EXP NO: 01 **PAGE NO: 01**

DATE:

IMPLEMENTATION OF SINGLY LINKED LIST AND ITS OPERATION

PROGRAM 1: LINKED LIST TO ARRAY

ALGORITHM:

STEP 1: Start

STEP 2: Create object and allocate memory using malloc function.

STEP 3: Get the value to be stored in the node ; store it in the data field

STEP 4: insert the node into the head

STEP 5: initialize an array

5.1: store the data from the linked list to the array

STEP 6: display the elements of the array

STEP 7: Stop.

PROGRAM 1-CODING:

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

struct node{
   int value;
   struct node *next;
}*head=NULL;

struct node *temp;
void insert()
{
```

```
    int data;
    struct node *newnode;
    newnode=(struct node*)malloc(sizeof(struct node));
    printf("\n enter the data :");
    scanf("%d",&data);
    newnode->value=data;
    newnode->next=NULL;
    if (head==NULL){
        head=newnode;
    }
    else{
        newnode->next=head;
        head=newnode;
    }
}

void convert()
{
    int arr[100],i=0,j;
    int n=count();
    printf("%d",n);
    temp=head;
    for(i=0;i<n;i++){
        arr[i]=temp->value;
        temp=temp->next;
    }
    printf("\n displaying array:");
    for(j=0;j<n;j++)
    {
        printf("%d \t",arr[j]);
    }
```

```
}

int count()
{
   int count=0;
   temp=head;
   while(temp!=NULL){
      count++;
      temp=temp->next;
   }
   return count;
}

int main()
{
   int choice;
   while(1)
   {
       printf("\n 1.insert  2.convert");
       printf("\nenter choice: ");
       scanf("%d",&choice);
      switch(choice)
      {
         case 1:
         insert();
         break;

         case 2:
         convert();
         break;
```

```
        default:
        printf("\n thank you");
        exit(0);
      }
   }
   return 0;
}
```

PROGRAM 1-OUTPUT:

```
1.  insert  2.convert
enter choice:1
Enter data:10
1.  insert  2.convert
enter choice:1
Enter data:20
1.  insert  2.convert2
10 20
1.  insert  2.convert2
Enter choice:3
Thank you
```

PROGRAM 2: FIND XTH NODE FROM THE END OF THE LINKED LIST

ALGORITHM:

STEP 1: Start

STEP 2: Create object and allocate memory using malloc function.

STEP 3: Get the value to be stored in the node ; store it in the data field

STEP 4: add the nodes to the linked list

STEP 5: reverse the linked list and create a head to it

STEP 6: enter the x node to be displayed

STEP 7: traverse to the xth node and display the node

STEP 8: Stop

PROGRAM 2-CODING:

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
   int data;
   struct node *next;
};
typedef struct node node;
node *head = NULL, *chead = NULL, *dummy;
void insert_last(int data)
{
   node *new = (node *)malloc(sizeof(node));
   new->next = NULL;
   new->data = data;
   if (head == NULL)
      head = new;
   else
   {
      node *temp = head;
      while (temp->next != NULL)
         temp = temp->next;
      temp->next = new;
   }
}

void reversal()
{
```

```
    node *temp;
    // To find chead
    temp = head;
    while (temp->next != NULL)
    {
        temp = temp->next;
    }
    chead = temp;
    dummy = chead;
    while (1)
    {
        temp = head;
        while (temp->next != chead)
            temp = temp->next;
        chead->next = temp;
        chead = chead->next;
        if (chead == head)
            break;
    }
    chead->next = NULL;
}

void find(int x)
{
    node *temp;
    int i=0;
    temp=dummy;
    while(i<x)
    {
        temp=temp->next;
        i++;
```

```
    }
    printf("\n the value of xth node from last is:%d",temp->data);
}
int main()
{
    int choice,data,x;
    while (1)
    {
        printf("\n enter your choice:");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
        printf("\n enter the no.");
        scanf("%d",&data);
        insert_last(data);
        break;

        case 2:
        printf("\n find xth element from last");
        printf("\n enter the xth element:");
        scanf("%d",&x);
        reversal();
        find(x);
        break;

        default:
        printf("\n thank you");
        exit(0);
        }
    }
```

```
}
```

PROGRAM 2-OUTPUT:

```
enter your choice:1
Enter data:10
enter your choice:1
Enter data:20
enter your choice:1
Enter data:30
enter your choice:2
find xth element from last
enter the xth element:1
20
```

PROGRAM 3: REVERSE A LINKED LIST IN K-GROUPS

ALGORITHM:

STEP 1: Start

STEP 2: Create object and allocate memory using malloc function.

STEP 3: Get the value to be stored in the node ; store it in the data field

STEP 4: create a linked list and insert all data into the nodes of linked list

STEP 5: enter the value of k for the nodes to be grouped

STEP 6: reverse the k-groups and display the linked list

STEP 7: Stop.

PROGRAM 3-CODING:

```
#include<stdio.h>
#include<stdlib.h>
struct Node
{
int data;
```

```
struct Node* next;
}*head=NULL;
struct Node *newnode;
struct Node *reverse (struct Node *head, int k)
{
if (!head)
return NULL;
struct Node* current = head;struct Node* next = NULL;
struct Node* prev = NULL;
int count = 0;
while (current != NULL && count < k)
{
next = current->next;
current->next = prev;
prev = current;
current = next;
count++;
}
if (next != NULL)
head->next = reverse(next, k);
return prev;
}
void insertlast(struct Node *Newnode)
{
int data;
struct Node *p;
Newnode=(struct Node*)malloc(sizeof(struct Node));
printf("Enter the value to be inserted->:");
scanf("%d",&data);
Newnode->data=data;
Newnode->next=NULL;
```

```
if(head==NULL)
{
head=Newnode;
}
else
{
p=head;
while(p->next!=NULL)
p=p->next;
p->next=Newnode;
}
}
void display(struct Node *node)
{
while(node->next!=NULL)
{
printf("%d->",node->data);
node=node->next;
}
printf("%d",node->data);
}
int main(void)
{int ch,n;
while(1)
{
printf("\nEnter the choice 1.Insert and Dislay 2.kreverse 3.Exit Enter choice::");
scanf("%d",&ch);
switch(ch)
{
case (1):
insertlast(newnode);
```

```
display(head);
break;
case (2):
printf("Enter at which K it should reverse::");
scanf("%d",&n);
head = reverse (head, n);
display(head);
break;
case (3):
exit(0);
}
}
return(0);
}
```

PROGRAM 3-OUTPUT:

Enter the choice 1.Insert and Display 2.kreverse 3.Exit Enter choice::1

Enter the value to be inserted->:1

Enter the choice 1.Insert and Display 2.kreverse 3.Exit Enter choice::1

Enter the value to be inserted->:2

Enter the choice 1.Insert and Display 2.kreverse 3.Exit Enter choice::1

Enter the value to be inserted->:3

Enter the choice 1.Insert and Display 2.kreverse 3.Exit Enter choice::2

Enter at which K it should reverse::2

2->1->3

DESCRIPTION	MAXIMUM MARK	MARKS SCORED
OBSERVATION	20	
RECORD	05	
TOTAL	25	

RESULT:

Thus the all the three given programs based on singly linked list are executed and outputs are verified.