~~Lab~~ Lab Prg-4

WAP to implement singly linked list with following operations

a) Create a linked list
b) Insertion of a node at 1st position, at any position, end of the list.

c) Display the contents of linked list.

Pseudo code.

```
Structure node
    Integer data
    struct node * Next;
    Return.
   head ← NULL;

Function insertAtBeginning (data)
  create node * newnode;
   newnode.data = data;
   newnode ->next = head;
    head = newnode;

Function insertAt position (value, position)
   if (position < 1 ) : Return
    create node * newnode
     if ( position == 1) {
            insert at beginning (value); }

  struct node * temp = head;
  for ( int i = 1 ; i < position - 1 && temp != NULL ; i++)
        temp = temp -> Next
 if (temp == NULL) {
      free(node);
  return ; }
```

```
newNode -> next = temp -> next;
 temp -> next = newnode;
  }.
MAIN:
   loop:
   print " Insert At Beginning, Insert at End,
        Insert At a position, Display, exit"
INPUT CHOICE
if choice 1 : input value
              Insert At Beginning (value);
              break
if choice 2 : Input value
              Insert At End (value);
               break
if choice 3 : Input value, position
              Insert At position (value, position);
              break
if choice 4 : Display
  return;

function Display
   struct node * temp = head;
      if (head == NULL) {
        printf ("empty");
         return; }

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

[illegible]
10/11

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

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

struct node* head = NULL;

void create (int n)
{
   struct node *newnode, *temp;
   int data, i;
   if (n<=0) return;
   for (i=0; i<n; i++)
   {
    newnode = (struct node*) malloc(sizeof
                    (struct node));
    scanf ("%d", &data);
    newNode -> data = data;
    newnode -> next = NULL;
     if (head == NULL)
         head = newnode;
    else
    {
       temp = head;
       while (temp -> next != NULL)
              temp = temp -> next;
       temp -> next = newnode;
    }
  }
}
```

```
void insertAtBeginning (int data)
{
   struct node * newnode = (struct node*)
    malloc (sizeof (struct node ));

   newnode -> data = data;
   newnode -> next = head;
    head = new node;
  }.

void insertAtEnd (int data)
{.
     struct Node * new node, *temp;
      newnode = (struct node *) malloc
                 (sizeof (struct node));
      newnode -> data = data;
      newnode -> next = NULL;
      if (head == NULL)
           head = new node;
      else
      {
          temp = head;
          while (temp->next != NULL)
               temp = temp -> next;
          temp -> next = newnode;
        }
      }

void insertAtposition (int data, int pos)
{
      int i;
      struct Node * newnode, * temp;
      newnode = (struct node * ) malloc
                 (sizeof (struct node ));
```

```
newnode -> data = data;
if (pos == 1)
{
  newnode -> next = head;
   head = newnode;
   return;
 }
 temp = head;
 for (i=1; i< pos-1 && temp != Null; i++)
      temp = temp -> next;
  if (temp == null) return;
  newnode -> next = temp -> next;
   temp -> next = newnode;
  }

void  display()
{
  struct Node * temp = head;
  while (temp != Null)
  {
   printf(" %d ", temp->data);
    temp = temp -> next;
  }
   printf("\n");
  }

int main()
{
 int n, choice, data, pos;
 printf("Enter number of initial nodes
            to create:");
 scanf(" %d ", &n);
 printf("Enter %d elements:", n);
 create(n);
```

```
printf("\n Linked list Menu \n 1. Insert
     at Beginning \n 2. Insert at end \n
     3. Insert at position \n 4. Display \n
     5. Exit \n");

   while (1) {
       printf("\n Enter your choice:");
       scanf("%d", &choice);

       switch (choice) {
        case 1:
            printf("Enter data:");
            scanf("%d", &data);
            ~~break~~

insertATBeginning (data);
                  break;
        case 2:
            printf("Enter data:");
            scanf("%d", &data);
            insertAtEnd (data);
            break;
        case 3:
            printf("Enter data:");
            scanf("%d", &data);
            printf("\n Enter pos");
            insertAtPosition (data, pos);
            break;

        case 4:
            printf("Linked list:");
            display();
            break;
case 5:
     exit(0);
default:
     printf("Invalid choice\n");
   }
  }
  return 0;
 }
```

output:

```
enter number of intial nodes to create: 4
Enter 4 elements: 2 3 4 5

Linked list menu
1. Insert at beginning
2. Insert at End
3. Insert at position
4  Display
5. Exit

enter your choice: 1
 Enter data: 1

Enter your choice: 2
 Enter data: 6

 enter your choice: 3
  enter data: 9

  enter pos 2
enter your choice: 4
Linked list: 1 9 2 3 4 5 6
```

Enter your choice :5

[illegible]