Daffodil International University

Course Tittle: Data Structure Lab

Course Code: SE 132

Assignment on:

Linked List

Submitted to:

Sazia Sharmin

Lecturer, Department of Software Engineering

Submitted by:

Name : Md. Shahin Alam Toha

ID: 222-35-1212

Section: E

Submission date: 01/10/2022

Q: Create a single linked list with 5 elements and define the following functions

1. Traverse each node of the list.

Source Code:

```c
#include<stdio.h>
#include<stdlib.h>

struct node
{
    int data;
    struct node *next;
};

int main()
{
    int i,n,item;
    struct node *p,*q,*head;

    printf("Enter the number of nodes: ");
    scanf("%d",&n);

    printf("Enter the value of the head node: ");
    scanf("%d",&item);

    q = (struct node *) malloc(sizeof(struct node));
    q->data=item;
    q->next=NULL;

    head = q;
    p = head;

    for(i=1; i<n; i++)
    {
    printf("Enter the value of the next node: ");
    scanf("%d",&item);

    q = (struct node *) malloc(sizeof(struct node));
    q->data=item;
    q->next=NULL;

    p->next=q;   //link the nodes
    p = p->next; //jump to the current node

    }

    printf("\n");
    p = head;
```

```
    while(p!=NULL)
    {
        printf("%d\t",p->data);
        p = p->next;
    }

  return 0;
}
```

Output:

```
PS C:\Users\MSI\Desktop\Programs> cd "c:\Users\MSI\Desktop\Programs\College works\SE131_Data_S
tructure\" ; if ($?) { gcc linkedList.c -o linkedList } ; if ($?) { .\linkedList }
Enter the number of nodes: 5
Enter the value of the head node: 3
Enter the value of the next node: 5
Enter the value of the next node: 7
Enter the value of the next node: 9
Enter the value of the next node: 1

3       5       7       9       1
PS C:\Users\MSI\Desktop\Programs\College works\SE131_Data_Structure>
```

2. Insert a node in the list.

a. Insert a node at head

```
#include<stdio.h>
#include<stdlib.h>

typedef struct node
{
    int data;
    struct node *next;
}node;

node* insertNodeAsHeadNode(int head_value , node* head)
{
    node* p;
    p=(node*)malloc(sizeof(node));
    p->data = head_value;
    p->next = NULL;

    p->next = head;

    head = p;
```

```
    return head;
}


int main()
{
    int a , i = 1 , n ,r , head_value;
    node *p,*q,*start;
    printf("Enter the number of nodes: ");
    scanf("%d",&n);

    printf("Enter node %d:  \n",i);
    p = (node*)malloc(sizeof(node));
    scanf("%d",&a);
    p->data = a;
    p->next = NULL;
    start = p;

    for(i=2;i<=n;i++)
        {
            printf("Enter node %d:  \n",i);
            q = (node*)malloc(sizeof(node));
            scanf("%d",&a);
            q->data = a;
            q->next = NULL;
            p->next = q;
            p = p->next;

        }

    p = start;
    while(p!=NULL)
    {
        printf("\t %d", p->data);
        p = p->next;
    }


    printf("\n \nEnter the value which you want to include as head node: ");
    scanf("%d",&head_value);

    start = insertNodeAsHeadNode(head_value , start);

    printf("\n");
    printf("Final Output: ");
    p = start;
    while(p!=NULL)
    {
        printf("\t %d",p->data);
        p = p->next;
```

```
    }

return 0;
}
```

Output:

```
PS C:\Users\MSI\Desktop\Programs> cd "c:\Users\MSI\Desktop\Programs\College works\SE131_Data_S
tructure\" ; if ($?) { gcc linkedList_InsertionatHead.c -o linkedList_InsertionatHead } ; if (
$?) { .\linkedList_InsertionatHead }
Enter the number of nodes: 5
Enter node 1:
3
Enter node 2:
5
Enter node 3:
7
Enter node 4:
9
Enter node 5:
1
        3       5       7       9       1

Enter the value which you want to include as head node: 10

Final Output:   10      3       5       7       9       1
PS C:\Users\MSI\Desktop\Programs\College works\SE131_Data_Structure>
```

