Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)
Department of Information Technology

COURSE CODE: DJS22ITL302
COURSE NAME: Data Structure Laboratory
NAME : Ayush Vinod Upadhyay
ROLL NO. : I025

DATE: 25-10-23
CLASS: II-Batch1
SAP ID: 60003220131

Experiment No. 2

CO/LO: CO1

Aim: Implementation of Priority Queues using Arrays.

Theory: A Priority Queue is a special type of queue in which each element is associated with a priority and is served ~~acc~~ accordingly to it's priority. If elements with the same priority occur, they are served according to their ordering in the queue.

Properties :-

- Each item has a priority associated with it
- An element without priority is dequeued before an element with low priority.
- If two elements have the same priority, they are served acording to their ordererring in the queue.

Operations :-

- Insertion :- This involves adding a new item to the queue
- Deletion :- This involves removing an item from the queue The item with the highest priority is the one that gets deleted.

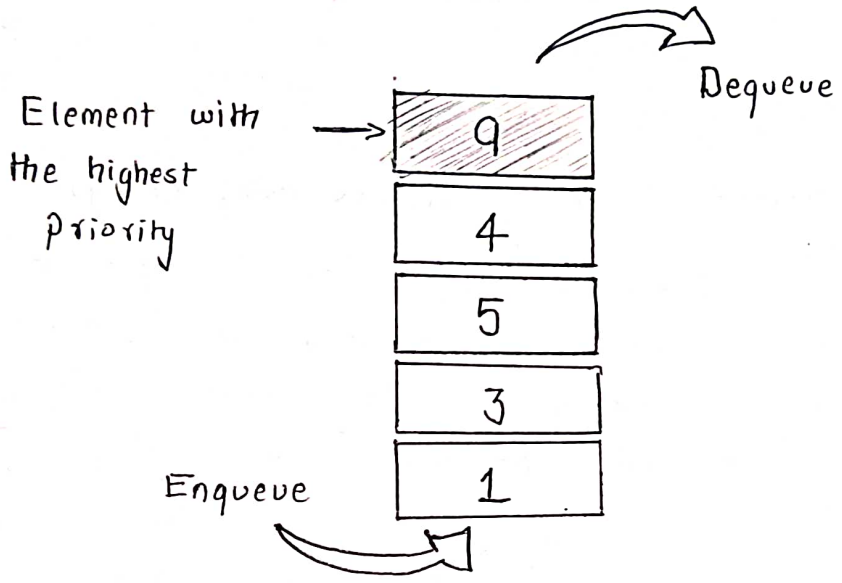
Applications :-

- CPU Scheduling
- Graph algorithms like Dijkstra's shortest path algorithm, Prim's Minimum

Spanning Trees etc.

- Data Compression:-
- In real life scenarios where certain task is needed to give more priority

Output :



Conclusion : I understood that Priority Queue is a fundamental data structure that assigns priorities to each element. The element with highest priority is served first. I understood the implementation using Arrays of Priority Queue.

References : geeksforgeeks.org

atnyla.com

Self implemented the code.

COURSE CODE: DJS22ITL302 **DATE:25/10/2023**
COURSE NAME: Data Structure Laboratory **CLASS: I1-Batch1**

Program:

```c
#include <stdio.h>
#include <stdlib.h>
#define MAX 10
void create_queue();
void insert_element(int);
void delete_element(int);
void check_priority(int);
void display_priorityqueue();
int pqueue[MAX];
int front, rear;
void main()
{
    int n, choice;
    printf("\nEnter 1 to insert element by priority ");
    printf("\nEnter 2 to delete element by priority ");
    printf("\nEnter 3 to display priority queue ");
    printf("\nEnter 4 to exit");
    create_queue();
    while (1)
    {
        printf("\nEnter your choice : ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            printf("\nEnter element to insert : ");
            scanf("%d", &n);
            insert_element(n);
            break;
        case 2:
            printf("\nEnter element to delete : ");
            scanf("%d", &n);
            delete_element(n);
            break;
        case 3:
            display_priorityqueue();
            break;
        case 4:
            exit(0);
        default:
            printf("\n Please enter valid choice");
```

COURSE CODE: DJS22ITL302 **DATE:25/10/2023**
COURSE NAME: Data Structure Laboratory **CLASS: I1-Batch1**

```
        }
    }
}
void create_queue()
{
    front = rear = -1;
}
void insert_element(int data)
{
    if (rear >= MAX - 1)
    {
        printf("\nQUEUE OVERFLOW");
        return;
    }
    if ((front == -1) && (rear == -1))
    {
        front++;
        rear++;
        pqueue[rear] = data;
        return;
    }
    else
        check_priority(data);
    rear++;
}
void check_priority(int data)
{
    int i, j;
    for (i = 0; i <= rear; i++)
    {
        if (data >= pqueue[i])
        {
            for (j = rear + 1; j > i; j--)
            {
                pqueue[j] = pqueue[j - 1];
            }
            pqueue[i] = data;
            return;
        }
    }
    pqueue[i] = data;
}
```

COURSE CODE: DJS22ITL302 **DATE:25/10/2023**
COURSE NAME: Data Structure Laboratory **CLASS: I1-Batch1**

```
void delete_element(int data)
{
    int i;
    if ((front == -1) && (rear == -1))
    {
        printf("\nEmpty Queue");
        return;
    }
    for (i = 0; i <= rear; i++)
    {
        if (data == pqueue[i])
        {
            for (; i < rear; i++)
            {
                pqueue[i] = pqueue[i + 1];
            }
            pqueue[i] = -99;
            rear--;
            if (rear == -1)
                front = -1;
            return;
        }
    }
    printf("\n%d element not found in queue", data);
}
void display_priorityqueue()
{
    if ((front == -1) && (rear == -1))
    {
        printf("\nEmpty Queue ");
        return;
    }
    for (; front <= rear; front++)
    {
        printf(" %d ", pqueue[front]);
    }
    front = 0;
}
```

Output screenshots:

COURSE CODE: DJS22ITL302 **DATE:25/10/2023**
COURSE NAME: Data Structure Laboratory **CLASS: I1-Batch1**

```
Enter 1 to insert element by priority
Enter 2 to delete element by priority
Enter 3 to display priority queue
Enter 4 to exit
Enter your choice : 1

Enter element to insert : 10

Enter your choice : 1

Enter element to insert : 20

Enter your choice : 1

Enter element to insert : 30

Enter your choice : 2

Enter element to delete : 20

Enter your choice : 1

Enter element to insert : 40

Enter your choice : 3
 40  30  10
```