LAB-5 22/01/24

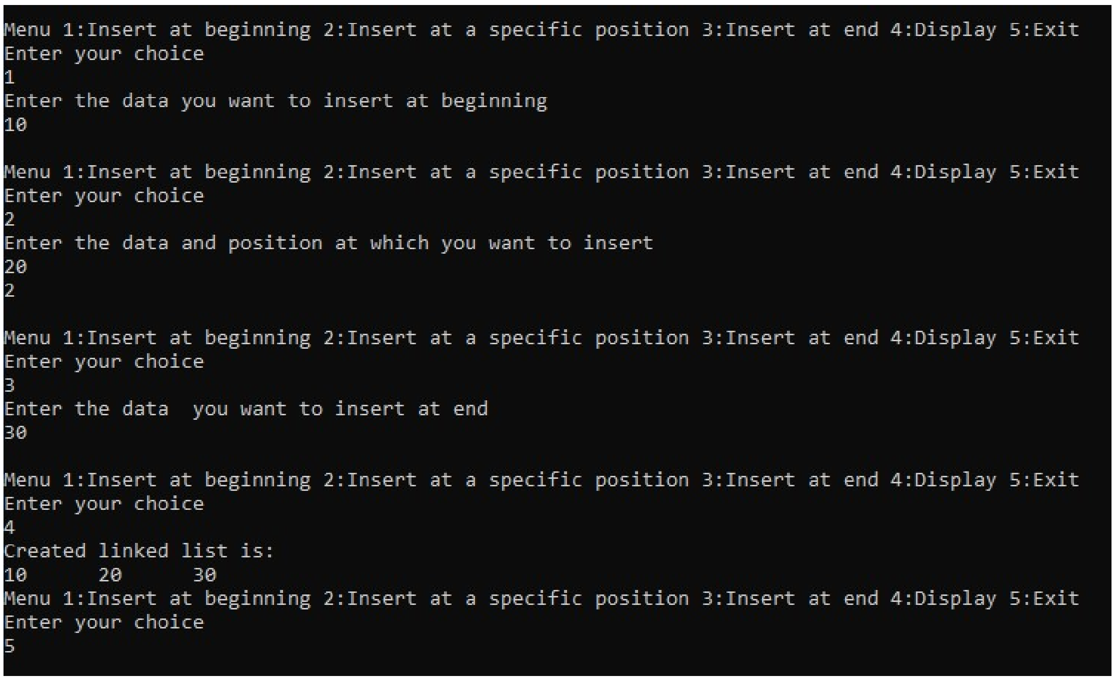
1)INSERT

```
#include <stdio.h>

#include <stdlib.h>

struct node{

   int data;

   struct node* next;

};

void insertAtBeginning(struct node** head ,int val){

   struct node* newnode=(struct node*)malloc(sizeof(struct node));

   newnode->data=val;

   newnode->next=*head;

   *head=newnode;

}

void insertAtEnd(struct node** head,int val){

   struct node* newnode=(struct node*)malloc(sizeof(struct node));

   struct node* temp=*head;

   newnode ->data=val;

   newnode->next=NULL;

   if(*head==NULL){

      *head=newnode;

      return;

   }

   while(temp->next != NULL){

      temp=temp->next;
   }
   temp->next=newnode;
}
void insertAtPosition(struct node** head,int val,int pos){
    if(pos<=0){
      printf("Invalid position\n");
      return;
    }
    if(pos==1 || *head==NULL){
      insertAtBeginning(head,val);
      return;
    }
    struct node* newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=val;
    struct node* temp=*head;
    int count=1;

    while(count<pos-1 && temp->next !=NULL){
      temp=temp->next;
      count++;
    }

    if(count<pos-1){
      printf("Invalid Position\n");
      return;
    }
```

```
     newnode->next=temp->next;

     temp->next=newnode;

}


void display(struct node* head){

    struct node* temp=head;


    if(temp==NULL){

        printf("Linked List is Empty");

        return;

    }

    while(temp!=NULL){

        printf("%d\t",temp->data);

        temp=temp->next;

    }

}


int main()

{

    int ch,new,pos;

    struct node* head=NULL;

    while(ch!=5)

    {

    printf("Menu 1:Insert at beginning 2:Insert at a specific position 3:Insert at end 4:Display
5:Exit\n");
```

```
printf("Enter your choice\n");

scanf("%d",&ch);

switch(ch)

{

   case 1:

   {

