

CSE 2003: Lab Assignment #5

Due on Saturday, March 4, 2017

Shaik Naseera 2:00pm

Jacob John

Contents

Problem 1	3
-----------	---

Problem 1

Implement a doubly linked list program using C.

Listing 1: Doubly linked list program in C

```
/*
 * C Program to Implement a Doubly Linked List
 */
#include <stdio.h>
5 #include <stdlib.h>

struct node
{
    struct node *prev;
10    int n;
    struct node *next;
} *h, *temp, *temp1, *temp2, *temp4;

void insert1();
15 void insert2();
void insert3();
void traversebeg();
void traverseend(int);
void sort();
20 void search();
void update();
void delete();

int count = 0;
25

int main()
{
    int ch;

30    h = NULL;
    temp = temp1 = NULL;

    printf("\n 1 - Insert at beginning");
    printf("\n 2 - Insert at end");
35    printf("\n 3 - Insert at position i");
    printf("\n 4 - Delete at i");
    printf("\n 5 - Display");
    printf("\n 6 - Search for element");
    printf("\n 7 - Count");
40    printf("\n 8 - Exit");

    while (1)
    {
        printf("\n Enter choice : ");
45        scanf("%d", &ch);
        switch (ch)
        {
            case 1:
```

```

        insert1();
50      break;
    case 2:
        insert2();
        break;
    case 3:
55      insert3();
        break;
    case 4:
        delete();
        break;
60      case 5:
        traversebeg();
        break;
    case 6:
        search();
65      break;
    case 7:
        printf("Number of elements are: %d",count);
    case 8:
        exit(0);
70      default:
        printf("\n Wrong choice menu");
        }/*End of switch()*/
    }/*End of while()*/
}/*End of main*/
75
/* TO create an empty node */
void create()
{
    int data;
80
    temp =(struct node *)malloc(1*sizeof(struct node));
    temp->prev = NULL;
    temp->next = NULL;
    printf("\n Enter value to node : ");
85    scanf("%d", &data);
    temp->n = data;
    count++;
}/*End of create*/

90 /* TO insert at beginning */
void insert1()
{
    if (h == NULL)
    {
95        create();
        h = temp;
        temp1 = h;
    }
    else
100    {
        create();
    }
}
```

```
        temp->next = h;
        h->prev = temp;
        h = temp;
105     }
    }/*End of insert1()*/

    /* To insert at end */
    void insert2()
110 {
        if (h == NULL)
        {
            create();
            h = temp;
115         temp1 = h;
        }
        else
        {
            create();
120         temp1->next = temp;
            temp->prev = temp1;
            temp1 = temp;
        }
    }/*End of insert2()*/
125

    /* To insert at any position */
    void insert3()
    {
130         int pos, i = 2;

        printf("\n Enter position to be inserted : ");
        scanf("%d", &pos);
        temp2 = h;

135         if ((pos < 1) || (pos >= count + 1))
        {
            printf("\n Position out of range to insert");
            return;
        }
140         if ((h == NULL) && (pos != 1))
        {
            printf("\n Empty list cannot insert other than 1st position");
            return;
        }
145         if ((h == NULL) && (pos == 1))
        {
            create();
            h = temp;
            temp1 = h;
150             return;
        }
        else
        {
            while (i < pos)
```

```
155     {
        temp2 = temp2->next;
        i++;
    }
    create();
160    temp->prev = temp2;
    temp->next = temp2->next;
    temp2->next->prev = temp;
    temp2->next = temp;
    }
165 } /*End of insert3() */

/* To delete an element */
void delete()
{
170     int i = 1, pos;

    printf("\n Enter position to be deleted : ");
    scanf("%d", &pos);
    temp2 = h;

175     if ((pos < 1) || (pos >= count + 1))
    {
        printf("\n Error : Position out of range to delete");
        return;
180     }
    if (h == NULL)
    {
        printf("\n Error : Empty list no elements to delete");
        return;
185     }
    else
    {
        while (i < pos)
        {
190            temp2 = temp2->next;
            i++;
        }
        if (i == 1)
        {
195            if (temp2->next == NULL)
            {
                printf("Node deleted from list");
                free(temp2);
                temp2 = h = NULL;
200                return;
            }
        }
        if (temp2->next == NULL)
        {
205            temp2->prev->next = NULL;
            free(temp2);
            printf("Node deleted from list");
```

```
        return;
    }
210    temp2->next->prev = temp2->prev;
    if (i != 1)
        temp2->prev->next = temp2->next;
    if (i == 1)
        h = temp2->next;
215    printf("\n Node deleted");
    free(temp2);
}
count--;
}/*End of delete()*/
220
/* Traverse from beginning */
void traversebeg()
{
    temp2 = h;
225
    if (temp2 == NULL)
    {
        printf("List empty to display \n");
        return;
230    }
    printf("\n Linked list elements from begining : ");

    while (temp2->next != NULL)
    {
235        printf(" %d ", temp2->n);
        temp2 = temp2->next;
    }
    printf(" %d ", temp2->n);
}/*End of travarsebeg*/
240

/* To search for an element in the list */
void search()
{
245    int data, count = 0;
    temp2 = h;

    if (temp2 == NULL)
    {
250        printf("\n Error : List empty to search for data");
        return;
    }
    printf("\n Enter value to search : ");
    scanf("%d", &data);
255    while (temp2 != NULL)
    {
        if (temp2->n == data)
        {
260            printf("\n Data found in %d position", count + 1);
            return;
        }
    }
}
```

```

    }
    else
        temp2 = temp2->next;
        count++;
}
printf("\n Error : %d not found in list", data);
}/*End of search()*/

```

Output:

```

Linked list elements from begining : 20 10
Enter choice : 7
Number of elements are: 2
Enter choice : 8
Jacobs-MacBook-Pro:Downloads jacobjohn$ gcc doubly_linked_list.c
Jacobs-MacBook-Pro:Downloads jacobjohn$ ./a.out

1 - Insert at beginning
2 - Insert at end
3 - Insert at position i
4 - Delete at i
5 - Display
6 - Search for element
7 - Count
8 - Exit
Enter choice : 1

Enter value to node : 10

Enter choice : 1

Enter value to node : 20

Enter choice : 1

Enter value to node : 30

Enter choice : 4

Enter position to be deleted : 1

Node deleted
Enter choice : 5

Linked list elements from begining : 20 10
Enter choice : 6

Enter value to search : 10

Data found in 2 position
Enter choice : 7

Number of elements are: 2
Enter choice : 8
Jacobs-MacBook-Pro:Downloads jacobjohn$

```

```

37 printf("\n 5 - Display");
38 printf("\n 6 - Search for element");
39 printf("\n 7 - Count");
40 printf("\n 8 - Exit");
41
42 while (1)
43 {
44     printf("\n Enter choice : ");
45     scanf("%d", &ch);
46     switch (ch)
47     {
48         case 1:
49             insert1();
50             break;
51         case 2:
52             insert2();
53             break;
54         case 3:
55             insert3();
56             break;
57         case 4:
58             delete();
59             break;
60         case 5:
61             traversebeg();
62             break;
63         case 6:
64             search();
65             break;
66         case 7:
67             printf("\n Number of elements are: %d",count);
68             break;
69         case 8:
70             exit(0);
71         default:
72             printf("\n Wrong choice menu");
73     }/*End of switch()*/
74 }/*End of while()*/
75 }/*End of main*/
76
77 /* TO create an empty node */
78 void create()
79 {
80     int data;
81
82     temp =(struct node *)malloc(sizeof(struct node));
83     temp->prev = NULL;
84     temp->next = NULL;

```