Graphic Era
HILL UNIVERSITY
Established by an Act of the State Legislature of Uttarakhand (Adhiniyam Sankhya 12 of 2011)

Mid-Term work

on

Data Structures with C

(PCS 307)

2021-22

Submitted to:

Dr. Rakesh Patra

Professor
GEHU, D.Dun

Submitted by:

Achintya Mishra

University Roll. No.: 2018091
Class Roll. No./Section: 36/B

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GRAPHIC ERA HILL UNIVERSITY, DEHRADUN

ACKNOWLEDGMENT

I would like to particularly thank my Data Science using DSA Lab Faculty **Dr. Rakesh Patra** for his patience, support and encouragement throughout the completion of this Term work.

At last but not the least I greatly indebted to all other persons who directly or indirectly helped me during this course.

Photo

Achintya Mishra

University. Roll No.- 2018091

B.Tech CSE-A-III Sem

Session: 2021-22

GEHU, Dehradun

Contents

Q11. Write a C program to Evaluate Postfix Expression using Stack

Q 12. Write a C program to create TWO singly linked list L1 and L2 and sort both the list and finally merge both the list such that L2 comes after L1.[use double pointer]

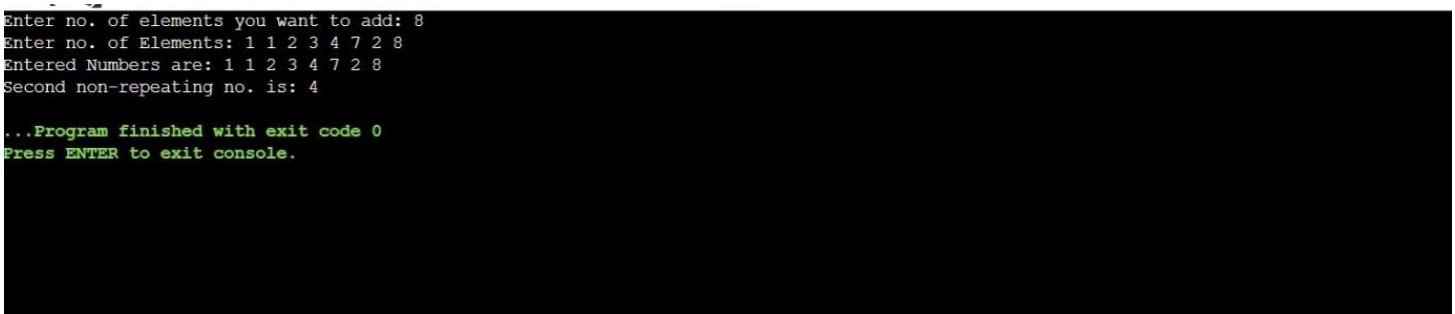
Q 13. Write C program to create a doubly link list by adding the node right hand side and then check list is in palindrome form or not.

Q14. Write a C program to create a circular link list by adding the nodes in right hand side and then print the list.

Q1. Write a the C program to create an array by inserting N elements in it then find second non repeating element from the array.

```
#include <stdio.h>
int check(int arr[],int n)
{
   int k=0,temp;
   for(int i=0;i<n;i++)
   {
      int j;
      for(j=0;j<n;j++)
      {
         if(arr[i]==arr[j] && i!=j)
         {
            break;
         }
      }
      if(j==n)
      {
         temp=arr[i];
         k++;
         {
            if(k==2)
            {
               printf("\nSecond non-repeating no. is: %d ",temp);
               return 0;
            }
         }
      }
   }
}
int main()
{
   int n;
```