   printf("Enter the data you want to insert at beginning\n");

   scanf("%d",&new);

   insertAtBeginning(&head,new);

   break;

   }

   case 2:

   {

   printf("Enter the data and position at which you want to insert \n");

   scanf("%d%d",&new,&pos);

   insertAtPosition(&head,new,pos);

   break;

   }

   case 3:

   {

   printf("Enter the data  you want to insert at end\n");

   scanf("%d",&new);

   insertAtEnd(&head,new);

   break;

   }

   case 4:
        {
           printf("Created linked list is:\n");
           display(head);
           break;
        }
        case 5:
        {
           return 0;
           break;
        }
        case 6:
        {
           printf("Invalid data!");
           break;
        }
        }
}
return 0;
}
```

OUTPUT :


```
Menu 1:Insert at beginning 2:Insert at a specific position 3:Insert at end 4:Display 5:Exit
Enter your choice
1
Enter the data you want to insert at beginning
10

Menu 1:Insert at beginning 2:Insert at a specific position 3:Insert at end 4:Display 5:Exit
Enter your choice
2
Enter the data and position at which you want to insert
20
2

Menu 1:Insert at beginning 2:Insert at a specific position 3:Insert at end 4:Display 5:Exit
Enter your choice
3
Enter the data  you want to insert at end
30

Menu 1:Insert at beginning 2:Insert at a specific position 3:Insert at end 4:Display 5:Exit
Enter your choice
4
Created linked list is:
10      20      30
Menu 1:Insert at beginning 2:Insert at a specific position 3:Insert at end 4:Display 5:Exit
Enter your choice
5
```

2)DELETE

```
#include <stdio.h>

#include<stdlib.h>



typedef struct Node {

   int data;

   struct Node *next;

}Node;


void InsertAtBeginning( Node **head_ref,int new_data);

void DeleteAtBeginning( Node **head_ref);

void DeleteAtEnd( Node **head_ref);
