

    }

};

bool op(char x) {

    if(x == '+' or x == '-' or x == '\*' or x == '/') {

        return 1;

    }

    return 0;

}

void infixToPostfix(string s, vector<char> &ans) {

    map<char, int> map1;

    map1['+'] = 1;

    map1['-'] = 1;

    map1['\*'] = 2;

    map1['/'] = 2;

    Stack<char> stk;

    bool f = 0;

    for(auto i:s) {

        if(!op(i)) {

            if(i == '(') {

                stk.push(i);

            }

            else if(i == ')') {

                while(!stk.isEmpty()) {

                    if(stk.top() != '(') {

                        ans.push\_back(stk.top());

                    }

                    stk.pop();

                }

            }

            else {

                ans.push\_back(i);

            }

        }

        else {

            if(map1[stk.top()] > map1[i]) {

                ans.push\_back(stk.pop());

            }

            stk.push(i);

        }

    }

    while(!stk.isEmpty()) {

        if(stk.top() != '(') {

            ans.push\_back(stk.top());

        }

        stk.pop();

    }

}

float postfixEvaluate(string s) {

    Stack<float> stk;

    for(auto i:s) {

        if(!op(i)) {

            stk.push(i-48);

        }

        else if(op(i)) {

            float y = stk.pop();

            float x = stk.pop();

            float z;

            if(i == '+') {

                z = x + y;

            }

            else if(i == '-') {

                z = x - y;

            }

            else if(i == '\*') {

                z = x \* y;

            }

            else if(i == '/') {

                z = x / y;

            }

            stk.push(z);

        }

    }

    return stk.pop();

}

int main() {

    cout << "\n-----------PROGRAM FOR INFIX TO POSTFIX CONVERSION AND EVALUATION----------------\n";

    cout << "\nMENU:\n1. Infix to Postfix\n2. Postfix Evaluation\n3. Exit\n";

    do {

        cout << "\nEnter choice: ";

        int n;

        cin >> n;

        switch(n) {

            case 1: {

                cout << "\n--------------INFIX -> POSTFIX-----------------\n";

                vector<char> ans;

                string s;

                cout << "Enter the infix expression: ";

                cin >> s;

                infixToPostfix(s, ans);

                cout << "Postfix expression: ";

                for(auto i:ans) {

                    cout << i;

                }

                cout << endl;

                break;

            }

            case 2: {

                cout << "\n-----------------POSTFIX EVALUATION------------------\n";

                cout << "Enter the postfix expression: ";

                string s;

                cin >> s;

                cout << "Value of the given PostFix expression: " << fixed << setprecision(2) << postfixEvaluate(s) << endl;

                break;

            }

            case 3:

                cout << "\n----------------EXITING---------------------\n";

                exit(0);

        }

    }

    while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-out4izh0.5oe' '--stdout=Microsoft-MIEngine-Out-r31r2we0.aqf' '--stderr=Microsoft-MIEngine-Error-k5sgs0cs.05h' '--pid=Microsoft-MIEngine-Pid-0etpztpc.32s' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

-----------PROGRAM FOR INFIX TO POSTFIX CONVERSION AND EVALUATION----------------

MENU:

1. Infix to Postfix

2. Postfix Evaluation

3. Exit

Enter choice: 1

--------------INFIX -> POSTFIX-----------------

Enter the infix expression: (A+B)/(C+D)\*(E-F)\*G

Postfix expression: AB+CD+/EF-\*G\*

Enter choice: 2

-----------------POSTFIX EVALUATION------------------

Enter the postfix expression: 23+4\*759+/-

Value of the given PostFix expression: 19.50

Enter choice: 3

----------------EXITING---------------------

**EXPERIMENT 14**

**CODE:**

#include<bits/stdc++.h>

using namespace std;

#define n 100

class Queue {

    int arr[n];

    int front;

    int rear;

    public:

    Queue() {

        front = -1;

        rear = -1;

    }

    void insert(int x) {

        if(rear == n-1) {

            cout << "Queue Overflow\n";