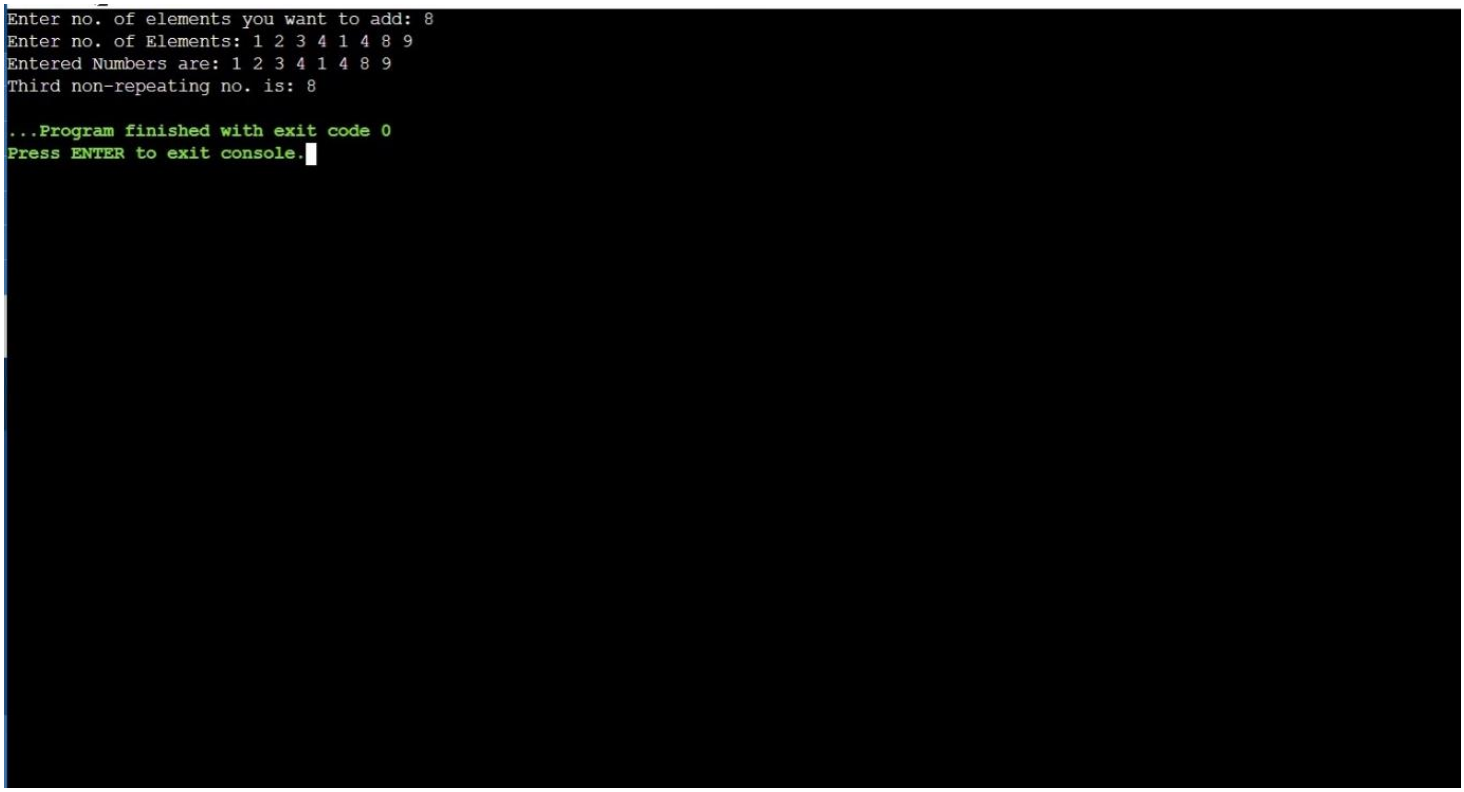
```
    printf("Enter no. of elements you want to add: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter no. of Elements: ");
    for(int i=0;i<n;i++)
    {
       scanf("%d",&arr[i]);
    }
    printf("Entered Numbers are: ");
    for(int i=0;i<n;i++)
    {
       printf("%d ",arr[i]);
    }
    check(arr,n);
}
```



Q2. Write a the C program to create an array by inserting N elements in it then find third repeating element from the array.

```
#include <stdio.h>
int check(int arr[],int n)
{
   int k=0,temp;
   for(int i=0;i<n;i++)
   {
      int j;
      for(j=0;j<n;j++)
      {
         if(arr[i]==arr[j] && i!=j)
         {
            break;
         }
      }
      if(j==n)
      {
         temp=arr[i];
         k++;
         {
            if(k==3)
            {
               printf("\nThird non-repeating no. is: %d ",temp);
               return 0;
            }
         }
      }
   }
}
int main()
{
   int n;
   printf("Enter no. of elements you want to add: ");
   scanf("%d",&n);
   int arr[n];
   printf("Enter no. of Elements: ");
   for(int i=0;i<n;i++)
   {
```

```
        scanf("%d",&arr[i]);
    }
    printf("Entered Numbers are: ");
    for(int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    check(arr,n);
}
```



Q3. Write a C program Create a Dynamic array and then Reverse the array using recursion and then finally print the array.

```
#include<stdio.h>
int rev(int arr[],int start,int end)
```

```
{
   int temp;
   if(start>=end)
      return 1;
   temp=arr[start];
   arr[start]=arr[end];
   arr[end]=temp;
   rev(arr,start+1,end-1);

}
int disp(int arr[], int n)
{
   printf("\nElements after swaping are: ");
   for(int i=0;i<n;i++)
   {
      printf("%d ",arr[i]);
   }
}

int main()
{
   int n;
   printf("Enter No. of Element you want to Add: ");
   scanf("%d",&n);
   int arr[n];
   printf("Enter Elements: ");
   for(int i=0;i<n;i++)
   {
      scanf("%d",&arr[i]);
   }
   printf("Elements before swaping are: ");
   for(int i=0;i<n;i++)
   {
      printf("%d ",arr[i]);
   }
   rev(arr,0,n-1);
   disp(arr,n);
```

```
}
```

```
Enter No. of Element you want to Add: 4
Enter Elements: 1 2 3 4
Elements before swaping are: 1 2 3 4
Elements after swaping are: 4 3 2 1

...Program finished with exit code 0
Press ENTER to exit console.
```

Q4. Write a C Program implement STACK using array in menu driven form.

```
#include<stdio.h>
#include<stdlib.h>
#define size 5
```

```
int Insert(int *arr,int top)
{
   if(top==size-1)
   {
      printf("STACK IS FULL \n");
   }
   else
   {
      int num;
      printf("Enter the value: \n");
      scanf("%d",&num);
      top++;
      arr=(int *)realloc(arr,sizeof(int)*(top+1));
      arr[top]=num;


      printf("ADDED %d SUCCESSFULLY\n",arr[top]);

      return top;
   }
}
int Delete(int *arr,int top)
{
   if(top==-1)
   {
      printf("STACK IS EMPTY\n");
   }
   else{

      printf("DELETED %d SUCCESSFULLY\n",arr[top]);

      top--;
      arr=(int *)realloc(arr,sizeof(int)*(top+1));
      return top;
```

```
    }
}

void Display(int *arr,int top)
{
    if(top==-1)
    {
        printf("STACK IS EMPTY\n");
    }
    else
    {
        printf("Stack is \n");
        for(int i=top;i>=0;i--)
        {
            printf("----------\n");
            printf("|   %d   |\n",arr[i]);
            printf("----------\n");
        }

    }
}



int main()
{
    int *stack,top=-1,counter=0,choice;

    stack=(int*)malloc(sizeof(int )*(top+1));

    while(counter==0)
    {
        printf("Enter\n");
        printf("1.INSERT  \n2.DELETE \n3.DISPLAY \n4.EXIT \n");
        printf("Enter your choice: ");
        scanf("%d",&choice);
```

```
        switch(choice)
        {
            case 1:
                top=Insert(stack,top);
                break;

            case 2:
                top=Delete(stack,top);
                break;

            case 3:
                Display(stack,top);
                break;

            case 4:
                printf("EXITED \n");
                counter++;
                break;

            default:
                printf("WRONG CHOICE \n");
        }
    }


    return 0;
}
```

```
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter the value:
7
ADDED 7 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter the value:
8
ADDED 8 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter the value:
5
ADDED 5 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
```