void Delete( Node **prev_node,int pos);
void PrintList(Node * next);


void InsertAtBeginning( Node **head_ref,int new_data)
{
   Node *new_node=(struct Node*)malloc(sizeof( Node));
   new_node->data=new_data;
   new_node->next=*head_ref;
   *head_ref=new_node;
}


void DeleteAtBeginning( Node **head_ref)
{
   Node *ptr;
if(head_ref == NULL)
{
printf("\nList is empty");
}
else
{
ptr = *head_ref;
*head_ref = ptr->next;
free(ptr);
printf("\n Node deleted from the beginning ...");


}


}
```

```
void DeleteAtEnd(Node **head_ref)

{

   Node *ptr,*ptr1;


if(*head_ref == NULL)


{


printf("\nlist is empty");


}


else if((*head_ref)-> next == NULL)


{


free(*head_ref);


*head_ref= NULL;


printf("\nOnly node of the list deleted ...");


}


else
```

```
{

ptr = *head_ref;

while(ptr->next != NULL)

{

ptr1 = ptr;

ptr = ptr ->next;

}

ptr1->next = NULL;

free(ptr);

printf("\n Deleted Node from the last ...");

}


}
void Delete(Node **head_ref, int pos)
{
   Node *temp = *head_ref, *prev;


   if (temp == NULL)
```

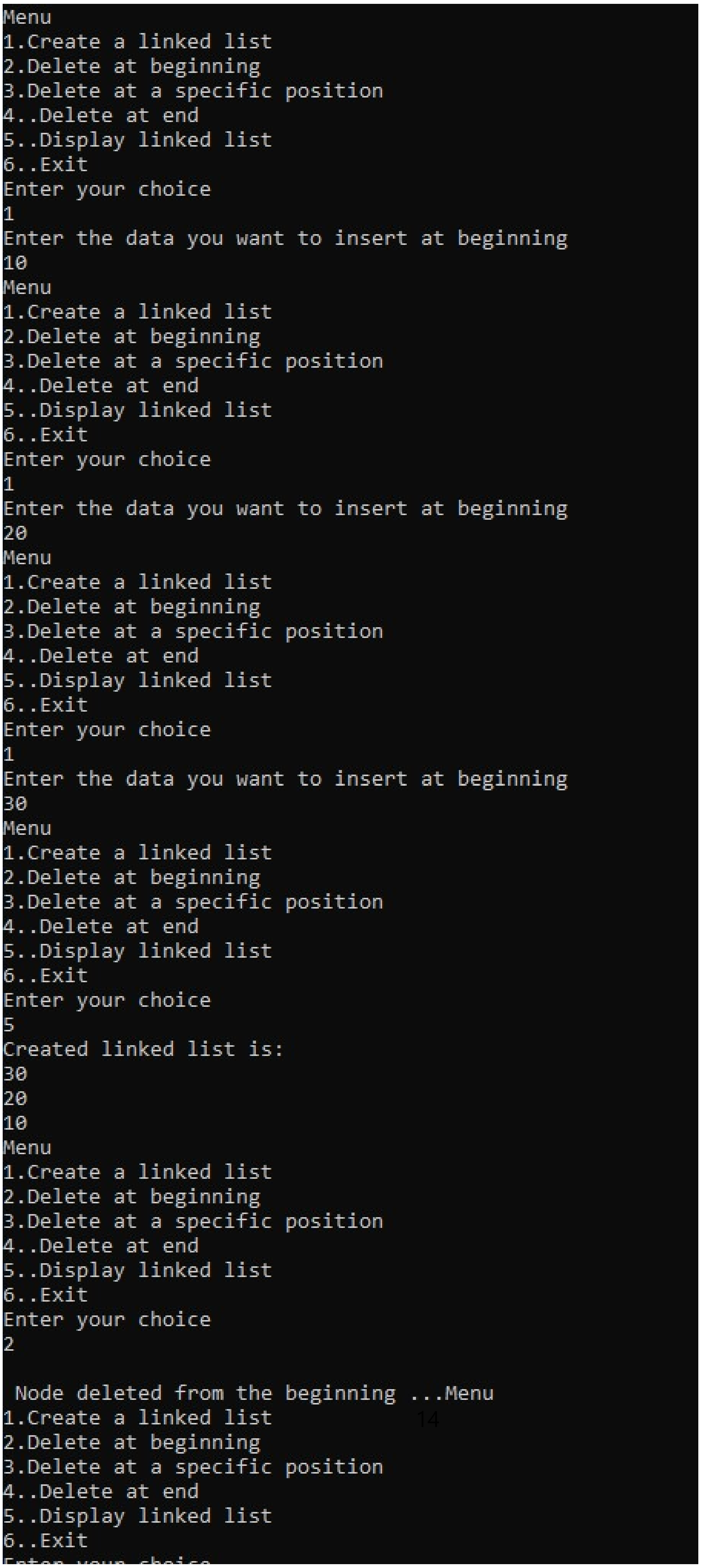
```
{
   printf("\nList is empty");
   return;
}

if (pos == 1)
{
   *head_ref = temp->next;
   free(temp);
   printf("\nDeleted node with position %d", pos);
   return;
}

for (int i = 0; temp != NULL && i < pos - 1; i++)
{
   prev = temp;
   temp = temp->next;
}

if (temp == NULL)
{
   printf("\nPosition out of range");
   return;
}

prev->next = temp->next;
free(temp);
printf("\nDeleted node with position %d", pos);
```

```
}
void PrintList(Node *node)
{
   while (node!=NULL)
   {
      printf("%d\n",node->data);
      node=node->next;
   }
}


int main()
{
   int ch,new,pos;
   Node* head=NULL;
   while(ch!=6)
   {
   printf("Menu\n");
   printf("1.Create a linked list\n");
   printf("2.Delete at beginning\n");
