**EX:5 DOUBLY LINKED LIIST-PALINDROME CHECKING**

**PROGRAM:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

// Define the struct node

struct node

{

char data;

struct node \* next;

struct node \* prev;

};

// Insert the characters in the work to a doubly linked list

void insertLast(struct node \*\*h, int val)

{

//create a new node

struct node \* newNode;

newNode = malloc(sizeof(struct node));

newNode -> data=val;

newNode -> prev=NULL;

newNode -> next=NULL;

//if head is NULL, it is an empty list

struct node \* temp = \*h;

if(\*h==NULL)

\*h=newNode;

else

{

//Otherwise, find the last node and add the newNode

while(temp->next!=NULL)

{

temp=temp->next;

}

temp->next=newNode;

newNode->prev=temp;

}

}

// Check for Palindrome using left and right struct pointer

int palindrome(struct node \*h){

struct node \*right = h;

struct node \*left = h;

int flag = 0;

// Move the right struct pointer to the last node

while(right->next!=NULL)

{

right=right->next;

}

// Move the left struct pointer and check is the data pointed by left and right are not same. If true, set the flag as 1 and break. Move the left and right pointer correspondingly

while(left->next!=NULL)

{

if(left->data!=right->data)

{

flag=1;

break;

}

else

{

left=left->next;

right=right->prev;

}

}

// return the flag variable

return flag;

}

// Display the doubly linked list

void display(struct node \*h)

{

if(h==NULL)

printf("Sorry Empty\n");

else

{

struct node \* temp=h;

while(temp!=NULL)

{

printf("%c->",temp->data);

temp = temp->next;

}

printf("\nNULL\n");

}

}

// Main function

int main()

{

struct node \*head1 = NULL;

printf("Name:R.Sridevi");

printf("\nRoll.no:20UIT021");

printf("\nProgram Name:Doubly linked list-Palindrome checking");

//Get the input value

char a[25],i;

printf("\nEnter the string to check Palindrome : \n");

scanf("%s",a);

// Generate a loop for length of the string and call the insertLast function by passing &head1 and input characters

for(i=0;i<strlen(a);i++)

{

insertLast(&head1,a[i]);

}

// Call the display function by passing the head1

display(head1);

// Declare a resultant variable and call the palindrome function by passing head1

int res=palindrome(head1);

// if the resultant variable is 0 the display palindrome else display not palindrome

if(res==1)

printf("Given string is not a Palindrome\n");

else

printf("Given string is a Palindrome\n");

return 0;

}

OUTPUT:

Name:R.Sridevi

Roll.no:20UIT021

Program Name:Doubly linked list-Palindrome checking

Enter the string to check Palindrome :

MALAYALAM

M->A->L->A->Y->A->L->A->M->

NULL

Given string is a Palindrome

Name:R.Sridevi

Roll.no:20UIT021

Program Name:Doubly linked list-Palindrome checking

Enter the string to check Palindrome :

WELCOME

W->E->L->C->O->M->E->

NULL

Given string is not a Palindrome