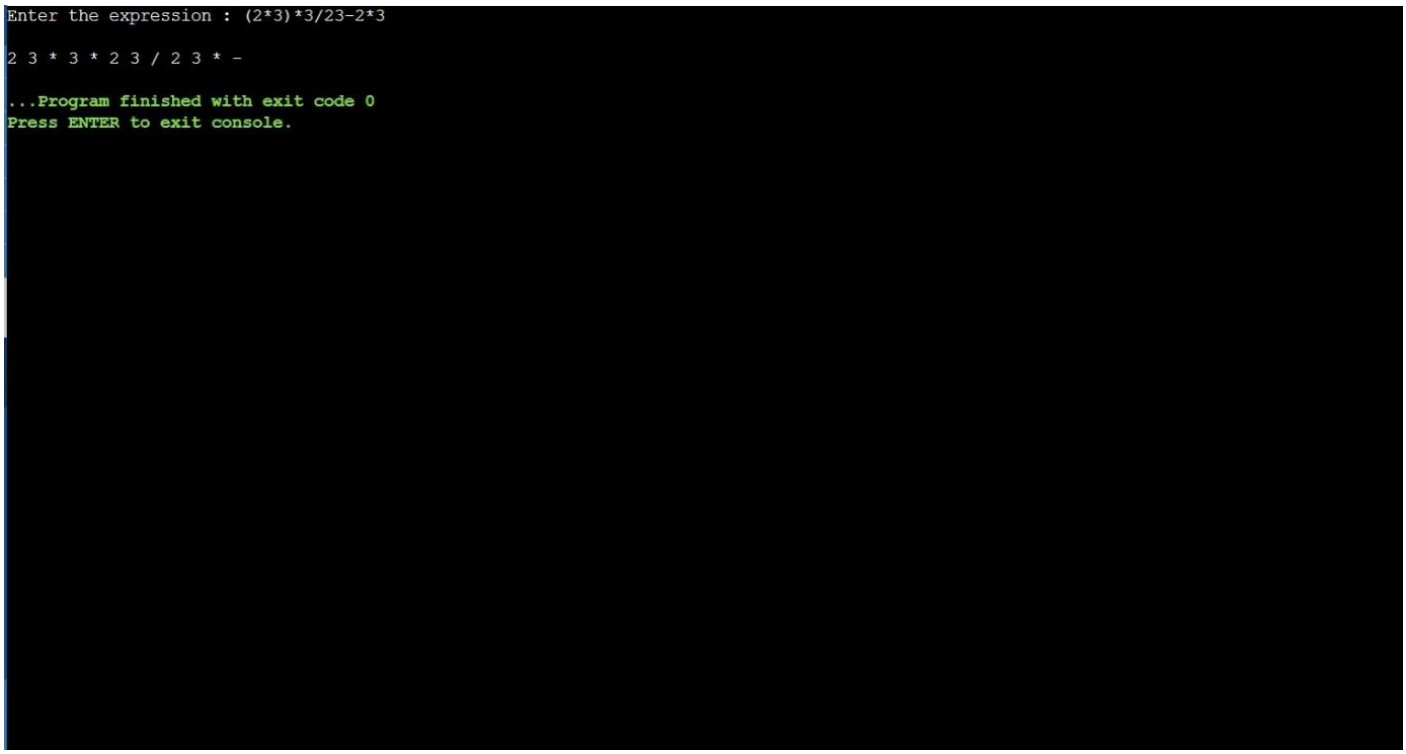
```
Enter your choice: 3
Stack is
----------
|   5   |
----------
----------
|   8   |
----------
----------
|   7   |
----------
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 2
DELETED 5 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
Stack is
----------
|   8   |
----------
----------
|   7   |
----------
Enter
1.INSERT
2.DELETE
```

```
----------
|   7   |
----------
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 2
DELETED 5 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
Stack is
----------
|   8   |
----------
----------
|   7   |
----------
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 4
EXITED


...Program finished with exit code 0
Press ENTER to exit console.
```

#include<stdio.h>
#include<ctype.h>

```
void push(char stack[],int *top,char x)
{
   *top=*top+1;
   stack[*top] = x;
}

char pop(char stack[],int *top)
{
   if(*top == -1)
      return -1;
   else
   {
      char temp= stack[*top];
      *top=*top-1;
      return temp;
   }
}

int priority(char x)
{
   if(x == '(')
      return 0;
   if(x == '+' || x == '-')
      return 1;
   if(x == '*' || x == '/')
      return 2;
   if(x == '^')
      return 3;
   return 0;
}

int main()
{
   char stack[100];
   int top = -1;
   char exp[100];
   char *e, x;
   printf("Enter the expression : ");
```

```
    scanf("%s",exp);
    printf("\n");
    e = exp;

    while(*e !='\0')
    {
       if(isalnum(*e))
          printf("%c ",*e);
       else if(*e == '(')
          push(stack,&top,*e);
       else if(*e == ')')
       {
          while((x = pop(stack,&top)) != '(')
             printf("%c ", x);
       }
       else
       {
          while(priority(stack[top]) >= priority(*e))

             printf("%c ",pop(stack,&top));
          push(stack,&top,*e);
       }
       e++;
    }

    while(top != -1)
    {
       printf("%c ",pop(stack,&top));
    }return 0;
}
```


Enter the expression : (2*3)*3/23-2*3
2 3 * 3 * 2 3 / 2 3 * -
...Program finished with exit code 0
Press ENTER to exit console.