   printf("3.Delete at a specific position\n");
   printf("4..Delete at end\n");
   printf("5..Display linked list\n");
   printf("6..Exit\n");
   printf("Enter your choice\n");
   scanf("%d",&ch);
   switch(ch)
   {
```

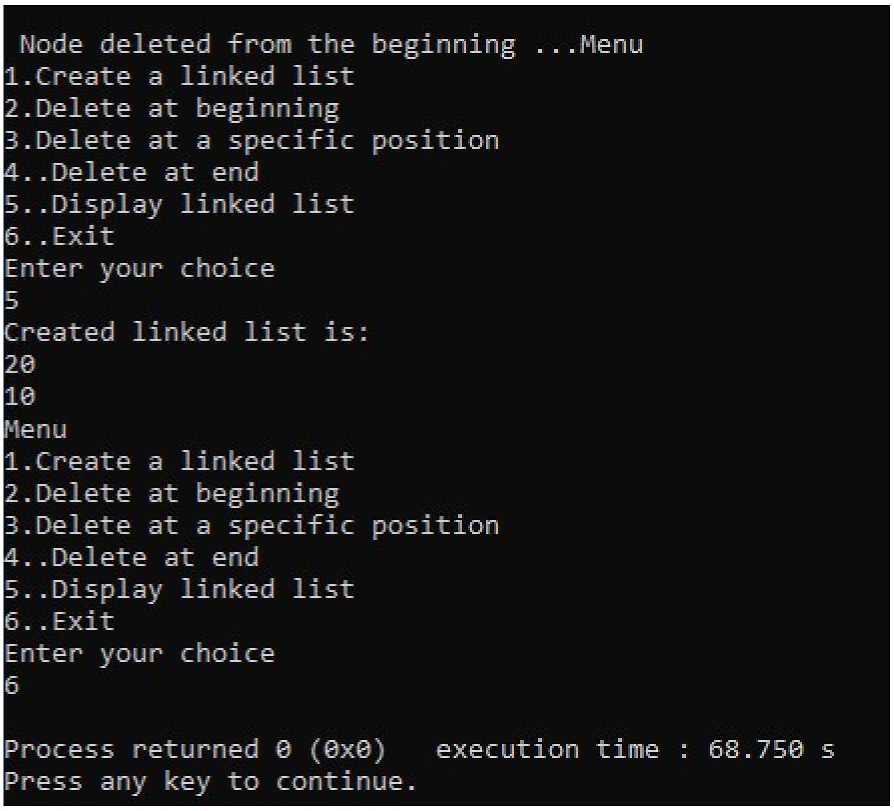
```
case 1:
{
printf("Enter the data you want to insert at beginning\n");
scanf("%d",&new);
InsertAtBeginning(&head,new);
break;
}
case 2:
{
DeleteAtBeginning(&head);
break;
}
case 3:
{
printf("Enter the position at which you want to delete \n");
scanf("%d",&pos);
Delete(&head,pos);
break;
}
case 4:
{
DeleteAtEnd(&head);
break;
}
case 5:
{
   printf("Created linked list is:\n");
   PrintList(head);
```

```
            break;
        }
        case 6:
        {
            return 0;
            break;
        }
        default:
        {
            printf("Invalid data!");
            break;
        }
        }
}
return 0;
}
```

OUTPUT:



```
Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
1
Enter the data you want to insert at beginning
10
Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
1
Enter the data you want to insert at beginning
20
Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
1
Enter the data you want to insert at beginning
30
Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
5
Created linked list is:
30
20
10
Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
2

 Node deleted from the beginning ...Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
```



```
 Node deleted from the beginning ...Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
5
Created linked list is:
20
10
Menu
1.Create a linked list
2.Delete at beginning
3.Delete at a specific position
4..Delete at end
5..Display linked list
6..Exit
Enter your choice
6

Process returned 0 (0x0)   execution time : 68.750 s
Press any key to continue.
```