program6: Linked list application

1) creation 2) insertion 3) display

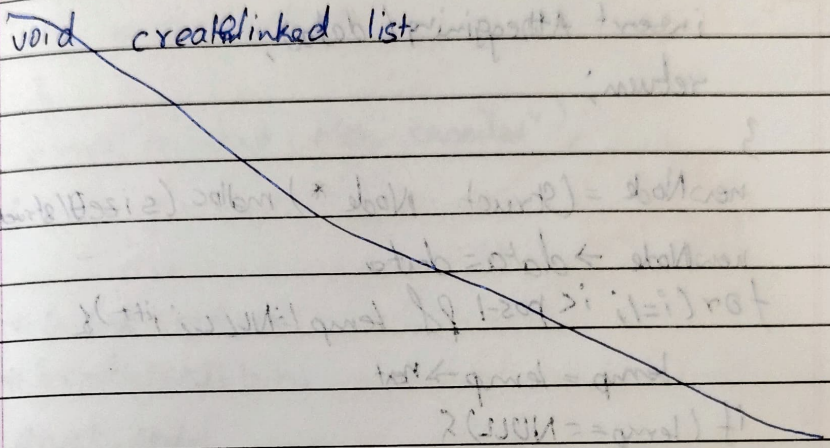
Date

Page

pseudo code:

```
void insertAtBeginning(int Data) {
    struct Node *newNode = (struct Node*) malloc
        (sizeOf(structNode))
    newNode -> data = data;
    newNode -> next = head;
    head = newNode
    printf("Node inserted at the beginning");
}

void creatLinked list
```



```
void insertAtEnd(int Data){
    struct Node *newNode = (struct Node*) malloc
        (sizeOf(struct(structNode))
    newNode -> data = data
    newNode -> next = NULL;
    if(head -> NULL){
        head = newNode;
    } else{
        struct Node *temp = head;
        while(tem ->next != NULL{
            temp = temp ->next;
            temp -> next = newNode.
    }
    printf("Node inserted at end"); }
```

```
void insertAtPosition(int Data, int pos){
    int i;
    Struct Node *newNode, *temp=head;
    if(pos<1){
    printf("wrong position");
    return;
}
 if(pos==1)
 {
    insertAtBeggining(data);
    return;
 }
 newNode=(Struct Node*)malloc(sizeof(struct
 newNode->data=data
 for(i=1; i<pos-1 && temp!=NULL; i++){
      temp=temp->next
 if(temp==NULL){
    printf("position out of range");
 }
 else{ newNode->next=temp->next;
      temp->next=newNode;}

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

void createList(int n){
    for(i=1; i<=n; i++){
    newNode=(struct Node*)malloc(sizeof
      struct Node));
```

```
if (newNode == NULL); }
scanf ("%d", &data
newNode -> data = data;
newNode -> next = NULL;
if (head == NULL){
    head = newNode
} else {
    temp -> next = newNode
}
temp = newNode;
}
printf("Linked list created");
}
```

code:

```
#include <stdio.h>
#include <stdlib.h>
struct Node{
  int datata
  structNode * next;
};

struct Node * head = NULL;

void createList (int n) {
  struct Node * newNode, * temp;
  int data, i;
  if (n <= 0){
  printf ("Number of nodes should be greater than 0\n");
  return;
  }
  for (i = 1, i <= n, i++){
    newNode = (struct Node*) malloc (sizeof (structNode))
```

```
if (newNode == NULL){
    printf(" Memory allocation failed \n");
    return;
}
printf(" Enter data for node %d: ", i);
scanf(" %d", &data);
newNode -> data = data;
newNode -> next = NULL;
if (head == NULL){
    head = newNode;
}else{
    temp -> next = newNode;
}
temp = newNode;
}
printf(" \n Linked list created ");
}

void insertAtBeginning(int data){
    struct Node *newNode = (struct Node*)malloc
                (sizeof(struct Node));
    newNode -> data = data;
    newNode -> next = head;
    head = newNode;
    printf("\n node inserted \n");
}

void insertAtEnd(int Data){
    struct Node *newNode = (struct Node*)malloc
                (sizeof(struct Node));
    newNode -> data = data;
    newNode -> next = NULL;
```

```
if (head == NULL) { head = newNode; }
else
struct Node *temp = head;
  while (temp -> next != NULL)
      temp = temp -> next;
  temp -> next = newNode;
}
  printf ("Node inserted at end");
}

void insertAtPosition(int data, int pos){
  int i;
  struct Node *newNode, *temp = head;
  if (pos<1){
    printf ("Invalid position \n");
    return;
  }
  if (pos==1){
    insertAtbeginning (data);
    return;
  }
  newNode = (struct Node*) malloc (sizeof (struct
                                    Node));
  newNode -> data = data;
  for (i=1; i<pos-1 && temp != NULL; i++)
      temp = temp -> next;
  if (temp == NULL){
      printf (" position out of range);
      free (newNode);
  }else {
      newNode -> next = temp -> next
      temp -> next = newNode;
      printf ("Inserted at position");
}}
```

```
void displaylist(){
    Struct Node * temp = head;
    if (head == NULL){
        printf("List is empty!");
        return;
    }
    printf("\n Linked list: ");
    while (temp != NULL){
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL");
}

int main(){
    int choice, n, data, pos;
    printf("\n 1) Create 2) insert At beginning
            3) insert at any position 4)
            insert at end 5) display 6) exit");
    printf("Enter your choice: ");
    scanf("%d", &choice);
    while(1){
        switch(choice){
        case 1:
            printf("enter number of nodes: ");
            scanf("%d", &n);
            createList(n);
            break;
        case 2: printf("Enter data to insert");
            scanf("%d", &data);
            insertAtBeginning(data);
            break;
```

```
case 3:
        printf("enter data and position: ");
        scanf (" %d %d", &data, &pos);
        insertAtposition(data, pos); break;
case 4:
        printf ("enter data to insert: ");
        scanf (" %d ", &data);
        insertAtEnd (data);
        break;
case 5:
        displayList();
        break;
case 6:
        printf ("Exiting \n");
        exit(0);
default:
        printf (" invalid choice ");
}
scan printf ("enter choice");
scanf ("%d", &choice);
}
}
```

output

```
--- Singly linked list operations ---
1) created linked list
2) insert at beginig
3) insert at any position
4) insert at End
5) display List
6) Exit
Enter your choice 1
Enter number of nodes 2
```

Enter data for node 1: 1
Enter data for node 2: 2
Linked list created
Enter your choice: 2
Enter data to insert 0
Node inserted at the beginning
Enter your choice 4
Enter data to insert 4
Enter your choice 3
Enter data and position 4
Enter your choice: 5
Linked list: 0 → 1 → 2 → 3 → 4 → NULL
Enter your choice 6
Exiting

Tracing --- Singly linked list operations ---
1) Create Linked list
2) insert at beginning
3) insert at any position
4) insert at end
5) display list
6) Exit
Enter your choice: 1
Enter number of nodes: 3
Enter data for node 1: 100
Enter data for node 2: 200
Enter data for node 3: 300
Linked list created
Enter your choice 3
Enter data & position: 250 3
Enter your choice 4
Enter data to insert: 500

Date
Page

Node inserted at the end

Enter your choice: 5

Linked list: 100 → 200 → 250 → 300 → 1500 → NULL

enter your choice: 6

Exiting

11/11/25