Q6. Write a C Program to create singly linked list by adding nodes in the right hand side and delete alternate node from the list and then print the final list.

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

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

void insertAtRightSide(node **,int);
void deleteAtAlternate(node*);
void display(node *head);
int main()
{
   node *head;
   int counter=0,choice,value,pos;

   head=NULL;
   while(counter==0)
   {
      printf("Enter\n");
      printf("1.INSERT AT RIGHT \n2.DELETE ALTERNATE NODES \n3.DISPLAY \n4.EXIT\n");
      printf("Enter your choice: ");
      scanf("%d",&choice);

      switch(choice)
      {
         case 1:
            printf("Enter a value : ");
            scanf("%d",&value);
```

```
            insertAtRightSide(&head,value);
            break;

        case 2:
            deleteAtAlternate(head);
            display(head);
            break;

        case 3:
            display(head);
            break;

        case 4:
            printf("EXITED\n");
            counter++;
            break;

        default:
            printf("WRONG CHOICE\n");
            break;
      }
   }
   return 0;
}
void insertAtRightSide(node **head,int value)
{
   node *temp=(node*)malloc(sizeof(node));
   temp->data=value;
   if(*head==NULL)
   {
      temp->next=*head;
      *head=temp;
   }
   else
   {
      node *temp2=*head;
      while(temp2->next!=NULL){
```

```
            temp2=temp2->next;
        }
        temp2->next=temp;
        temp->next=NULL;
    }

}
void display(node *head)
{
    node *temp;
    temp=head;
    printf("List is : ");
    while(temp!=NULL)
    {
        printf(" %d ",temp->data);
        temp=temp->next;
    }
    printf("\n");
}

void deleteAtAlternate(node* head)
{
    if (head == NULL)
        return;

    node *temp=head->next;
    if(temp==NULL)
        return;

    head->next=temp->next;
    free(temp);
    deleteAtAlternate(head->next);
}
```

```
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 5
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 8
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 7
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 9
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 3
```

```
Enter your choice: 1
Enter a value : 7
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 9
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 3
List is :  5  8  7  9
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 2
List is :  5  7
Enter
1.INSERT AT RIGHT
2.DELETE ALTERNATE NODES
3.DISPLAY
4.EXIT
Enter your choice: 4
EXITED


...Program finished with exit code 0
Press ENTER to exit console.
```

Q7. Write a C Program implement STACK using Link List in menu driven form.

```
#include<stdio.h>
#include<stdlib.h>
#define max 5
struct Node
{
   int data;
   struct Node *next;

};
typedef struct Node node;

void insert(node **,int,int*);
void delete(node**,int*);
void display(node *top);
int main()
{
   node *top;
   int counter=0,choice,value,pos,sizeRef=-1;

   top=NULL;
   while(counter==0)
   {
      printf("Enter\n");
      printf("1.INSERT  \n2.DELETE  \n3.DISPLAY \n4.EXIT\n");
      printf("Enter your choice: ");
      scanf("%d",&choice);

      switch(choice)
      {
         case 1:
            if(sizeRef==max-1)
            {
               printf("Stack is Full\n");
```

```
            }
            else{
                printf("Enter a value : ");
                scanf("%d",&value);
                insert(&top,value,&sizeRef);
            }
            break;

        case 2:
            if(sizeRef==-1)
            {
                printf("Stack is Empty\n");
            }
            else{
                delete(&top,&sizeRef);
            }
            break;

        case 3:
            display(top);
            break;

        case 4:
            printf("EXITED\n");
            counter++;
            break;

        default:
            printf("WRONG CHOICE\n");
            break;
      }
   }
   return 0;
}

void insert(node **top,int value,int *sizeRef)
{
```

```
    *sizeRef=*sizeRef+1;
    node *temp=(node*)malloc(sizeof(node));
    temp->data=value;
    temp->next=*top;
    *top=temp;

}

void display(node *top)
{
    node *temp;
    temp=top;
    printf("List is : ");
    while(temp!=NULL)
    {
        printf(" %d ",temp->data);
        temp=temp->next;
    }
    printf("\n");
}

void delete(node**top,int *sizeRef)
{
    if (top == NULL)
        return;

    *sizeRef=*sizeRef-1;
    node *temp=*top;
    printf("%d Deleted Successfully\n",temp->data);
    *top=temp->next;
    free(temp);
}
```

```
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 8
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
7Enter a value :
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 6
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
List is :  6  9  7  8
```

```
4.EXIT
Enter your choice: 1
Enter a value : 9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 6
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
List is :  6  9  7  8
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 2
6 Deleted Successfully
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 4
EXITED


...Program finished with exit code 0
Press ENTER to exit console.
```

Q8. Write a C Program implement QUEUE using Link List in menu driven form.

```
#include<stdio.h>
#include<stdlib.h>
#define max 5
struct Node
{
   int data;
   struct Node *next;

};
typedef struct Node node;

void insert(node **,int*,int*,int);
void delete_node(node**,int*,int*);
void display(node *top);
int main()
{
   node *top;
   int counter=0,choice,value,pos,front=-1,rear=-1;

   top=NULL;
   while(counter==0)
   {
      printf("Enter\n");
      printf("1.INSERT  \n2.DELETE  \n3.DISPLAY \n4.EXIT\n");
      printf("Enter your choice: ");
      scanf("%d",&choice);

      switch(choice)
      {
         case 1:
            if(rear==max-1)
            {
               printf("Queue is Full\n");
            }
            else{
```

```
            printf("Enter a value : ");
            scanf("%d",&value);
            insert(&top,&front,&rear,value);
        }
        break;

    case 2:
        if(front==-1)
        {
            printf("Queue is Empty\n");
        }
        else{
            delete_node(&top,&front,&rear);
        }
        break;

    case 3:
        display(top);
        break;

    case 4:
        printf("EXITED\n");
        counter++;
        break;

    default:
        printf("WRONG CHOICE\n");
        break;
    }
  }
  return 0;
}

void insert(node **top,int* front,int* rear,int value )
{
  if(*front==-1)
    *front=*front+1;
```

```
    node *temp=(node*)malloc(sizeof(node));
    temp->data=value;
    if(*top==NULL)
    {
        temp->next=*top;
        *top=temp;
    }
    else
    {
        node *temp2=*top;
        while(temp2->next!=NULL)
        {
            temp2=temp2->next;
        }
        temp2->next=temp;
        temp->next=NULL;
    }
    *rear=*rear+1;
}

void display(node *top)
{
    node *temp;
    temp=top;
    printf("List is : ");
    while(temp!=NULL)
    {
        printf(" %d ",temp->data);
        temp=temp->next;
    }
    printf("\n");
}

void delete_node(node**top,int* front,int* rear)
{
    if (top == NULL)
```

```
        return;

    *front=*front+1;
    node *temp=*top;
    printf("%d Deleted Successfully\n",temp->data);
    *top=temp->next;
    free(temp);
    if(*front>*rear)
        *front=*rear=-1;
}
```

```
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 4
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 7
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 8
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
List is :  4  7  8  9
Enter
1.INSERT
```

```
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value : 9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
List is :  4  7  8  9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 2
4 Deleted Successfully
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
List is :  7  8  9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 4
EXITED


...Program finished with exit code 0
Press ENTER to exit console.
```

Q9. Write a C Program implement priority QUEUE using array in menu driven form.

```
#include <stdio.h>
#include <stdlib.h>
struct priorityqueue
{
   int r;
   int *num;
   int *priority;
   int size;
};
int isempty(struct priorityqueue *pq)
{
   if (pq->r == -1)
   {
      return 1;
   }
   else
   {
      return 0;
   }
}
int isfull(struct priorityqueue *pq)
{
   if (pq->r == pq->size - 1)
   {
      return 1;
   }
   else
   {
      return 0;
   }
}
void enqueue(struct priorityqueue *pq, int val, int p)
```

```
{
   if (isfull(pq))
   {
      printf("The queue is full\n");
   }
   else
   {
      pq->r++;
      pq->num[pq->r]=val;
      pq->priority[pq->r] = p;
   }
}
int gethighestpriority(struct priorityqueue *pq)
{
   int i, p;
   p = -1;
   if (!(isempty(pq)))
   {
      for (i = 0; i <= pq->r; i++)
      {
         if (pq->priority[i] > p)
         {
            p = pq->priority[i];
         }
      }
   }
   return p;
}
int deletehighestpriority(struct priorityqueue *pq)
{
   int i, j, x, a;
   x = gethighestpriority(pq);
   for (i = 0; i <= pq->r; i++)
   {
      if (pq->priority[i] == x)
      {
         a = pq->num[i];
```

```
            break;
        }
    }
    if (i < pq->r)
    {
        for (j = i; j < pq->r; j++)
        {
            pq->num[j] = pq->num[j + 1];
            pq->priority[j] = pq->priority[j + 1];
        }
    }
    pq->r--;
    return a;
}
void display(struct priorityqueue *pq)
{
    int i;
    printf("Priority queue: \n");
    for (i = 0; i <= pq->r; i++)
    {
        printf("Element: %d Priority: %d\n", pq->num[i], pq-
>priority[i]);
    }
}

int main()
{
    struct priorityqueue pq;
    printf("Enter the size of your priority queue: ");
    scanf("%d", &pq.size);
    pq.r = -1;
    pq.num = (int *)malloc(pq.size * sizeof(int));
    pq.priority = (int *)malloc(pq.size * sizeof(int));
    int ch, val, p;
    do
    {
```

```
        printf("Press 1 to insert\nPress 2 to get highest priority\nPress 3
to delete\nPress 4 to display\nPress 5 to exit\n");
        printf("Enter your choice\n");
        scanf("%d", &ch);
        switch (ch)
        {
            case 1:
                printf("Enter element to insert: \n");
                scanf("%d", &val);
                printf("Enter priority: \n");
                scanf("%d", &p);
                enqueue(&pq, val, p);
                break;
            case 2:
                if (isempty(&pq))
                {
                    printf("The queue is empty\n");
                }
                else
                {
                    p = gethighestpriority(&pq);
                    printf("The highest priority = %d\n", p);
                }
                break;
            case 3:
                if (isempty(&pq))
                {
                    printf("Queue is empty\n");
                }
                else
                {
                    val = deletehighestpriority(&pq);
                    printf("%d is deleted\n", val);
                }
                break;
            case 4:
                display(&pq);
```

```
            break;

        default:
            break;
      }
   } while (ch != 5);
}
```

```
Enter the size of your priority queue: 5
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
1
Enter element to insert:
8
Enter priority:
3
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
1
Enter element to insert:
7
Enter priority:
6
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
1
Enter element to insert:
8
Enter priority:
4
Press 1 to insert
```

```
Press 4 to display
Press 5 to exit
Enter your choice
1
Enter element to insert:
8
Enter priority:
4
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
4
Priority queue:
Element: 8 Priority: 3
Element: 7 Priority: 6
Element: 8 Priority: 4
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
2
The highest priority = 6
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
3
7 is deleted
```

```
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
2
The highest priority = 6
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
3
7 is deleted
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
4
Priority queue:
Element: 8 Priority: 3
Element: 8 Priority: 4
Press 1 to insert
Press 2 to get highest priority
Press 3 to delete
Press 4 to display
Press 5 to exit
Enter your choice
5


...Program finished with exit code 0
Press ENTER to exit console.
```

Q10. Write a C Program implement QUEUE using array in menu driven form.

