CSE 2003: Lab Assignment #4

Due on Thursday, March 1, 2017

Shaik Naseera 2:00pm

Jacob John

Contents

Problem 1

Create a singly linked list program using C.

Listing 1: Singly linked list program in C

```
/*Program of single linked list*/
#include<stdio.h>
#include<stdlib.h>
#include<memory.h>
struct node
{
    int info;
    struct node *link;
};
struct node *create_list(struct node *start);
void display(struct node *start);
void count(struct node *start);
void search(struct node *start,int data);
struct node *addatbeg(struct node *start,int data);
struct node *addatend(struct node *start,int data);
struct node *addafter(struct node *start,int data,int item);
struct node *addbefore(struct node *start,int data,int item);
struct node *addatpos(struct node *start,int data,int pos);
struct node *del(struct node *start,int data);
struct node *reverse(struct node *start);

int main()
{
    struct node *start=NULL;
    int choice,data,item,pos;
    while(1)
    {
        printf("1.Create List\n");
        printf("2.Display\n");
        printf("3.Count\n");
        printf("4.Search\n");
        printf("5.Add to empty list / Add at beginning\n");
        printf("6.Add at end\n");
        printf("7.Add after node\n");
        printf("8.Add before node\n");
        printf("9.Add at position\n");
        printf("10.Delete\n");
        printf("11.Reverse\n");
        printf("12.Quit\n\n");
        printf("Enter your choice: ");
        scanf("%d",&choice);

        switch(choice)
        {
            case 1:
            start = create_list(start);
            break;
```

Problem 1 continued on next page...

```
            case 2:
            display(start);
            break;

            case 3:
            count(start);
            break;

            case 4:
            printf("Enter the element to be searched: ");
            scanf("%d",&data);
            search(start,data);
            break;

            case 5:
            printf("Enter the element to be inserted: ");
            scanf("%d",&data);
            start = addatbeg(start,data);
            break;

            case 6:
            printf("Enter the element to be inserted: ");
            scanf("%d",&data);
            start = addatend(start,data);
            break;

            case 7:
            printf("Enter the element to be inserted: ");
            scanf("%d",&data);
            printf("Enter the element before which to insert: ");
            scanf("%d",&item);
            start = addafter(start,data,item);
            break;

            case 8:
            printf("Enter the element to be inserted: ");
            scanf("%d",&data);
            printf("Enter the element before which to insert: ");
            scanf("%d",&item);
            start = addbefore(start,data,item);
            break;

            case 9:
            printf("Enter the element to be inserted: ");
            scanf("%d",&data);
            printf("Enter the position at which to insert: ");
            scanf("%d",&pos);
            start = addatpos(start,data,pos);
            break;

            case 10:
            printf("Enter the element to be deleted: ");
            scanf("%d",&data);
```

```
            start = del(start,data);
            display(start);
            break;

            case 11:
            start = reverse(start);
            break;

            case 12:
            exit(1);

        default:
            printf("Wrong choice\n");
        }/*End of switch*/
    }/*End of while*/
        }/*End of main*/

void display(struct node *start)
/*Function to display the linked list*/
{
    struct node *p;
    if(start==NULL) /*If linked list is empty*/
    {
        printf("List is empty\n");
        return;
    }
    p = start;
    printf("List is: \n"); /*If not then print list as follows*/
    while(p!=NULL)
    {
        printf("%d ",p->info);
        p = p->link;
    }
    printf("\n\n");
}/*End of display()*/

void count(struct node *start)
/*Counter for displaying number of nodes*/
{
    struct node *p;
    int cnt = 0;
    p = start;
    while(p!=NULL)
    {
        p = p->link; /*pointer towards link*/
        cnt++; /*adds one to the count*/
    }
    printf("Number of elements are %d\n",cnt);
}/*End of count()*/

void search(struct node *start,int item)
/*Function for searching an element in linked list*/
{
```

```
155     struct node *p=start;
        int pos = 1;
        while(p!=NULL)
        {
            if(p->info == item) /*If pointer results in target*/
160         {
                printf("Item %d found at position %d\n",item,pos);
                return;
            }
            p = p->link;
165         pos++;
        }
        printf("Item %d not found is list\n",item);
         /*Since last link points to NULL*/
}/*End of search()*/
170
struct node *addatbeg(struct node *start,int data)
/*Function to add an element to beginning of linked list*/
{
    struct node *tmp;
175 tmp = (struct node *)malloc(sizeof(struct node));
    tmp->info = data;
     /*Assign the element inputted by the user as first element*/
    tmp->link = start; /*add link to start*/
    start = tmp;
180 return start;
}/*End of addatbeg()*/

struct node *addatend(struct node *start,int data)
/*Function to add element to the end*/
185 {
    struct node *p,*tmp;
    tmp = (struct node *)malloc(sizeof(struct node));
    tmp->info = data; /*pointer*/
    p = start;
190 while(p->link!=NULL)
        p = p->link;
    p->link = tmp;
    tmp->link = NULL;
    return start;
195 }/*End of addatend()*/

struct node *addafter(struct node *start,int data,int item)
/*Function to add after node*/
{
200 struct node *tmp,*p;
    p = start;
    while(p!=NULL)
    {
        if(p->info == item)
205     {
            tmp = (struct node *)malloc(sizeof(struct node));
            tmp->info = data;
```

```
            tmp->link = p->link;
            p->link = tmp;
            return start;
        }
        p = p->link;
    }
    printf("%d not present in the list\n",item);
    return start;
}/*End of addafter()*/

struct node *addbefore(struct node *start,int data,int item)
/*Function to add before node*/
{
    struct node *tmp,*p;
    if(start == NULL)
    {
        printf("List is empty\n");
        return start;
    }
    /*If data to be inserted before first node*/
    if(item == start->info)
    {
        tmp = (struct node *)malloc(sizeof(struct node));
        tmp->info = data;
        tmp->link = start;
        start = tmp;
        return start;
    }
    p = start;
    while(p->link!=NULL)
    {
        if(p->link->info == item)
        {
            tmp = (struct node *)malloc(sizeof(struct node));
            tmp->info = data;
            tmp->link = p->link;
            p->link = tmp;
            return start;
        }
        p = p->link;
    }
    printf("%d not present in the list\n",item);
    return start;
}/*End of addbefore()*/

struct node *addatpos(struct node *start,int data,int pos)
/*Function to add at any desired position*/
{
    struct node *tmp,*p;
    int i;
    tmp = (struct node *)malloc(sizeof(struct node));
    tmp->info = data;
    if(pos==1)
```

```
    {
        tmp->link = start;
        start = tmp;
        return start;
    }
    p = start;
    for(i=1; i<pos-1 && p!=NULL;i++)
    p=p->link;
    if(p==NULL)
        printf("There are less than %d elements\n",pos);
    else
    {
        tmp->link = p->link;
        p->link = tmp;
    }
    return start;
}/*End of addatpos()*/

struct node *create_list(struct node *start)
/*Function to create a linked list*/
{
    int i,n,data;
    printf("Enter the number of nodes: ");
    scanf("%d",&n);
    start = NULL;
    if(n==0)
        return start;
    printf("Enter the element to be inserted: ");
    scanf("%d",&data);
    start = addatbeg(start,data);
    for(i=2; i<=n; i++)
    {
        printf("Enter the element to be inserted: ");
        scanf("%d",&data);
        start = addatend(start,data);
    }
    return start;
}/*End of create_list()*/

struct node *del(struct node *start,int data)
/*Function to delete a desired element*/
{
    struct node *tmp,*p;
    if(start == NULL)
    {
        printf("List is empty\n");
        return start;
    }
    if(start->info == data) /*Deletion of first node*/
    {
        tmp = start;
        start = start->link;
        free(tmp);
```

```
        return start;
    }
    p = start; /*Deletion in between or at the end*/
    while(p->link!=NULL)
    {
        if(p->link->info == data)
        {
            tmp = p->link;
            p->link = tmp->link;
            free(tmp);
            return start;
        }
        p = p->link;
    }
    printf("Element %d not found \n",data);
    return start;
}/*End of del()*/

struct node *reverse(struct node *start)
/*Function to reverse the linked list*/
{
    struct node *prev, *ptr, *next;
    prev = NULL;
    ptr = start;
    while(ptr!=NULL)
    {
        next = ptr->link;
        ptr->link = prev;
        prev = ptr;
        ptr = next;
    }
    start = prev;
    return start;
}/*End of reverse()*/
```