        }

        else {

            if(front == -1) {

                front ++;

            }

            rear = rear + 1;

            arr[rear] = x;

        }

    }

    int Delete() {

        int item;

        if(rear == -1 and front == -1) {

            cout << "Queue Empty\n";

        }

        else if(front == rear) {

            item = arr[front];

            front = -1;

            rear = -1;

        }

        else {

            item = arr[front];

            front += 1;

        }

        return front;

    }

    void display() {

        cout << "\n----------------DISPLAY------------------\n";

        cout << "Front: " << front << endl;

        cout << "Rear: " << rear << endl;

        if(front == -1 and rear == -1) {

            cout << "Queue is Empty\n";

            return;

        }

        for(int i = front; i <= rear; i++) {

            cout << arr[i] << " ";

        }

        cout << endl;

        cout << "-------------------------------------------\n\n";

    }

};

int main() {

    Queue q;

    cout << "\n----------PROGRAM FOR SIMULATING JOB QUEUE--------------\n";

    cout << "\nMENU:\n1. Insert Job\n2. Delete Job\n3. Display\n4. Exit\n";

    do {

        cout << "\nEnter choice: ";

        int choice;

        cin >> choice;

        switch(choice) {

            case 1:

                cout << "\n--------------INSERT----------------\n";

                cout << "Enter value: ";

                int x;

                cin >> x;

                q.insert(x);

                cout << "\n-------------------------------------\n";

                break;

            case 2:

                cout << "\n---------------DELETE----------------\n";

                q.Delete();

                break;

            case 3:

                q.display();

                break;

            case 4:

                cout << "\n------------EXITING-------------\n";

                exit(0);

        }

    }

    while(true);

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-kgbqkkhq.4vg' '--stdout=Microsoft-MIEngine-Out-mb5pk0zm.yga' '--stderr=Microsoft-MIEngine-Error-mtaa5hce.tll' '--pid=Microsoft-MIEngine-Pid-2gcl001s.eng' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

----------PROGRAM FOR SIMULATING JOB QUEUE--------------

MENU:

1. Insert Job

2. Delete Job

3. Display

4. Exit

Enter choice: 1

--------------INSERTION----------------

Enter Job: 1

-------------------------------------

Enter choice: 1

--------------INSERTION----------------

Enter Job: 2

-------------------------------------

Enter choice: 1

--------------INSERTION----------------

Enter Job: 3

-------------------------------------

Enter choice: 1

--------------INSERTION----------------

Enter Job: 4

-------------------------------------

Enter choice: 1

--------------INSERTION----------------

Enter Job: 5

-------------------------------------

Enter choice: 3

----------------DISPLAY------------------

Front: 0

Rear: 4

1 2 3 4 5

-------------------------------------------

Enter choice: 2

---------------DELETION----------------

Enter choice: 3

----------------DISPLAY------------------

Front: 1

Rear: 4

2 3 4 5

-------------------------------------------

Enter choice: 2

---------------DELETION----------------

Enter choice: 3

----------------DISPLAY------------------

Front: 2

Rear: 4

3 4 5

-------------------------------------------

Enter choice: 2

---------------DELETION----------------

Enter choice: 3

----------------DISPLAY------------------

Front: 3

Rear: 4

4 5

-------------------------------------------

Enter choice: 2

---------------DELETION----------------

Enter choice: 3

----------------DISPLAY------------------

Front: 4

Rear: 4

5

-------------------------------------------

Enter choice: 2

---------------DELETION----------------

Enter choice: 3

----------------DISPLAY------------------

Front: -1

Rear: -1

Queue is Empty

Enter choice: 1

--------------INSERTION----------------

Enter Job: 1

-------------------------------------

Enter choice: 3

----------------DISPLAY------------------

Front: 0

Rear: 0

1

-------------------------------------------

Enter choice: 4

------------EXITING-------------

**EXPERIMENT 15**

**CODE:**

#include<iostream>

#include<stdlib.h>

#define size 5

using namespace std;

class Dqueue {

    private:

    int que[size] ;

    public:

    int front,rear;

