**Assignment 2**

**DSA**

**Group**

**Mahad Elahi L1F21BSCS0022**

**Diyan Saeed L1F21BSCS0783**

**LIST.H**

class List

{

public:

virtual void insert() = 0;

virtual void rearrange() = 0;

virtual void print();

};

**Node.h**

class Node {

public:

char data;

Node\* next;

};

Main.cpp

#include <iostream>

#include"Node.h"

using namespace std;

class LinkedList {

private:

Node\* head;

public:

LinkedList() {

head = NULL;

}

// Function to insert a new node at the end of the list

void insert(char val) {

Node\* newNode = new Node();

newNode->data = val;

newNode->next = NULL;

if (head == NULL) {

head = newNode;

return;

}

Node\* curr = head;

while (curr->next != NULL) {

curr = curr->next;

}

curr->next = newNode;

}

void rearrange() {

if (head == NULL || head->next == NULL) {

return;

}

// create three temporary linked lists to store numbers,lower and upper alphabets and symbols

LinkedList numbersList, lowercaseList, symbolsList, uppercaseList;

Node\* curr = head;

while (curr != NULL) {

if (curr->data >= '0' && curr->data <= '9') {

numbersList.insert(curr->data);

}

else if (curr->data >= 'a' && curr->data <= 'z') {

lowercaseList.insert(curr->data);

}

else if (curr->data >= 'A' && curr->data <= 'Z') {

uppercaseList.insert(curr->data);

}

else {

symbolsList.insert(curr->data);

}

curr = curr->next;

}

// join the three lists in the desired order

head = NULL;

if (numbersList.head != NULL) {

head = numbersList.head;

}

if (lowercaseList.head != NULL) {

if (head == NULL) {

head = lowercaseList.head;

}

else {

Node\* curr = head;

while (curr->next != NULL) {

curr = curr->next;

}

curr->next = lowercaseList.head;

}

if (uppercaseList.head != NULL) {

if (head == NULL) {

head = uppercaseList.head;

}

else {

Node\* curr = head;

while (curr->next != NULL) {

curr = curr->next;

}

curr->next = uppercaseList.head;

}

}

if (symbolsList.head != NULL) {

if (head == NULL) {

head = symbolsList.head;

}

else {

Node\* curr = head;

while (curr->next != NULL) {

curr = curr->next;

}

curr->next = symbolsList.head;

}

}

}

}

// Function to print the contents of the linked list

void print() {

Node\* curr = head;

while (curr != NULL) {

cout << curr->data << " ";

curr = curr->next;

}

cout << endl;

}

};

int main() {

LinkedList myList;

myList.insert('A');

myList.insert('1');

myList.insert('b');

myList.insert('y');

myList.insert('6');

myList.insert('?');

myList.insert('F');

myList.insert('8');

myList.insert('&');

myList.insert('h');

myList.insert('j');

myList.insert('%');

cout << "Original list: ";

myList.print();

myList.rearrange();

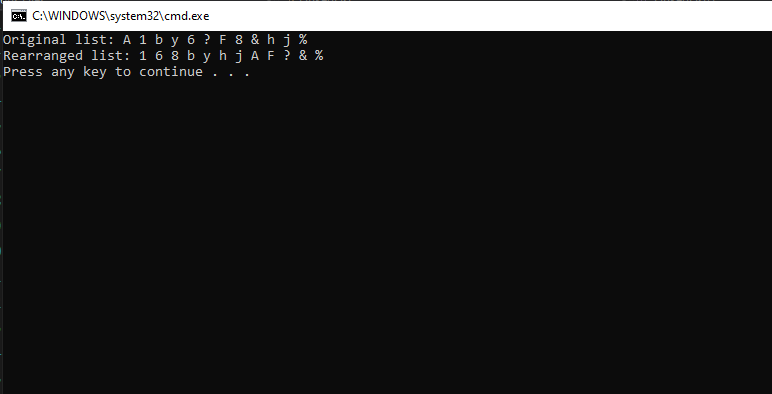
cout << "Rearranged list: ";

myList.print();

return 0;

}

**Screenshot:**

****