22/01/24

Q5. Write a program to implement a Singly Linked List with following operations.

a) Create a linked list.
b) Insertion of a node at first position, at any position and at end of the list.
Display the contents of the linked list

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
 int data;
 struct node *next;
};
void printData(struct node *head)
{
  if (head == NULL)
  {
    printf("The list is empty");
  }
  else
  {
    struct node *ptr = head;
    while (ptr != NULL)
    {
       printf("%d\n", ptr -> data);
       ptr = ptr -> next;
    }
  }
}
void insertBeg(struct node **head, int value)
{
  struct node *temp = (struct node *) malloc (sizeof (struct node));
```

```
    temp->data = value;
    temp->next = *head;
    *head = temp;
}
void insertEnd (struct node*head, int value)
{
    struct node *ptr = head;
    struct node *temp=(struct node *)malloc(sizeof(struct node));
    temp -> data = value;
    temp -> next = NULL;
    while (ptr -> next != NULL)
    {
        ptr = ptr -> next;
    }
    ptr -> next = temp;
}
Void insertAtPos(struct node *head, int value, int pos)
{
    struct node *ptr, *ptr2;
    struct node *temp = (struct node *) malloc(sizeof(struct node));
    temp -> data = value;
    temp -> next = NULL;
    int position = pos;
    ptr = head;
    while (pos != 1)
    {
        ptr2 = ptr;
        ptr = ptr -> next;
        pos--;
    }
    temp -> next = ptr2 -> next;
    ptr2 -> next = temp;
    printf(" value %d added successful at %d \n", value, position);
}
```

```
int main()
{
  struct node *head = NULL;
  insertBeg (&head, 34);
  printData (head);
  printf("- - - - - - - - - - - - - - - \n");
  insertEnd (head, 75);
  insertEnd ( head, 56);
  insert End ( head, 87);
  printData (head);
  printf(" - - - - - - - - - - - - - - \n");
  insertAtPos(head, 89,3);
  printData (head);
}
```

O/P

34

- - - - - - - - - - - - - - - - -

34

75

56

87

- - - - - - - - - - - - - - -

Value 89 added successful at 3

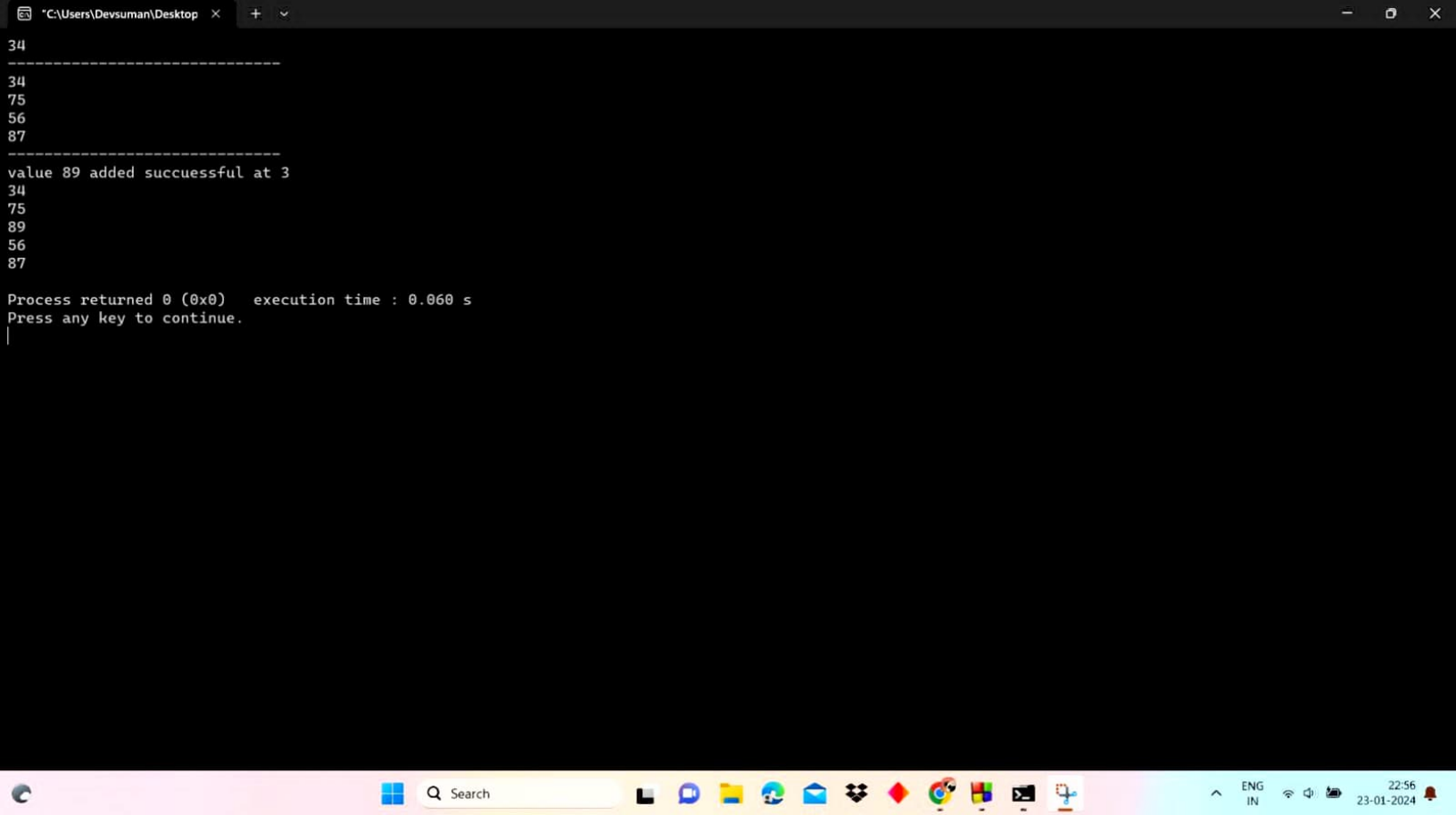
34

75

89

56

87


"C:\Users\Devsuman\Desktop
34
34
75
56
87
value 89 added succuessful at 3
34
75
89
56
87
Process returned 0 (0x0) execution time : 0.060 s
Press any key to continue.
Search
ENG
IN
22:56
23-01-2024

Q6. Write a program to implement singly linked list with following operations.

a). Create linked list

b). deletion of first element, specified element and last element in the list.

Display the contents of the linked list.

