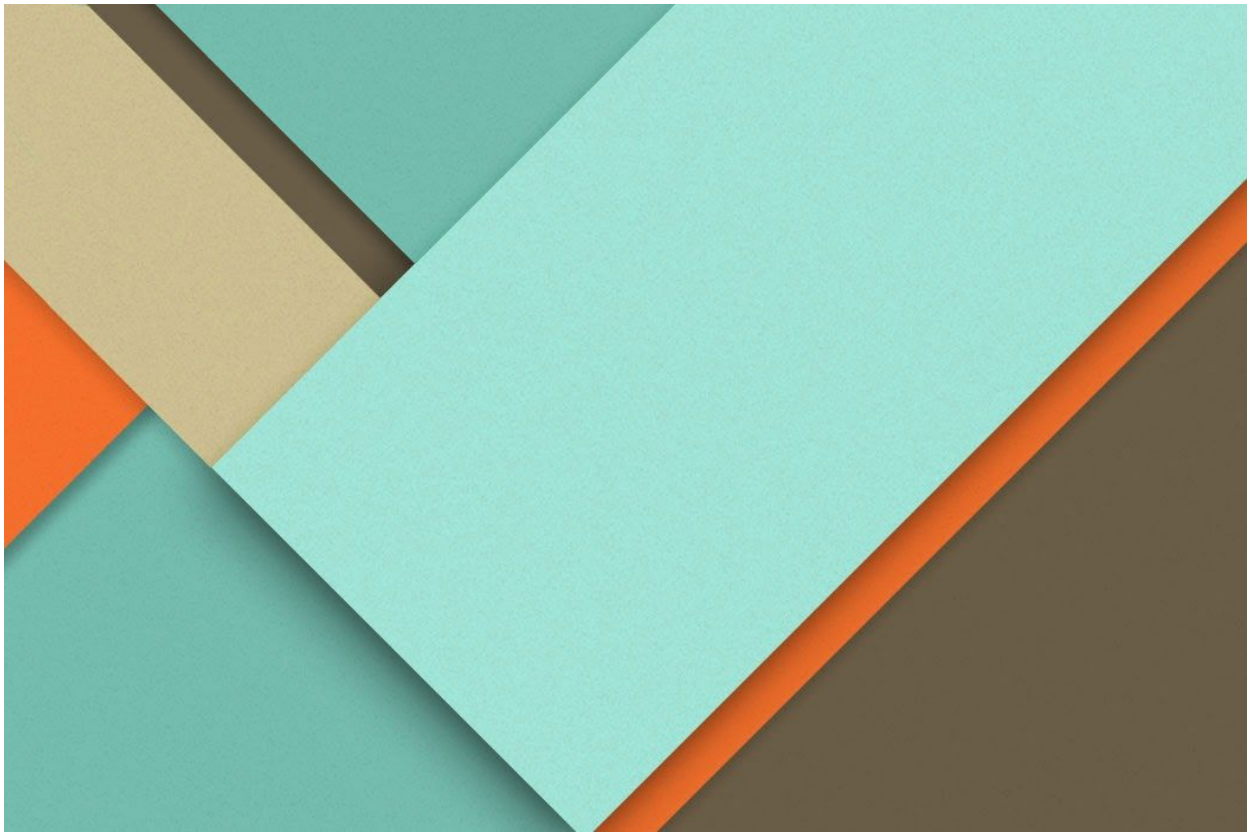

DSA Assignment

Boyapati Sai Venkat

AP19110010174

1st-year CSE-E.

Programs on the linked list

Write a menu-driven C Program to implement following operations (in the form of function)

on a link list.

a. Create an empty list.

b. Display the contents of the list

c. Insert an element at the beginning of the list.

d. Insert an element at the end of the list.

e. Insert an element at a given position. (e.g. 5th position)

f. Insert an element after a given number in the list.

g. Insert an element before a given number in the list.

h. Delete a given element from the list.

i. Reverse the contents of the link list.

j. Sum of all elements present in the list.

k. Print all the even number and odd number present in the list separately.

l. Count the frequency of occurrences of a given integer in the list.