---

b. Insert a node at intermediate position in Linked list

```
#include<stdio.h>
#include<stdlib.h>

typedef struct node
{
    int data;
    struct node *next;
}node;



node* insertAtIntermediatePosition(int node_data , int position , node* start)
{
    node *p , *q , *new_node;
    int i=0;
    p= start;

    while(i<position-1)
    {
        p=p->next;
```

```
        i++;
    }

    new_node = (node*)malloc(sizeof(node));
    new_node->next = NULL;
    new_node->data = node_data;

    q = p->next;
    p->next = new_node;
    new_node->next = q;

    return start;

}


int main()
{
    int a , i = 1 , n ,r , node_data , position;
    node *p,*q,*start;
    printf("Enter the number of nodes: ");
    scanf("%d",&n);

    printf("Enter node %d : \n",i);
    p = (node*)malloc(sizeof(node));
    scanf("%d",&a);
    p->data = a;
    p->next = NULL;
    start = p;

    for(i=2;i<=n;i++)
        {
            printf("Enter node %d : \n",i);
            q = (node*)malloc(sizeof(node));
            scanf("%d",&a);
            q->data = a;
            q->next = NULL;
            p->next = q;
            p = p->next;

        }

    p = start;
    while(p!=NULL)
    {
        printf("\t %d", p->data);
        p = p->next;
    }
```

```c
    printf("\nEnter the value of the node which you want to inset at Intermediate place: ");
    scanf("%d",&node_data);

    printf("\nEnter the position at which you want to place node: ");
    scanf("%d",&position);


    start = insertAtIntermediatePosition(node_data , position , start);

    printf("\n");
    p=start;
    while(p!=NULL)
    {
        printf("\t%d",p->data);
        p = p->next;
    }


return 0;
}
```

Output:

```
PS C:\Users\MSI\Desktop\Programs> cd "c:\Users\MSI\Desktop\Programs\College works\SE131_Data_S
tructure\" ; if ($?) { gcc linkedList_Insertion.c -o linkedList_Insertion } ; if ($?) { .\link
edList_Insertion }
Enter the number of nodes: 5
Enter node 1 :
1
Enter node 2 :
2
Enter node 3 :
3
Enter node 4 :
4
Enter node 5 :
5
        1       2       3       4       5
Enter the value of the node which you want to inset at Intermediate place: 60

Enter the position at which you want to place node: 2

        1       2       60      3       4       5
PS C:\Users\MSI\Desktop\Programs\College works\SE131_Data_Structure>
```

3. Delete a node from the list.

To delete a node from the linked list, we need to do the following steps.

1) Find the previous node of the node to be deleted.

2) Change the next of the previous node.

3) Free memory for the node to be deleted.

Source Code:

```c
#include <stdio.h>
#include <stdlib.h>

// A linked list node
struct Node {
    int data;
    struct Node* next;
};

/* Given a reference (pointer to pointer) to the head of a
list and an int, inserts a new node on the front of the
list. */
void push(struct Node** head_ref, int new_data)
{
    struct Node* new_node
        = (struct Node*)malloc(sizeof(struct Node));
    new_node->data = new_data;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}

/* Given a reference (pointer to pointer) to the head of a
list and a key, deletes the first occurrence of key in
linked list */
void deleteNode(struct Node** head_ref, int key)
{
    // Store head node
    struct Node *temp = *head_ref, *prev;

    // If head node itself holds the key to be deleted
    if (temp != NULL && temp->data == key) {
        *head_ref = temp->next; // Changed head
        free(temp); // free old head
        return;
    }

    // Search for the key to be deleted, keep track of the
    // previous node as we need to change 'prev->next'
```

```c
    while (temp != NULL && temp->data != key) {
        prev = temp;
        temp = temp->next;
    }

    // If key was not present in linked list
    if (temp == NULL)
        return;

    // Unlink the node from linked list
    prev->next = temp->next;

    free(temp); // Free memory
}

// This function prints contents of linked list starting
// from the given node
void printList(struct Node* node)
{
    while (node != NULL) {
        printf(" %d ", node->data);
        node = node->next;
    }
}

// Driver code
int main()
{
    /* Start with the empty list */
    struct Node* head = NULL;

    push(&head, 5);
    push(&head, 7);
    push(&head, 1);
    push(&head, 3);
    push(&head, 2);

    puts("Created Linked List: ");
    printList(head);
    deleteNode(&head, 1);
    puts("\nLinked List after Deletion of 1: ");
    printList(head);
    return 0;
}
```

Output:

```
PS C:\Users\MSI\Desktop\Programs> cd "c:\Users\MSI\Desktop\Programs\College works\SE131_Data_S
tructure\" ; if ($?) { gcc deletion.c -o deletion } ; if ($?) { .\deletion }
Created Linked List:
 2  3  1  7  5
Linked List after Deletion of 1:
 2  3  7  5
PS C:\Users\MSI\Desktop\Programs\College works\SE131_Data_Structure>
```