

Aim:

Write a program to implement queue using **arrays**.

Sample Input and Output:

```
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 1
Enter element : 23
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 1
Enter element : 56
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 3
Elements in the queue : 23 56
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 4
Queue is not empty.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 5
Queue size : 2
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 2
Deleted element = 23
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 2
Deleted element = 56
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 4
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 6
```

Source Code:

QueueUsingArray.c

```
#include <conio.h>
#include <stdio.h>
#include "QueueOperations.c"
int main() {
    int op, x;
    while(1) {
        printf("1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit\n");
        printf("Enter your option : ");
        scanf("%d",&op);
        switch(op) {
            case 1:
                printf("Enter element : ");
                scanf("%d",&x);
```

```

        enqueue(x);
        break;
    case 2:
        dequeue();
        break;
    case 3:
        display();
        break;
    case 4:
        isEmpty();
        break;
    case 5:
        size();
        break;
    case 6: exit(0);
    }
}
}

```

QueueOperations.c

```

#include<stdio.h>
#define MAX 5
int queue[MAX];
int front=-1;
int rear=-1;
int count=0;

void enqueue(int key)
{
    if(front== -1 && rear== -1)
    {
        front++;
        rear++;
        queue[rear]=key;
    }
    else
    {
        rear++;
        queue[rear]=key;
    }
    printf("Successfully inserted.\n");
}

void dequeue()
{
    if(front== -1 && rear== -1)
    {
        printf("Queue is underflow.\n");
    }
    else if(front<=rear)
    {
        printf("Deleted element = %d\n",queue[front]);
        front++;
    }
    else

```

```
{
    printf("Queue is underflow.\n");
}
}
void display()
{
    int i;
    if((front== -1 & rear== -1) || (front > rear))
    {
        printf("Queue is empty.\n");
    }
    else
    {
        printf("Elements in the queue : ");
        for(i=front; i<=rear; i++)

            printf("%d ", queue[i]);

            printf("\n");
    }
}
void isEmpty()
{
    if(front== -1 & rear== -1)
    {
        printf("Queue is empty.\n");
    }
    else if(front <= rear)
    {
        printf("Queue is not empty.\n");
    }
    else
    {
        printf("Queue is empty.\n");
    }
}
void size()
{
    if(front== -1 & rear== -1)
    {
        printf("Queue size : %d\n", 0);
    }
    else if(front == rear)
    {
        printf("Queue size : %d\n", 1);
    }
    else
    {
        printf("Queue size : %d\n", rear-front+1);
    }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 2
Enter your option : 2
Queue is underflow. 3
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 3
Enter your option : 3
Queue is empty. 4
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 4
Enter your option : 4
Queue is empty. 5
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 5
Enter your option : 5
Queue size : 0 1
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 14
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 78
Successfully inserted. 1
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 53
Successfully inserted. 3
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 3
Enter your option : 3
Elements in the queue : 14 78 53 5
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 5
Enter your option : 5
Queue size : 3 6
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 6
Enter your option : 6

Test Case - 2
User Output
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 25
Successfully inserted. 2
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 2
Enter your option : 2
Deleted element = 25 2
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 2
Enter your option : 2
Queue is underflow. 3
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 3
Enter your option : 3
Queue is empty. 1
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 1

Enter your option : 1
Enter element : 65
Successfully inserted. 3
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 3
Enter your option : 3
Elements in the queue : 65 4
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 4
Enter your option : 4
Queue is not empty. 2
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 2
Enter your option : 2
Deleted element = 65 4
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 4
Enter your option : 4
Queue is empty. 5
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 5
Enter your option : 5
Queue size : 0 1
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 1
Enter your option : 1
Enter element : 63
Successfully inserted. 5
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 5
Enter your option : 5
Queue size : 1 6
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit 6
Enter your option : 6