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

int f=-1,r=-1;
int queue[30];

void renqueue(int x);
void fdequeue();
void rdequeue();
void display();

void display()
{
    if (f==-1)
    {
        printf("\nUnderflow!\n");
    }
    else
    {
        printf("\nItems in list are as follows = \n\n");
        for(int i=f;i<=r;i++)
        {
            printf("%d ",queue[i]);
        }
        printf("\n");
    }
}

void renqueue(int x)
{
    if(r==29) printf("\nOverflow!\n");
    else
    {
        if(f==-1)
        {
            f++;r++;
        }
        else r++;
        queue[r]=x;
    }
}

void fdequeue()
{
    if(f==-1) printf("\nUnderflow!\n");
    else
    {
        printf("\ndequeued item = %d\n",queue[f]);
```

```
        f++;
    }
    if (f>r)
    {
        f=-1;r=-1;
    }
}

void rdequeue()
{
    if (r==-1) printf("\n Underflow!\n");
    else
    {
        printf("\ndequeued item = %d\n",queue[r]);
        r--;
    }
    if(r<f)
    {
        f=-1;
        r=-1;
    }
}

int main()
{
    int ch,ele;
    while(1)
    {
        printf("\nPlease select an option :\n1. Rear enqueue.\n2. Front
dequeue.\n3. Rear dequeue.\n4. Display.\n5. Exit.\n\n");
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {

        case 1:
            printf("\nEnter element: \n");
            scanf("%d",&ele);
            renqueue(ele);
            break;

        case 2:
            fdequeue();
            break;

        case 3:
            rdequeue();
            break;
```

```
        case 4:
            display();
            break;

        case 5:
            exit(0);
            break;

        default:
            printf("\nInvalid input!\n");
            break;
        }
    }
    return 0;
}
```

14.

```
#include <stdio.h>
#include<stdlib.h>
int f=-1,r=-1;
int queue[30];

void fenqueue();
void renqueue();
void fdequeue();
void display();


void display()
{
    if (f==-1)
    {
        printf("\nUnderflow!\n");
    }
    else
    {
        printf("\nItems of list are as follows = \n");
        for(int i=f;i<=r;i++)
        {
```

```
            printf("%d ",queue[i]);
        }
        printf("\n");
    }
}

void fenqueue()
{
    if (f==0) printf("\n no space available in front.\n");
    else
    {
        if (f==-1 )
        {
            f++;
            r++;
        }
        else f--;
        int x;
        printf("\nEnter element: ");
        scanf("%d",&x);
        queue[f]=x;
    }
}

void renqueue()
{
    if(r==29) printf("\nOverflow!\n");
    else
    {
        if(f==-1)
        {
            f++;r++;
        }
        else r++;
        int x;
        printf("\nEnter element: ");
        scanf("%d",&x);
        queue[r]=x;
    }
}

void fdequeue()
{
    if(f==-1) printf("\nUnderflow!\n");
    else
    {
        printf("\ndequeued item = %d",queue[f]);
        f++;
```

```
    }
    if (f>r)
    {
        f=-1;r=-1;
    }
}

int main()
{
    int ch,elem;
    while(1)
    {
        printf("\nPlease select an option :\n1. Front enqueue\n2. Rear
enqueue.\n3. Front dequeue.\n4. Display.\n5. Exit.\n\n");
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
        case 1:
            fenqueue();
            break;

        case 2:
            renqueue();
            break;

        case 3:
            fdequeue();
            break;

        case 4:
            display();
            break;

        case 5:
            exit(0);
            break;

        default:
            printf("\nInvalid input!\n");
            break;
        }
    }
    return 0;
}
```

15.

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

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

node *start,*temp,*last;
int num=0;

void create_node()
{
    temp=( node *)malloc(sizeof(node));
    printf("\nEnter new value : ");
    scanf("%d",&temp->data);
    temp->next=NULL;
    num++;
}

void InsertFirst()
{
    create_node();
    if (start==NULL)
    {
        start=temp;
        last=temp;
    }
    else
    {
        temp->next = start;
        start=temp;
    }
}

void InsertIntermediate()
{
    int pos;
    node *position;
    if(start==NULL)
    {
        create_node();
        start=temp;
        last=temp;
    }
```

```
    else
    {
        printf("\nEnter position at which you want to insert element.\n");
        scanf("%d",&pos);
        create_node();
        if(num<pos) printf ("\nIncorrect position ! %d nodes present
only\n",num);
        else
        {
            if (num==pos)
            {
                last->next = temp;
                last = temp;
            }
            else
            {
                if(pos==1)
                {
                    temp->next=start;
                    start=temp;
                }
                else
                {
                    position=start;
                    for(int i=1;i<pos-1;i++)
                        position=position->next;

                    temp->next=position->next;
                    position->next=temp;
                }
            }
        }
    }
}

void InsertLast()
{
    create_node();
    if (start==NULL)
    {
        start=temp;
        last=temp;
    }

    else
    {
        last->next = temp;
        last=temp;
```

```
    }
    last->next = NULL;
}

void display()
{
    if(start==NULL) printf("\n\nEmpty list!\n\n");
    else
    {
        temp=start;
        printf("\nEntered values are as follows = \n");
        while (temp!=NULL)
        {
            printf("%d ",temp->data);
            temp=(temp->next);
        }
    }

}
int main()
{
    while (1)
    {
        printf("\nChoose one operation =\n1. Create a linked list\n2. Insert
element at starting\n3. Insert element at intermediate location\n4. Insert
element at last position.\n5. Display.\n6. Exit\n\n");
        int ch,n;
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
            case 1:
                printf("\nEnter number of values you want to store = ");
                scanf("%d",&n);
                for (int i = 1; i <= n; i++)
                {
                    create_node();
                    if (start == NULL)
                    {
                        start = temp;
                        last = temp;
                    }
                    else
                    {
                        (last->next)=temp;
                        last=(last->next);
                    }
                }
```

```
                break;

            case 2:
                InsertFirst();
                break;

            case 3:
                InsertIntermediate();
                break;

            case 4:
                InsertLast();
                break;

            case 5:
                display();
                break;

            case 6:
                free(start);
                free(temp);
                free(last);
                exit(0);
                break;

            default:
                printf("Invalid Choice !!");
                break;
        }
    }
    return 0;
}
```

