Experiment 19
Doubly Linked List

Date : 19-11-2020

Aim: To implement a Doubly Linked List and check whether the given string is palindrome

Data Structure used : Linked List

Algorithms

1. Algorithm for checking palindrome

Input: A Doubly Linked List with the Head pointing to the first element of the string and the Tail pointing to the last
Output: 1 if the string is palindrome 0 if otherwise
Data Structure: Doubly Linked List

Steps:

1. Step 1: if(Head==NULL)
2. Step 1: print(The list is empty)
3. Step 2: return 0
4. Step 2: else
5. Step 1: i = Header→rlink
6. Step 2: j = Tail → llink
7. Step 3: while(i!=Head and j!=Tail) do
8. Step 1: if(i→data!=j→data) then
9. Step 1: endWhile
10. Step 2: endif
11. Step 4: EndWhile
12. Step 5: if(i==Head and j==Tail) do
13. Step 1: return 1
14. Step 6: else
15. Step 1: return 0
16. Step 7: endif
17. Step 3: endif
18. Step 4: Sop

Result: the Program compiled successfully and the desired output was obtained.

Program code:

```
/************************************
 * Program to check whether the given
 * string is palindrome using doubly linked list
 * Done By: Rohit Karunaran
 * ***********************************/
#include<stdio.h>
#include<stdlib.h>

typedef struct char_doubly_linked_list
{
    struct char_doubly_linked_list *next;
    struct char_doubly_linked_list *prev;
    char data;
```

```
} ddchar;

void initString(ddchar **Header)
{
    *Header = (ddchar*)malloc(sizeof(ddchar));
    (*Header)->next = NULL;
    (*Header)->prev = NULL;
}

void insert(ddchar *Header,char ch)
{
    ddchar *newNode = (ddchar*)malloc(sizeof(ddchar));

    if(newNode!=NULL)
    {
        ddchar *Tail = Header;
        newNode->data = ch;
        if(Header->next == NULL) //That is the string is empty
        {
            Tail = NULL;
            Header->next = newNode;
            newNode->prev = Header;
            newNode->next=NULL;
        }
        else
        {
            while(Tail->next!=NULL) Tail = Tail->next;
            Tail->next = newNode;
            newNode->prev = Tail;
            newNode->next=NULL;
        }

    }
}
 void stringToList(ddchar *Header,char *s)
{
    for(int i=0;s[i]!='\0';i++)
        insert(Header,s[i]);
}

int checkPalindrome(ddchar *Header)
{
    ddchar *i,*j;
    if(Header->next!=NULL)
    {
        i=Header->next;
        j=Header;
        while(j->next!=NULL)j=j->next; //j becomes the tail pointer

        while(i!=NULL&&j!=Header)
        {
            if(i->data!=j->data)
                break;
            i=i->next;
            j=j->prev;
        }
```

```
        if(i==NULL && j==Header)
        {
            return 1;
        }
        return 0;
    }
    else{
        return 0;
    }
}

int main()
{
    ddchar *str = (ddchar*) malloc(sizeof(ddchar));
    initString(&str);
    char input[50];
    printf("Enter the string to be checked : ");
    scanf("%[^\n]%*c",input);
    stringToList(str,input);
    if(checkPalindrome(str))
    {
        printf("The String is palindorme");
    }
    else
    {
        printf("The String is not palindorme");
    }
    return 0;
}
```

Sample Input/Output

```
..ograming/C/CSL201/2020-11-16❱ ./palindrome.o
Enter the string to be checked : help
The String is not palindorme
..ograming/C/CSL201/2020-11-16❱ ./palindrome.o
Enter the string to be checked : malayalam
The String is palindorme
..ograming/C/CSL201/2020-11-16❱ ./palindrome.o
Enter the string to be checked : technology
The String is not palindorme
```