

Queue

Queue Implementation

Queue can be implemented using Array and Linked List.

1. Queue implementation using Array.

```
#include <stdio.h>

#include <conio.h>

#define MAX 50

void insert();

void delete();

void display();

int queue_array[MAX];

int rear = - 1;

int front = - 1;

void main()
{
    int choice;
    while (1)
    {
        printf("1.Insert element to queue \n");
        printf("2.Delete element from queue \n");
        printf("3.Display all elements of queue \n");
        printf("4.Quit \n");
        printf("Enter your choice : ");
```

```

scanf("%d", &choice);

switch (choice)
{
    case 1: insert();

    break;

    case 2: delete();

    break;

    case 3: display();

    break;

    case 4: exit(1);

    default: printf("Wrong choice \n");
}
}}

void insert()
{
    int add_item;

    if(rear == MAX - 1)
    {
        printf("Queue Overflow \n");

    }

    else if(front == - 1)
    {
        front=0;

        rear=0;

    }
}

```

```
    else
{
    printf("Inset the element in queue : ");
    scanf("%d", &add_item);
    rear = rear + 1;
    queue_array[rear] = add_item;
    front=0;
}
}
```

```
void delete()
{
    if (front == - 1 || front > rear)
    {
        printf("Queue Underflow \n");
        return;
    }
    else
    {
        printf("Element deleted from queue is : %d\n", queue_array[front]);
        front = front + 1;
    }
}
```

```
void display()
```

```

    {
        int i;
        if (front == - 1)
            printf("Queue is empty \n");
        else
            {
                printf("Queue is : \n");
                for (i = front; i <= rear; i++)
                    printf("%d ", queue_array[i]);
                printf("\n");
            }
    }
}

```

2. Queue implementation using linked list

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

```
#include<conio.h>
```

```
struct Node
```

```
{
```

```
    int data;
```

```
    struct Node *next;
```

```
}*front = NULL,*rear = NULL;
```

```
void insert(int);
```

```
void delete();
```

```
void display();
```

```
void main()
```

```
{
```

```
    int choice, value;
```

```
    clrscr();
```

```
    printf("\n:: Queue Implementation using Linked List ::\n");
```

```
    while(1)
```

```
{
```

```
    printf("1. Insert\n2. Delete\n3. Display\n4. Exit\n");
```

```
    printf("Enter your choice: ");
```

```
    scanf("%d",&choice);
```

```
    switch(choice){
```

```
        case 1: printf("Enter the value to be insert: ");
```

```
                scanf("%d", &value);
```

```
                insert(value);
```

```
                break;
```

```
        case 2: delete(); break;
```

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

```
        case 4: exit(0);
```

```
        default: printf("\nWrong selection!!! Please try again!!!\n");
```

```

    }

}

}

void insert(int value)

{

    struct Node *newNode;

    newNode = (struct Node*)malloc(sizeof(struct Node));

    newNode->data = value;

    newNode -> next = NULL;

    if(front == NULL)

        front = rear = newNode;

    else{

        rear -> next = newNode;

        rear = newNode;

    }

    printf("\nInsertion is Success!!!\n");

}

void delete()

{

    struct Node *temp;

    if(front == NULL)

        printf("\nQueue is Empty!!!\n");

    else{

```

```
    temp = front;

    front = front -> next;

    printf("\nDeleted element: %d\n", temp->data);

    free(temp);

}

}

void display()

{

    if(front == NULL)

        printf("\nQueue is Empty!!!\n");

    else{

        struct Node *temp = front;

        while(temp->next != NULL){

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

            temp = temp -> next;

        }

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

    }

}
```