16.

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

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

node *start,*temp,*last;
int num=0;
```

```
void create_node()
{
    temp=( node *)malloc(sizeof(node));
    printf("\nEnter new value : ");
    scanf("%d",&temp->data);
    temp->next=NULL;
    num++;
}

void DeleteFirst()
{
    if (start==NULL)
    {
        printf("\nEmpty!\n");
    }
    else if (start==last)
    {
        printf("\nDeleted value = %d\n\n",start->data);
        free(start);
        free(last);
        temp=NULL;
        num--;
    }

    else
    {
        node *del;
        printf("\nDeleted value = %d\n\n",start->data);
        del = start;
        start = start->next;
        num--;
        free(del);
    }
}

void DeleteLast()
{
    if (start==NULL)
    {
        printf("\nEmpty!\n");
    }

    else if (start==last)
    {
        printf("\nDeleted value = %d\n\n",last->data);
        free(start);
        free(last);
        temp=NULL;
```

```
        num--;
    }

    else
    {
        temp=start;
        while (temp->next != last)
            temp=temp->next;
        printf("\nDeleted value = %d\n\n",last->data);
        free(last);
        last=temp;
        last->next = NULL;
        num--;
    }
}

void DeleteIntermediate()
{
    int node_no;
    node *p;
    printf("\nEnter node number to be deleted\n");
    scanf("%d",&node_no);
    if(num<node_no) printf("\nNo such node exists\n");
    else if (num==node_no) DeleteLast();
    else
    {
        temp=start;
        for(int i=1;i<node_no;i++)
        {
            p=temp;
            temp=temp->next;
        }
        p->next = temp->next;
        printf("\nDeleted value = %d\n\n",temp->data);
        free(temp);
        temp=NULL;
        num--;
    }
}

int main()
{
    while (1)
    {
        printf("\nChoose one operation =\n1. Create a linked list\n2. Delete
first element.\n3. Delete an intermediate element.\n4. Delete last
element.\n5. Display linked list\n6. Exit\n");
        int ch,n;
```

```
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
            case 1:
                printf("\nEnter number of values you want to store = ");
                scanf("%d",&n);
                for (int i = 1; i <= n; i++)
                {
                    create_node();
                    if (start == NULL)
                    {
                        start = temp;
                        last = temp;
                    }
                    else
                    {
                        (last->next)=temp;
                        last=(last->next);

                    }
                }
                break;

            case 2:
                DeleteFirst();
                break;

            case 3:
                DeleteIntermediate();
                break;

            case 4:
                DeleteLast();
                break;

            case 5:
                if(start==NULL) printf("\nEmpty!\n");
                else
                {
                    temp=start;
                    printf("\nEntered values are as follows = \n");
                    while (temp!=NULL)
                    {
                        printf("%d ",temp->data);
                        temp=(temp->next);
                    }
                }
```

```
                break;

            case 6:
                free(start);
                free(temp);
                free(last);
                exit(0);
                break;

            default:
                printf("Invalid Choice !!");
                break;
        }
    }
    return 0;
}
```

17.

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

void push();
void pop();
void peek();
void display();

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

node *top,*temp;

void push()
{
    temp = (node*)malloc(sizeof(node));
    printf("\nEnter element to insert in stack = ");
    scanf("%d",&temp->data);
    temp->next = NULL;
    if(top==NULL) top=temp;
    else
    {
        temp->next = top;
        top=temp;
```

```
    }
}

void pop()
{
    if(top==NULL) printf("\nUnderflow\n");
    else
    {
        temp=top;
        printf("\nPopped data = %d\n",temp->data);
        top=top->next;
        free(temp);
        temp=NULL;
    }
}

void peek()
{
    if(top==NULL) printf("\nUnderflow\n");
    else printf("%d is at the top.\n",top->data);
}

void display()
{
    if(top==NULL) printf("\nEmpty Stack.\n");
    else
    {
        temp=top;
        printf("\nElements of the stack are=\n");
        while(temp!=NULL)
        {
            printf("%d ",temp->data);
            temp=temp->next;
        }
    }
}

int main()
{
    int ch;
    while(1)
    {
        printf("\nchoose any one operation =
\n1.Push\n2.Pop\n3.Peek\n4.Display\n5.Exit\n\n");
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
```

```
        case 1:
            push();
            break;

        case 2:
            pop();
            break;

        case 3:
            peek();
            break;

        case 4:
            display();
            break;

        case 5:
            exit(0);
            break;

        default:
            printf("\nInvalid choice!!\n");
            break;
        }
    }
    return 0;
}
```

18.

