Searching for a specific node in Doubly Linked List

We just need traverse the list in order to search for a specific element in the list. Perform following operations in order to search a specific operation.

- Copy head pointer into a temporary pointer variable ptr.

```
ptr = head
```

- declare a local variable I and assign it to 0.

```
i=0
```

- Traverse the list until the pointer ptr becomes null. Keep shifting pointer to its next and increasing i by +1.
- Compare each element of the list with the item which is to be searched.
- If the item matched with any node value then the location of that value I will be returned from the function else NULL is returned.

Algorithm

⇧

```
Step 1: IF HEAD == NULL

     WRITE "UNDERFLOW"
    GOTO STEP 8
    [END OF IF]

Step 2: Set PTR = HEAD

Step 3: Set i = 0

Step 4: Repeat step 5 to 7 while PTR != NULL

Step 5: IF PTR → data = item

   return i
   [END OF IF]

Step 6: i = i + 1

Step 7: PTR = PTR → next

Step 8: Exit
```

C Function

```c
#include<stdio.h>
#include<stdlib.h>
void create(int);
void search();
struct node
{
    int data;
    struct node *next;
    struct node *prev;
};
struct node *head;
```

```
void main ()
{
    int choice,item,loc;
    do
    {
        printf("\n1.Create\n2.Search\n3.Exit\n4.Enter your choice?");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            printf("\nEnter the item\n");
            scanf("%d",&item);
            create(item);
            break;
            case 2:
            search();
            case 3:
            exit(0);
            break;
            default:
            printf("\nPlease enter valid choice\n");
        }

    }while(choice != 3);
}
void create(int item)
{

   struct node *ptr = (struct node *)malloc(sizeof(struct node));
```

```
    if(ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {


    if(head==NULL)
    {
        ptr->next = NULL;
        ptr->prev=NULL;
        ptr->data=item;
        head=ptr;
    }
    else
    {
        ptr->data=item;printf("\nPress 0 to insert more ?\n");
        ptr->prev=NULL;
        ptr->next = head;
        head->prev=ptr;
        head=ptr;
    }
     printf("\nNode Inserted\n");
}


}
void search()
{
```

```
struct node *ptr;
int item,i=0,flag;
ptr = head;
if(ptr == NULL)
{
    printf("\nEmpty List\n");
}
else
{
    printf("\nEnter item which you want to search?\n");
    scanf("%d",&item);
    while (ptr!=NULL)
    {
        if(ptr->data == item)
        {
            printf("\nitem found at location %d ",i+1);
            flag=0;
            break;
        }
        else
        {
            flag=1;
        }
        i++;
        ptr = ptr -> next;
    }
    if(flag==1)
    {
        printf("\nItem not found\n");
        }
    }


}
```

Output

```
1.Create
2.Search
3.Exit
4.Enter your choice?1

Enter the item
23

Node Inserted

1.Create
2.Search
3.Exit
4.Enter your choice?1

Enter the item
90

Node Inserted

1.Create
2.Search
3.Exit
4.Enter your choice?2
```

```

Enter item which you want to search?
90

item found at location 1
```

← prev next →

Please Share

Learn Latest Tutorials

Swift

Pig

Flask

C. Graphics

⇧

Automata

Testing

Preparation

Aptitude

Reasoning

Verbal A.

Interview

B.Tech / MCA

DBMS

DS

DAA

OS

C. Network

Compiler D.

COA

D. Math.

E. Hacking

Web Tech.

Cyber Sec.

C

C++

Java

.Net

Python

Programs

Control S.

⇧