```
#include <stdio.h>
struct node
{
  int data;
  struct node *next;
};
struct node *head = NULL, *newnode, *temp;
void create()
{
   int i, n;
   printf("Enter the number of elements : \n");
   scanf("%d", &n);
   for(i=0; i<n; i++)
   {
      newnode = (struct node*) malloc(sizeof(struct node));
      printf("Enter the element %d : \n", i+1);
      scanf("%d", &newnode->data);
      newnode -> next = NULL;
      if(head == NULL)
      {
        temp = head = newnode;
      }
      else
      {
        temp -> next = newnode;
        temp = newnode;
      }
   }
}
```

```
void display()
{
  temp= head;
  printf("The elements are :\n");
  while (temp!= NULL)
  {
    printf("%d\n",tem->data);
    temp=temp->next;
  }
}
void delete_beg()
{
  temp=head;
  if (head == NULL)
  {
    printf("List is empty\n");
  }
  else
  {
    head=temp->next;
    free(temp);
  }
}
void delete_end()
{
  temp=head;
  struct node* prenode;
  while (temp->next!=NULL)
  {
    prenode=temp;
    temp=temp->next;
  }
  if(temp==head)
  {
```

```
    head = NULL;
  }
  else
  {
    prenode -> next = NULL;
  }
  free(temp);
}
void delete_pos()
{
  struct node * nextnode;
  int pos, i = 1;
  printf(" Enter the position: \n");
  scanf("%d", & pos);
  temp = head;
  while(i < pos - 1)
  {
    temp = temp -> next;
    i++;
  }
  next node = temp -> next;
  temp -> next = nextnode -> next;
  free(nextnode);
}
Void main()
{
  int choice;
  while(1)
  {
    printf("Enter operation: \n 1. Create \n 2. Display \n 3. delete at
    beginning \n 4. delete at ~~ed~~ end \n 5. delete at position \n
    6. -1 to end \n");
    scanf("%d", & choice);
    if( choice == -1)
    {
```

```
        printf("Operation completed\n");
        break;
    }
    else
    {
      switch(choice)
      {
        case 1: create();
                break;
        case 2: display();
                break;
        case 3: delete beg();
                break;
        case 4: delete end();
                break;
        case 5: delete pos();
                break;
        default: printf("Invalid output\n");
      }
    }
  }
}
```

O/P

```
Enter operation
1. Create
2. Display
3. delete at beginning
4. delete at end
5. delete at position
6. -1 to end
1
Enter the number of elements
5
Enter the number 1
10
Enter the number 2
20
Enter the number 3
30
Enter the number 4
40
Enter the number 5
50
The elements are:
10
20
30
40
50
```

```
Enter operation:
1.create
2.display
3.delete at beginnning
4.delete at end
5.delete at position
6.-1 to end
1
enter the number of elements:
5
Enter the element 1:
10
Enter the element 2:
20
Enter the element 3:
30
Enter the element 4:
40
Enter the element 5:
50
Enter operation:
1.create
2.display
3.delete at beginnning
4.delete at end
5.delete at position
6.-1 to end
2
The elements are:
10
20
30
40
50
Enter operation:
1.create
2.display
3.delete at beginnning
4.delete at end
5.delete at position
6.-1 to end
3
Enter operation:
1.create
2.display
3.delete at beginnning
4.delete at end
5.delete at position
6.-1 to end
4
Enter operation:
1.create
2.display
3.delete at beginnning
4.delete at end
5.delete at position
6.-1 to end
5
enter the position:
2
Enter operation:
1.create
2.display
3.delete at beginnning
4.delete at end
5.delete at position
6.-1 to end
-1
operation completed!

Process returned 0 (0x0)   execution time : 37.571 s
Press any key to continue.
```