

Experiment 12

Priority Queue Implementation Using Array

Date : 05-11-2020

Aim: To implement a priority queue using array

Data Structure used : Priority Queue, