**Abstract Data Type**

#include <iostream>

using namespace std;

class linkedList {

private: // private members of the class

struct Node {

int data;

Node \*next;

};//end struct Node

public: // public memebers & member methods of the class

Node \*rear, \*start, \*temp, \*newptr;

//constructor

linkedList() {

this->start = this->rear = NULL;

}// end constructor

// function which takes an integer as it's parameter and return a Node type data item

Node \*createNewNode(int data) {

temp = new Node;

temp->data = data;

temp->next = NULL;

return temp;

}//end createNewNode

// function to insert data item into the beginning of linked list

void insertBeg(Node \*ptr) {

// if linked list is empty then start = rear = given\_node

if (start == NULL)

start = rear = ptr;

// else add to the beginning of linked list

else {

temp = start;

start = ptr;

ptr->next = temp;

}

}//end insertBeg

// function to insert data item in the end of linked list

void insertEnd(Node \*ptr) {

// if linked list is empty then start = rear = given\_node

if (rear == NULL)

start = rear = ptr;

// else add to the end of linked list

else {

rear->next = ptr;

rear = ptr;

}

}//end insertEnd

// function to display the items in the linked list

void display() {

Node \*np = start;

if (np == NULL)cout << "List Empty";

else cout << "The Linked List is ... \n";

while (np->next != NULL) {

cout << np->data << "->";

np = np->next;

}

cout << np->data << endl;

}//end display

// // function to remove the first item in linked list

void deleteFront() {

// if linked list is empty print Underflow

if (start == NULL)

cout << "Underflow\n";

// else print one by one

else {

temp = start;

start = start->next;

cout << "Deleted data ... " << temp->data << endl;

delete temp;

}

}//end deleteFront

// function to remove the last item in linked list

void deleteRear() {

// if linked list is empty print Underflow

if (start == NULL)

cout << "Underflow\n";

// else print one by one

else if (start->next == NULL) {

temp = start;

start = NULL;

cout << "Deleted data ... " << temp->data << endl;

delete temp;

} else {

temp = start;

while (temp->next->next != NULL) {

temp = temp->next;

}

rear = temp;

rear->next = NULL;

delete temp->next;

}

}//end deleteRear

};//end class linkedList

int main() {

// create an linkedList object which represents our ADT

linkedList object;

int d, choice;

// loop until user wishes to exit the program !

while (true) {

cout << "\n>> LINKED LIST CHOICES <<\n";

cout

<< "Enter 1 to enter data in the beginning of linked list.\nEnter 2 to enter data in the end of linked list.\n"

"Enter 3 to remove the first element in linked list\nEnter 4 to remove the last element in linked list\n"

"Enter 5 to display the Linked List \nEnter anything else to exit \n\n";

cin >> choice;

switch (choice) {

case 1 : // calling function to insert data at the beginning of linked list

cout << "Enter data:" << endl;

cin >> d;

object.newptr = object.createNewNode(d);

object.insertBeg(object.newptr);

break;

case 2 : // calling function to insert data at the end of linked list

cout << "Enter data:" << endl;

cin >> d;

object.newptr = object.createNewNode(d);

object.insertEnd(object.newptr);

break;

case 3 : // calling function to remove the first element of linked list

object.deleteFront();

break;

case 4 : // calling function to remove the last element of linked list

object.deleteRear();

break;

case 5 :

object.display();

break;

default:

cout << "Exiting the program." << endl;

exit(1);

}

}

return 0;

}

**Test Cases** 