```
#include<stdio.h>
#define size 5

void Enqueue(int*,int*,int*);
void Dequeue(int*,int*,int*);
void Display(int*,int*,int*);



int main()
{
   int queue[size],front=-1,rear=-1;
   int counter=0,choice;
   while(counter==0)
   {
      printf("Enter\n");
      printf("1.INSERT  \n2.DELETE \n3.DISPLAY \n4.EXIT \n");
      printf("Enter your choice: ");
      scanf("%d",&choice);

      switch(choice)
      {
         case 1:
            Enqueue(queue,&front,&rear);
            break;

         case 2:
            Dequeue(queue,&front,&rear);
            break;

         case 3:
            Display(queue,&front,&rear);
            break;
```

```
            case 4:
                printf("EXITED \n");
                counter++;
                break;

            default:
                printf("WRONG CHOICE \n");
                break;
        }
    }
    return 0;
}

void Enqueue(int queue[],int *front ,int *rear)
{
    int value;
    if(*rear==size-1)
    {
        printf("Queue is Full\n");
    }
    else
    {
        printf("Enter a value\n");
        scanf("%d",&value);
        if(*front==-1)
        {
            *front=*front+1;
        }
        *rear=*rear+1;
        queue[*rear]=value;
        printf("Entered %d SUCCESSFULLY \n",value);
    }
}

void Dequeue(int queue[],int *front,int *rear)
{
    if(*front==-1)
```

```
    {
        printf("Queue is Empty\n");
    }
    else
    {
        printf("Deleted %d Successfully\n",queue[*front]);
        *front=*front+1;
        if(*front>*rear)
        {
            *front=*rear=-1;
        }
    }
}

void Display(int queue[],int *front,int *rear)
{
    if(*front==-1)
    {
        printf("Queue is Empty\n");
    }
    else
    {
        for(int i=*front;i<=*rear;i++)
        {
            printf("%d ",queue[i]);
        }
        printf("\n");
    }
}
```

```
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value
5
Entered 5 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value
6
Entered 6 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value
7
Entered 7 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value
```

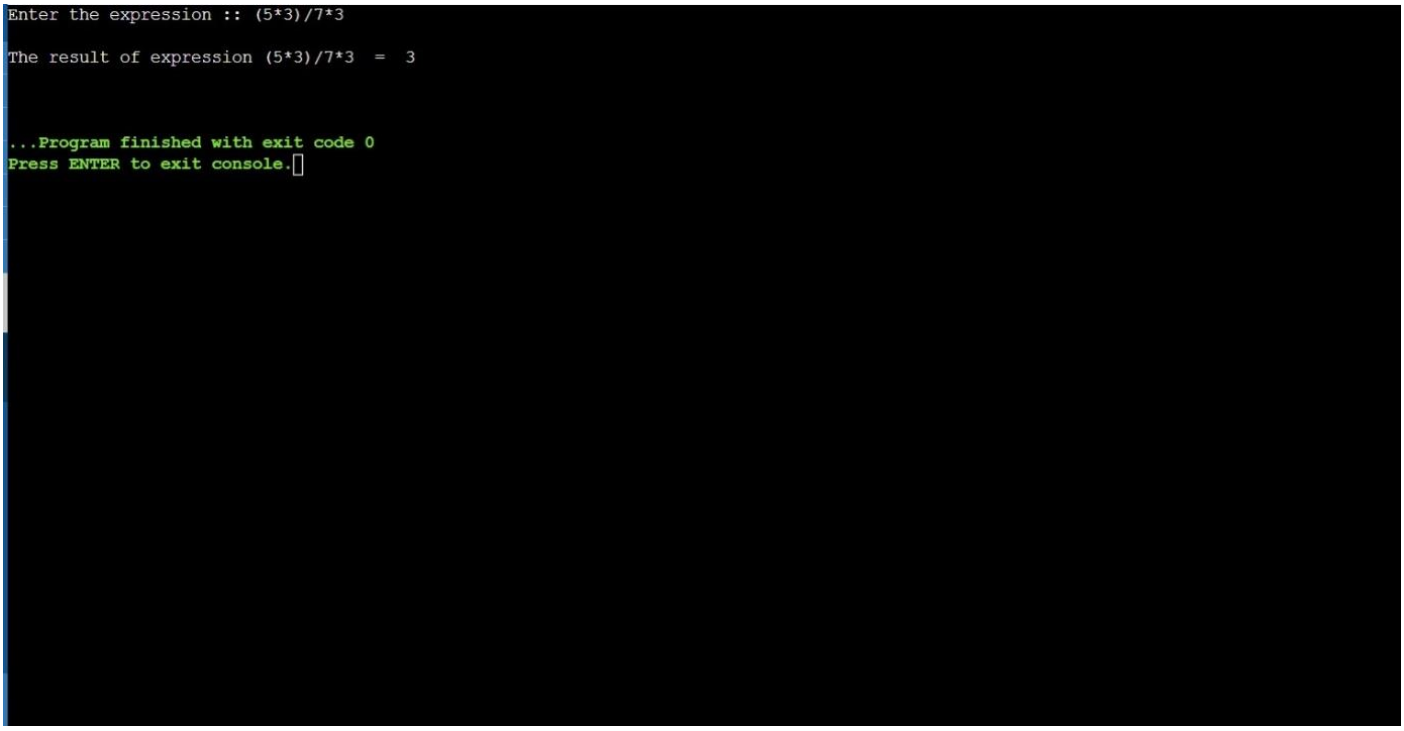
```
Enter your choice: 1
Enter a value
8
Entered 8 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value
9
Entered 9 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
5 6 7 8 9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 2
Deleted 5 Successfully
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 4
EXITED
```


Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 1
Enter a value
9
Entered 9 SUCCESSFULLY
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 3
5 6 7 8 9
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 2
Deleted 5 Successfully
Enter
1.INSERT
2.DELETE
3.DISPLAY
4.EXIT
Enter your choice: 4
EXITED
...Program finished with exit code 0
Press ENTER to exit console.

Q11. Write a C program to Evaluate Postfix Expression using Stack