Output:

```
Jacobs-MacBook-Pro:~ jacobjohn$ gcc /Users/jacobjohn/OneDrive/Documents/VIT/Semester\ 2/CSE2003-Data\ Structures\ and\ Algorithms/Lab/assignment_4/single_linked_list.c
Jacobs-MacBook-Pro:~ jacobjohn$ ./a.out
1.Create List
2.Display
3.Count
4.Search
5.Add to empty list / Add at beginning
6.Add at end
7.Add after node
8.Add before node
9.Add at position
10.Delete
11.Reverse
12.Quit

Enter your choice: 1
Enter the number of nodes: 3
Enter the element to be inserted: 10
Enter the element to be inserted: 20
Enter the element to be inserted: 30
1.Create List
2.Display
3.Count
4.Search
5.Add to empty list / Add at beginning
6.Add at end
7.Add after node
8.Add before node
9.Add at position
10.Delete
11.Reverse
12.Quit

Enter your choice: 2
List is:
10 20 30

1.Create List
2.Display
3.Count
4.Search
5.Add to empty list / Add at beginning
6.Add at end
7.Add after node
```

single_linked_list.c

```c
/*Program of single linked list*/
#include<stdio.h>
#include<stdlib.h>
#include<memory.h>
struct node
{
    int info;
    struct node *link;
};
struct node *create_list(struct node *start);
void display(struct node *start);
void count(struct node *start);
void search(struct node *start,int data);
struct node *addatbeg(struct node *start,int data);
struct node *addatend(struct node *start,int data);
struct node *addafter(struct node *start,int data,int item);
struct node *addbefore(struct node *start,int data,int item);
struct node *addatpos(struct node *start,int data,int pos);
struct node *del(struct node *start,int data);
struct node *reverse(struct node *start);

int main()
{
    struct node *start=NULL;
    int choice,data,item,pos;
    while(1)
    {
        printf("1.Create List\n");
        printf("2.Display\n");
        printf("3.Count\n");
        printf("4.Search\n");
        printf("5.Add to empty list / Add at beginning\n");
        printf("6.Add at end\n");
        printf("7.Add after node\n");
        printf("8.Add before node\n");
        printf("9.Add at position\n");
        printf("10.Delete\n");
        printf("11.Reverse\n");
        printf("12.Quit\n\n");
        printf("Enter your choice: ");
        scanf("%d",&choice);

        switch(choice)
        {
            case 1:
            start = create_list(start);
```

Line: 335 | C | Tab Size: 4 | Symbols