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

void enqueue();
void dequeue();
void display();

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

node *temp,*front,*rear;

void enqueue()
```

```
{
    temp=(node *)malloc(sizeof(node));
    printf("\nEnter new element = \n");
    scanf("%d",&temp->data);
    temp->next=NULL;
    if (rear==NULL)
    {
        rear=temp;
        front=temp;
    }
    else
    {
        rear->next = temp;
        rear=temp;
    }
}

void dequeue()
{
    if(front==NULL)
        printf("\nUnderflow!\n");

    else if(front==rear)
    {
        printf("\nDequeued element = %d\n",front->data);
        free(front);
        front=NULL;
        rear=NULL;
    }
    else
    {
        printf("\nDequeued element = %d\n",front->data);
        temp=front;
        front=front->next;
        free(temp);
        temp=NULL;
    }
}

void display()
{
    if(front==NULL) printf("\nEmpty!\n");
    else
    {
        printf("\nElements of the queue are as follows = \n");
        temp=front;
        while (temp!=NULL)
        {
```

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

int main()
{
    int ch;
    while(1)
    {
        printf("\nchoose any operation =
\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n\n");
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
        case 1:
            enqueue();
            break;

        case 2:
            dequeue();
            break;

        case 3:
            display();
            break;

        case 4:
            exit(0);
            break;

        default:
            printf("\nInvalid choice!!\n");
            break;
        }
    }
    return 0;
}
```

19.

```c
#include<stdio.h>
#include<stdlib.h>
#define MAX 5
int queue[MAX];

void enqueue();
void dequeue();
void display();
int front= -1;
int rear= -1;

void enqueue()
{
    if((rear+1)%MAX==front) printf("\nOverflow\n");
    else
    {
        if(front==-1)
        {
            rear++;
            front++;
        }
        else if(rear==MAX-1 && front!=0) rear=0;
        else rear++;
        printf("\nEnter new element = ");
        scanf("%d",&queue[rear]);
    }
}

void dequeue()
{
    if(front==-1) printf("\nUnderflow!\n");
    else
    {
        printf("\nDequeued element = %d \n",queue[front]);
        if(front==rear)
        {
            front= -1;
            rear= -1;
        }
        else if (front==MAX-1) front =0;
        else front ++;
    }
}

void display()
{
   if(front==-1) printf("\nUnderflow!\n");
```

```
   else
   {
       printf("\nElements of the queue are = \n");
       int i=front;
       while(i!=rear)
       {
           printf("%d ",queue[i]);
           i=(i+1)%MAX;
       }
       printf("%d",queue[rear]);
   }
}

int main()
{
    int ch;
    while(1)
    {
        printf("\nchoose any operation =
\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n\n");
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
        case 1:
            enqueue();
            break;

        case 2:
            dequeue();
            break;

        case 3:
            display();
            break;

        case 4:
            exit(0);
            break;

        default:
            printf("\nInvalid choice!!\n");
            break;
        }
    }
    return 0;
}
```

20.

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

void enqueue();
void dequeue();
void display();

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

node *temp,*last;

void enqueue()
{
    temp=(node*)malloc(sizeof(node));
    printf("\nEnter new element = ");
    scanf("%d",&temp->data);
    if(last==NULL)
    {
        last=temp;
        last->next=temp;
    }
    else
    {
        temp->next=last->next;
        last->next=temp;
        last=temp;
    }
}

void dequeue()
{
    if (last==NULL) printf("\nQueue Empty.\n");
    else
    {
        printf("\nDequeued element = %d \n",(last->next)->data);
        if (last->next==last)
        {
            free(last);
            last=NULL;
        }
        else
        {
            temp=last->next;
```

```
            last->next=temp->next;
            free(temp);
            temp=NULL;
        }
    }
}

void display()
{
    if (last==NULL)
        printf("\nQueue Empty.\n");
    else
    {
        printf("\nElements of the queue are = \n");
        temp=last->next;
        do
        {
            printf("%d ",temp->data);
            temp=temp->next;
        }while (temp!=last->next);
    }
}

int main()
{
    int ch;
    while(1)
    {
        printf("\nchoose any operation =
\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n\n");
        fflush(stdin);
        scanf("%d",&ch);
        switch (ch)
        {
        case 1:
            enqueue();
            break;

        case 2:
            dequeue();
            break;

        case 3:
            display();
            break;

        case 4:
            exit(0);
```

```
            break;

        default:
            printf("\nInvalid choice!!\n");
            break;
        }
    }
    return 0;
}
```