```
#include<stdio.h>
#include<ctype.h>
int stack[20];
int top = -1;

void push(int x)
{
   stack[++top] = x;
}

int pop()
{
   return stack[top--];
}

int main()
{
   char exp[20];
   char *e;
   int n1,n2,n3,num;
   printf("Enter the expression :: ");
   scanf("%s",exp);
   e = exp;
   while(*e != '\0')
   {
      if(isdigit(*e))
      {
         num = *e - 48;
         push(num);
      }
      else
      {
         n1 = pop();
         n2 = pop();
```

```
            switch(*e)
            {
            case '+':
            {
               n3 = n1 + n2;
               break;
            }
            case '-':
            {
               n3 = n2 - n1;
               break;
            }
            case '*':
            {
               n3 = n1 * n2;
               break;
            }
            case '/':
            {
               n3 = n2 / n1;
               break;
            }
            }
            push(n3);
         }
         e++;
      }
      printf("\nThe result of expression %s  =  %d\n\n",exp,pop());
      return 0;
}
```


Enter the expression :: (5*3)/7*3
The result of expression (5*3)/7*3 = 3
...Program finished with exit code 0
Press ENTER to exit console.

Q 12. Write a C program to create TWO singly linked list L1 and L2 and sort both the list and finally merge both the list such that L2 comes after L1.[use double pointer]

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
   int info;
   struct node *next;
};

void push(struct node **head, int val)
{
   struct node *newNode = (struct node*)malloc(sizeof(struct node));
   newNode->info = val;
   newNode->next = NULL;
   if (*head == NULL)
      *head = newNode;
   else
   {
      struct node *lastNode = *head;
      while (lastNode->next != NULL)
      {
         lastNode = lastNode->next;
      }
      lastNode->next = newNode;
   }
}
void sort(struct node *head)
{
   struct node *temp;
   while(head!=NULL)
   {
      temp=head->next;
```

```
        while(temp!=NULL)
        {
           if(head->info>temp->info)
           {
              int hold=head->info;
              head->info=temp->info;
              temp->info=hold;
           }
           temp=temp->next;
        }
        head=head->next;
    }
}
void merge(struct node *l1,struct node *l2)
{
    while(l1->next!=NULL)
    {
        l1=l1->next;
    }
    l1->next=l2;
}
void print(struct node *ptr)
{
    struct node *temp = ptr;
    while (temp != NULL)
    {
        printf("%d ", temp->info);
        temp = temp->next;
    }
}
int main()
{
    struct node *l1 = NULL,*l2 = NULL;
    push(&l1,19);
    push(&l1,18);
    push(&l1,12);
    push(&l1,11);
```