Solution:

```
#include<stdio.h>

void createlinklist();
void displaylinklist();
void insertatbegining();
void insertatend();
void insertatapos();
void sumofelements();
void insertgivenbefore();
void insertgivenafter();
void deletegivenelement();
void reverseoflist();
void countofelement();
void rearrangeevenorodd();
struct node
{
        int info;
        struct node *next;
};
struct node *start=NULL;

void createlinklist()
{
        struct node *newnode,*ptr;
        newnode=(struct node *)malloc(sizeof(struct node));
```

```
        if(newnode==NULL)
        {
                printf("here not found anything.");
                exit(0);
        }
        printf("Enter the elements in the created node:");
        scanf("%d",&newnode->info);
        newnode->next=NULL;
        if(start==NULL)
        {
                start=newnode;
        }
        else
        {
                ptr=start;
                while(ptr->next!=NULL)
                {
                        ptr=ptr->next;
                }
                ptr->next=newnode;
        }
}

void displaylinklist()
{
        struct node *ptr;
        if(start==NULL)
        {
                printf("the created list has no elements the list is empty.");
```

```
            return;
        }
        else
        {
            ptr=start;
            printf("the elemments present in the created list are:");
            while(ptr!=NULL)
            {
                printf("%d",ptr->info );
                ptr=ptr->next ;
            }
        }
}

void insertatend()
{
        struct node *newnode,*ptr;
        newnode=(struct node *)malloc(sizeof(struct node));
        if(newnode==NULL)
        {
            printf("here not found anything.");
            return;
        }
        printf("Enter the elements in the created node:");
        scanf("%d",&newnode->info );
        newnode->next =NULL;
        if(start==NULL)
        {
            start=newnode;
```

```
        }
        else
        {
                ptr=start;
                while(ptr->next !=NULL)
                {
                        ptr=ptr->next ;
                }
                ptr->next =newnode;
        }
}

void insertatbegining()
{
        struct node *newnode;
        newnode=(struct node *)malloc(sizeof(struct node));
        if(newnode==NULL)
        {
                printf("here not found anything.");
                return;
        }
        printf("Enter the elements in the created node:");
        scanf("%d",&newnode->info);
        newnode->next =NULL;
        if(start==NULL)
        {
                start=newnode;
        }
        else
```

```
        {
                newnode->next=start;
                start=newnode;
        }
}

void insertatapos()
{
        struct node *ptr,*newnode;
        int i,position;
        newnode=(struct node *)malloc(sizeof(struct node));
        if(newnode==NULL)
        {
                printf("here not found anything.");
                return;
        }
        printf("\nEnter the position for the new node to be inserted:\t");
        scanf("%d",&position);
        printf("\nEnter the data value of the node:\t");
        scanf("%d",&newnode->info) ;

        newnode->next=NULL;
        if(position==0)
        {
                newnode->next=start;
                start=newnode;
        }
        else
        {
```

```
            for(i=0,ptr=start;i<position-1;i++)
            {
                  ptr=ptr->next;
                  if(ptr==NULL)
                  {
                        printf("the position of the given element does not found");
                        return;
                  }
            }
            newnode->next =ptr->next ;
            ptr->next=newnode;
      }
}

void sumofelements()
{
   struct node *newnode = start;
   int sum = 0;
   while(newnode != NULL)
   {
      sum = sum + newnode->info;
      newnode = newnode->next;
   }
   return sum;
}

void insertgivenbefore()
{
   struct node *pp,*p,*newnode;
```

```
int element,location=-1,position,i,count=0;
printf("At what before number you need to insert");
scanf("%d",&element);
struct node *ptr=start;
while(ptr!=NULL)
{
    if(ptr->info==element)
    {
        location=location+1;
        break;
    }
    count=count+1;
    ptr=ptr->next;
}
position=count+1;
newnode=(struct node *)malloc(sizeof(struct node));
printf("Enter the data");
scanf("%d",&newnode->info);
if(location==-1)
{
    printf("The element is not found");
}
else if(position==1)
{
    newnode->next=start;
    start=newnode;
}
else
{
```

```
        p=start;
        pp=NULL;
        for(i=1;i<position;i++)
        {
            pp=p;
            p=p->next;
        }
        pp->next=newnode;
        newnode->next=p;
    }
}

void insertgivenafter()
{
    struct node *ptr,*newnode,*pp,*p;
    int element,location=-1,position,count=0,i;
    printf("At what after number you need to insert");
    scanf("%d",&element);
    ptr=start;
    while(ptr!=NULL)
    {
        if(ptr->info==element)
        {
            location=location+1;
            break;
        }
        count=count+1;
        ptr=ptr->next;
    }
```

```
    position=count+1;
    if(location==-1)
    {
        printf("The element is not found");
    }
    else
    {
        newnode=(struct node *)malloc(sizeof(struct node));
        printf("Enter the data");
        scanf("%d",&newnode->info);
        p=start;
        pp=NULL;
        for(i=0;i<position;i++)
        {
            pp=p;
            p=p->next;
        }
        pp->next=newnode;
        newnode->next=p;
    }
}

void deletegivenelement()
{
    struct node *newnode,*q,*ptr,*p,*pp;
    int element,location=-1,position,count=0,i;
    printf("Which number you need to delete");
    scanf("%d",&element);
    ptr=start;
```

```
while(ptr!=NULL)
{
    if(ptr->info==element)
    {
        location=location+1;
        break;
    }
    count=count+1;
    ptr=ptr->next;
}
position=count+1;
if(location==-1)
{
    printf("The element is not found");
}
else if(position==1)
{
    q=start;
    start=start->next;
    free(q);
}
else
{
    p=start;
    pp=NULL;
    for(i=1;i<position;i++)
    {
        pp=p;
        p=p->next;
```

```
        }
        pp->next=p->next;
        free(p);
    }
}


void reverseoflist()
{
    struct node *ptr,*s,*e;
    int length=0,i,j,k,temp;
    ptr=start;
    while(ptr!=NULL)
    {
        length=length+1;
        ptr=ptr->next;
    }
    s=e=start;
    i=0;
    j=length-1;
    while(i<j)
    {
        k=0;
        while(k<j)
        {
            e=e->next;
            k=k+1;
        }
        temp=s->info;
```

```
        s->info=e->info;
        e->info=temp;
        i=i+1;
        j=j-1;
        s=s->next;
        e=start;
    }
}

void rearrangeevenorodd()
{
    int location=-1,position=-1;
    struct node *ptr=start,*p=start;
    printf("Even numbers\n");
    while(ptr!=NULL)
    {
        if((ptr->info)%2==0)
        {
            location=location+1;
            printf("%d\n",ptr->info);
        }
        ptr=ptr->next;
    }
    if(location==-1)
    {
        printf("There are no even numbers");
    }
    printf("Odd numbers\n");
    while(p!=NULL)
```

```
    {
        if((p->info%2)!=0)
        {
            position=position+1;
            printf("%d\n",p->info);
        }
        p=p->next;
    }
    if(position==-1)
    {
        printf("There are no odd numbers");
    }
}
void countofelement()
{
    int count=0,element;
    printf("Enter the number");
    scanf("%d",&element);
    struct node *ptr=start;
    while(ptr!=NULL)
    {
        if(ptr->info==element)
        {
            count=count+1;
        }
        ptr=ptr->next;
    }
    printf("The frequency of occurence of %d in the list is %d",element,count);
}
```

```
void main()
{
   int choice;
   while(10)
   {


printf("\n1.createlinklist\n2.displaylinklist\n3.insertatbegining\n4.insertatend\n5.insertatapos\n6.in
sertgivenbefore\n7.insertgivenafter\n8.deletegivenelement\n9.reverseoflist\n10.sumofelements\
n11.rearrangeevenorodd\n12.countofelement\n13.exit");
   printf("\nEnter your choice");
   scanf("%d",&choice);
   switch(choice)
   {
      case 1:
         createlinklist();
         break;
      case 2:
         displaylinklist();
         break;
      case 3:
         insertatbegining();
         break;
      case 4:
         insertatend();
         break;
      case 5:
         insertatapos();
         break;
```

```
        case 6:
            insertgivenbefore();
            break;
        case 7:
            insertgivenafter();
            break;
        case 8:
            deletegivenelement();
            break;
        case 9:
            reverseoflist();
            break;
        case 10:
            sumofelements();
            break;
        case 11:
            rearrangeevenorodd();
            break;
        case 12:
            countofelement();
            break;
        case 13:
            exit(10);
        default:
            printf("\nchoose the above options");
    }
    }
}
```