    Dqueue();

    int Qfull();

    int Qempty();

    int insert\_rear(int item);

    int delete\_front();

    int insert\_front(int item);

    int delete\_rear();

    void display();

};

Dqueue::Dqueue() {

    front=-1;

    rear= -1;

    for(int i=0;i <size;i++) {

        que[i]=-1;

    }

}

int Dqueue::Qfull() {

    if(rear==size-1)

        return 1;

    return 0;

}

int Dqueue::Qempty()

{

    if((front>rear) || (front==-1&&rear==-1))

        return 1;

    return 0;

}

int Dqueue::insert\_rear(int item)

{

    if(front == -1 && rear==-1)

        front++;

    que[++rear] = item;

    return rear;

}

int Dqueue::delete\_front()

{

    int item;

    if(front== -1)

        front++ ;

    item=que[front];

    que[front]= -1;

    front++ ;

    return item;

}

int Dqueue::insert\_front(int item)

{

    int i,j;

    if(front==-1)

        front++;

    i=front-1;

    while(i >= 0) {

        que[i+1] =que[i];

        i--;

    }

    j=rear;

    while(j >= front) {

        que[j+1]=que[j];

        j--;

    }

    rear++ ;

    que[front] =item;

    return front;

}

int Dqueue::delete\_rear()

{

    int item;

    item=que[rear];

    que [rear] = -1 ;/\*logical deletion\*/

    rear--;

    return item;

}

void Dqueue::display()

{

    int i;

    cout<<"\nStraight Queue is:";

    for(i=front;i <= rear; i ++) {

        cout<<" "<<que[i];

    }

}

int main() {

    int choice,item;

    char ans;

    ans='y';

    Dqueue obj;

    cout<<"\n\t\t Program For doubly ended queue using arrays";

    do

    {

        cout<<"\n1.insert by rear\n2.delete by front\n3.insert by front\n4.delete by rear";

        cout<<"\n5.display\n6.exit";

        cout<<"\n Enter Your choice ";

        cin>>choice;

        switch(choice) {

            case 1:

                if(obj.Qfull())

                    cout<<"\n Doubly ended Queue is full";

                else

                {

                    cout<<"\n Enter The item to be inserted: ";

                    cin>>item;

                    obj.rear=obj.insert\_rear(item);

                }

                break;

            case 2:

                if(obj.Qempty())

                    cout<<"\n Doubly ended Queue is Empty";

                else

                {

                    item= obj.delete\_front();

                    cout<<"\n The item deleted from queue is "<<item;

                }

                break;

            case 3:

                if(obj.Qfull())

                    cout<<"\n Doubly ended Queue is full";

                else

                {

                    cout<<"\n Enter The item to be inserted: ";

                    cin>>item;

                    obj.front= obj.insert\_front(item);

                }

                break;

            case 4:

                if(obj.Qempty())

                    cout<<"\n Doubly ended Queue is Empty";

                else {

                    item= obj.delete\_rear();

                    cout<<"\n The item deleted from queue is "<< item;

                }

                break;

            case 5:

                obj.display();

                break;

            case 6:

                exit(0);

        }

    }

    while(true);

    return 0;

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-ggqi1ktl.oyz' '--stdout=Microsoft-MIEngine-Out-xyzpq15g.vem' '--stderr=Microsoft-MIEngine-Error-2kekc5om.4h3' '--pid=Microsoft-MIEngine-Pid-b1ld0njc.nfs' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

Program For doubly ended queue using arrays

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 3

Enter The item to be inserted1

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 3

Enter The item to be inserted1

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 5

Straight Queue is: 1 1

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 1

Enter The item to be inserted2

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 5

Straight Queue is: 1 1 2

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 1

Enter The item to be inserted3

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 5

Straight Queue is: 1 1 2 3

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 3

Enter The item to be inserted4

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 5

Straight Queue is: 4 1 1 2 3

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 2

The item deleted from queue is 4

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 5

Straight Queue is: 1 1 2 3

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 4

The item deleted from queue is 3

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 5

Straight Queue is: 1 1 2

1.insert by rear

2.delete by front

3.insert by front

4.delete by rear

5.display

6.exit

Enter Your choice 6

------------EXITING-------------

**EXPERIMENT 16**

**CODE:**

#include <iostream>

using namespace std;