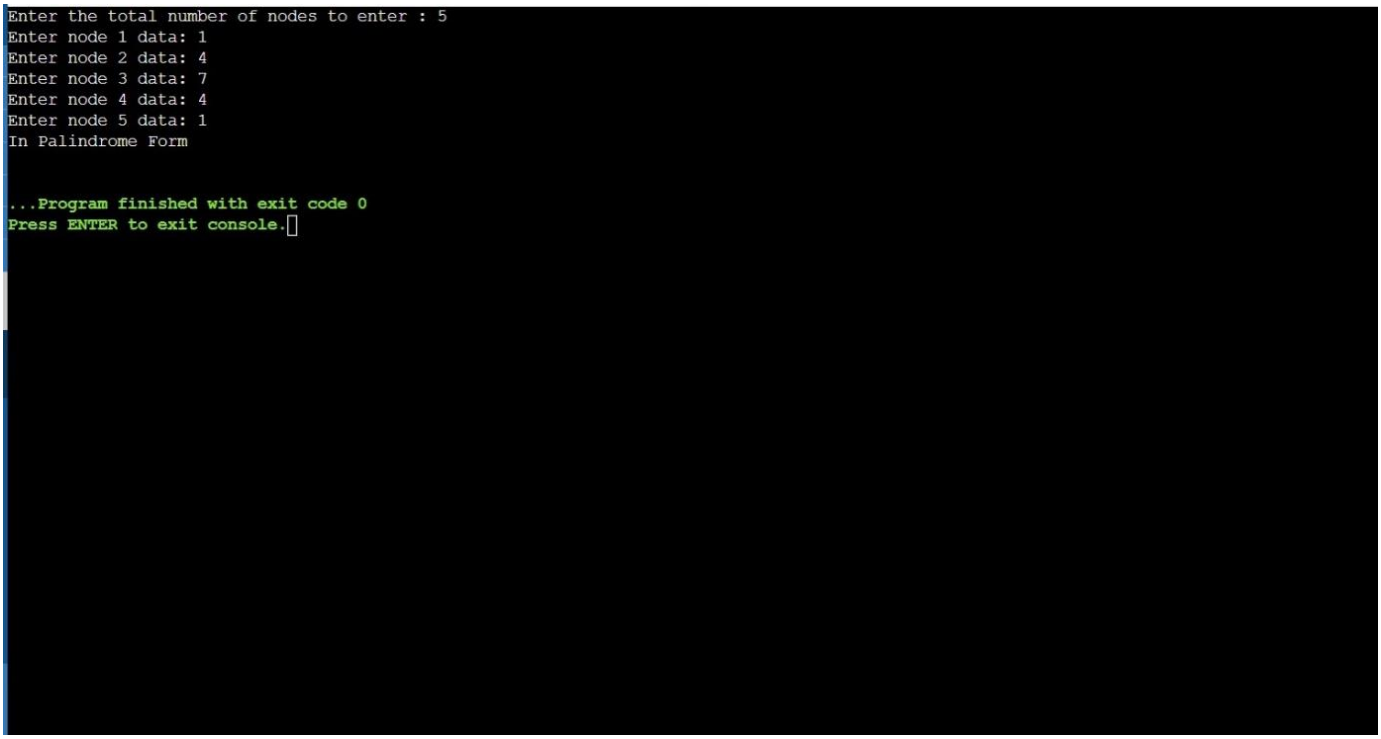
```
    push(&l1,10);
    sort(l1);
    push(&l2,1);
    push(&l2,21);
    push(&l2,8);
    push(&l2,17);
    push(&l2,16);
    sort(l2);
    merge(l1,l2);
    print(l1);

}
```


10 11 12 18 19 1 8 16 17 21
...Program finished with exit code 0
Press ENTER to exit console.

Q 13. Write C program to create a doubly link list by adding the node right hand side and then check list is in palindrome form or not.

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

struct Node
{
  int data;
  struct Node *prev;
  struct Node *next;
};

typedef struct Node node;

void insert(node **,node**,int value);
void PalindromeChecker(node *,node*);

 int main()
 {
   int num,value;
   node *head=NULL,*tail=NULL;
   printf("Enter the total number of nodes to enter : ");
   scanf("%d",&num);
   for(int i=0;i<num;i++)
   {
     printf("Enter node %d data: ",(i+1));
     scanf("%d",&value);
     insert(&head,&tail,value);
   }
   PalindromeChecker(head,tail);


 return 0;
```

```
 }

 void insert(node **head,node **tail, int value)
 {
 node *temp=(node*)malloc(sizeof(node));
 temp->data=value;
 temp->prev=temp->next=NULL;
 if(*head==NULL&&*tail==NULL)
{
  *head=temp;
  *tail=temp;
}
else
{
  (*tail)->next=temp;
  temp->prev=*tail;
  *tail=temp;
}
}
void PalindromeChecker(node *head,node *tail)
{
  int counter=0;
  while(head->next!=NULL||tail->prev!=NULL)
  {
    if(head->data!=tail->data)
    {
      counter++;
      break;
    }
    head=head->next;
    tail=tail->prev;
  }
  if(counter)
    printf("Not in Palindrome Form\n");
  else
    printf("In Palindrome Form\n");
}
```


Enter the total number of nodes to enter : 5
Enter node 1 data: 1
Enter node 2 data: 4
Enter node 3 data: 7
Enter node 4 data: 4
Enter node 5 data: 1
In Palindrome Form
...Program finished with exit code 0
Press ENTER to exit console.

Q14. Write a C program to create a circular link list by adding the nodes in right hand side and then print the list.

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

typedef struct Node node;

void insert(node **,int );
void display(node*);
int main()
{
    node *last;
    int counter=0,choice,value;

    last=NULL;
    while(counter==0)
    {
        printf("Enter\n");
        printf("1.INSERT \n2.DISPLAY \n3.EXIT\n");
        printf("Enter you choice: ");
        scanf("%d",&choice);

        switch(choice)
        {
            case 1:
                printf("Enter a value : ");
                scanf("%d",&value);
                insert(&last,value);
                break;
```

```
            case 2:
                display(last);
                break;

            case 3:
                printf("EXITED\n");
                counter++;
                break;

            default:
                printf("WRONG CHOICE\n");
                break;
        }
    }
    return 0;
}

void insert(node **last,int value)
{
    node *temp=(node*)malloc(sizeof(node));
    temp->data=value;
    if(temp==NULL)
        temp->next=NULL;
    if(*last==NULL)
    {
        *last=temp;
        (*last)->next=temp;
    }
    else{
        temp->next=(*last)->next;
        (*last)->next=temp;
        *last=(*last)->next;
    }
    printf("Entered %d Sucessfully\n",value);
}

void display(node *last)
```

```
{
   node *head=last->next;
   while(head!=last)
   {
      printf("%d ",head->data);
      head=head->next;
   }
   printf("%d \n",head->data);

}
```

```
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 5
Entered 5 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 7
Entered 7 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 8
Entered 8 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 9
Entered 9 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 2
5 7 8 9
Enter
```

```
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 7
Entered 7 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 8
Entered 8 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 1
Enter a value : 9
Entered 9 Sucessfully
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 2
5 7 8 9
Enter
1.INSERT
2.DISPLAY
3.EXIT
Enter you choice: 3
EXITED


...Program finished with exit code 0
Press ENTER to exit console.
```