**TITLE:-** Implementation and Usage of Linked list

**Objective:-** Development of arithmetic expressions evaluation system.

**THEORY:- Write a C program to represent a character string in alinked list.**

• DSA : A data structure Is a named location that is used to store and organize data. And, an algorithm is a collection of steps to solve a particular problem. Learning data structures and algorithms allow us to write optimized and efficient computer programs.

• LINKED LIST : A linked list is a sequence of data structures, which are connected together via links. Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array.

**Implementation:**

#include<iostream>

#include<stdio.h>

using namespace std;

class Node{

public:

char data;

Node\* next;

Node(char data){

this-->data=data;

this-->next=NULL;

}

};

//take input for Node (Linked List) array into linkedlist

Node\* takeInput(char data[]){

Node \*head = NULL;

Node \*tail = NULL;

int i=0;

while(data[i]!='\0') {

//create new LL

Node \*node = new Node(data[i]);

if (head == NULL) {

head = node;

tail = node;

} else {

tail->next = node;//Inserting at tail

tail = tail->next;

}

i++;

}

return head;

}

//fuction for linked list to array convert

char\* linkedListToArray(Node \*head){

Node \*tempHead=head;

static char arr[50];

int i=0;

while(tempHead!=NULL){

arr[i]=tempHead->data;

tempHead=tempHead->next;

i++;

}

return arr;

}

//function for extract the sub-string part of the linked list.

char\* strsubl(Node \*head,int startingPoint,int endPoint){

Node \*tempHead=head;

static char subarr[50];

int count=0,i=0;

while(tempHead!=NULL){

if(startingPoint==count){

subarr[i]=tempHead->data;

tempHead=tempHead->next;

i++;

count++;

while(endPoint!=count){

subarr[i]=tempHead->data;

tempHead=tempHead->next;

i++;

count++;

}

subarr[i]=tempHead->data;

break;

}

tempHead=tempHead->next;

count++;

}

return subarr;

}

//combine two linked list

//Node\* zipll(Node \*head1,Node \*head2){

// Node \*tempHead1=head1;

// Node \*tempHead2=head2;

//

// Node \*combineHead=NULL;

// Node \*combineTail=NULL;

// while(tempHead1!=NULL || tempHead2!=NULL){

// Node \*combineNode=new Node(tempHead1->data);

// if(combineHead==NULL) {

// combineHead=combineNode;

// combineTail=combineTail;

// combineTail->data += tempHead2->data;

// tempHead1=tempHead1->next;

// tempHead2=tempHead2->next;

// }

// else{

// combineTail->data += tempHead2->data;

// combineTail->next=combineNode;

// combineTail=combineTail->next;

// tempHead1=tempHead1->next;

// tempHead2=tempHead2->next;

// }

//

// }

// return combineHead;

//}

Node\* zipll(Node \*head1,Node \*head2){

Node \*tempHead1=head1;

Node \*tempHead2=head2;

Node \*combineHead=NULL;

Node \*combineTail=NULL;

while(tempHead1!=NULL || tempHead2!=NULL){

Node \*combineNode=new Node(tempHead1->data+tempHead2->data);

if(combineHead==NULL) {

combineHead=combineNode;

combineTail=combineNode;

}

else{

combineTail->next=combineNode;

combineTail=combineTail->next;

}

tempHead1=tempHead1->next;

tempHead2=tempHead2->next;

}

return combineHead;

}

//Printing the data from linked list

void print(Node \*head){

Node \*tempHead=head;

while(tempHead!=NULL){

cout<<tempHead->data<<"->";

tempHead=tempHead->next;

}

cout<<"NULL";

}

int main(){

char word[50]="House";

//question no.1 array to linked list

Node \*head= takeInput(word);

print(head);

//question no.2 Linked List to Array

char \*arr= linkedListToArray(head);

int i=0;

cout<<endl;

while(arr[i]!='\0'){

cout<<arr[i];

i++;

}

//question no 3 extract the sub-string

char \*character= strsubl(head,1,3);

int j=0;

cout<<endl;

while(character[j]!='\0'){

cout<<character[j];

j++;

}

//question no.4 combine two liked list;

// char start1[50]="LneList";

// char start2[50]="ikd";

char start1[]="12345";

char start2[]="125";

Node \*head1= takeInput(start1);

Node \*head2= takeInput(start2);

Node \*combineHead = zipll(head1,head2);

cout<<endl;

print(combineHead);

//print(combineHead);

return 0;

}

**DISCUSSSIN AND CONCLUSION:**

**Here,** we got to know the implementation of given real problem by using Linked List.

Hence, we became familiar with Linked List by using real world problem.