#define SIZE 100

class Pr\_Q

{

    private:

    int que[SIZE];

    public:

    int rear,front;

    Pr\_Q();

    int insert(int rear,int front);

    int Qfull(int rear);

    int delet(int front);

    int Qempty(int rear,int front);

    void display(int rear,int front);

};

Pr\_Q::Pr\_Q()

{

    front=0;

    rear=-1;

}

int Pr\_Q::insert(int rear,int front)

{

    int item,j;

    cout<<"\nEnter the element: ";

    cin>>item;

    if(front ==-1)

        front++ ;

    j=rear;

    while(j>=0 && item<que[j])

    {

        que[j + 1] =que[j];

        j--;

    }

    que[j+1]=item;

    rear=rear+ 1;

    return rear;

}

int Pr\_Q::Qfull(int rear)

{

    if(rear==SIZE-1)

        return 1;

    return 0;

}

int Pr\_Q::delet(int front)

{

    int item;

    item=que[front];

    cout << "\n The item deleted is "<< item;

    front++;

    return front;

}

int Pr\_Q::Qempty(int rear,int front)

{

    if((front==-1) || (front>rear))

        return 1;

    return 0;

}

void Pr\_Q::display(int rear,int front)

{

    int i;

    cout<<"\n The queue is: ";

    for(i=front;i <=rear;i++) {

        cout<<" "<<que[i] ;

    }

}

int main() {

    int choice;

    char ans;

    Pr\_Q obj;

    do

    {

        cout<<"\n\t\t Priority Queue\n";

        cout<<"\n Main Menu";

        cout << "\n1.Insert\n2.Delete\n3.Display\n4.Exit\n";

        cout<<"\n Enter Your Choice: ";

        cin>>choice;

        switch(choice)

        {

            case 1:

                if(obj.Qfull(obj.rear))

                    cout<<"\n Queue IS full";

                else {

                    obj.rear= obj .insert( obj.rear ,obj.front);

                }

                break;

            case 2 :

                if( obj.Qempty( obj.rear ,obj.front))

                    cout<<"\n Cannot delete element";

                else {

                    obj.front= obj.delet( obj.front);

                }

                break;

            case 3 :

                if( obj.Qempty( obj.rear ,obj.front))

                    cout<<"\n Queue is empty";

                else {

                    obj.display( obj.rear ,obj.front);

                }

                break;

            case 4:

                cout << "\n-----------EXITING----------\n";

                exit(0);

            default:

                cout<<"\nWrong choice: ";

                break;

        }

    }

    while(true);

    return 0;

}

**OUTPUT:**

PS H:\Coding\FDS\expt> & 'c:\Users\Dell\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-isrsjdhp.ki1' '--stdout=Microsoft-MIEngine-Out-jqvz3erb.hyy' '--stderr=Microsoft-MIEngine-Error-j2uq5rbm.wnz' '--pid=Microsoft-MIEngine-Pid-jsp3a0hn.dhl' '--dbgExe=C:\MinGW\bin\gdb.exe' '--interpreter=mi'

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 1

Enter the element: 1

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 1

Enter the element: 2

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 1

Enter the element: 3

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 1

Enter the element: 4

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 3

The queue is: 1 2 3 4

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 2

The item deleted is 1

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 3

The queue is: 2 3 4

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 2

The item deleted is 2

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 3

The queue is: 3 4

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 2

The item deleted is 3

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 3

The queue is: 4

Priority Queue

Main Menu

1.Insert

2.Delete

3.Display

4.Exit

Enter Your Choice: 4

-----------EXITING----------