Output:

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice1

Enter the elements in the created node:1

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice1

Enter the elements in the created node:2

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice1

Enter the elements in the created node:3

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice2

the elements present in the created list are:123

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice3

Enter the elements in the created node:4

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice3 4

Enter the elements in the created node:5

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice5

Enter the position for the new node to be inserted: 2

Enter the data value of the node: 6

1.createlinklist
2.displaylinklist
3.insertatbegining
4.insertatend
5.insertatapos
6.insertgivenbefore
7.insertgivenafter
8.deletegivenelement
9.reverseoflist
10.sumofelements
11.rearrangeevenorodd
12.countofelement
13.exit
Enter your choice6
At what before the number you need to insert3
Enter the data7

1.createlinklist
2.displaylinklist
3.insertatbegining
4.insertatend
5.insertatapos
6.insertgivenbefore
7.insertgivenafter
8.deletegivenelement
9.reverseoflist
10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice8

Which number you need to delete4

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice2

the elements present in the created list are:162735

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice12

Enter the number4

The frequency of occurrence of 4 in the list is 0

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice11

Even numbers

6

2

Odd numbers

1

7

3

5

1.createlinklist

2.displaylinklist

3.insertatbegining

4.insertatend

5.insertatapos

6.insertgivenbefore

7.insertgivenafter

8.deletegivenelement

9.reverseoflist

10.sumofelements

11.rearrangeevenorodd

12.countofelement

13.exit

Enter your choice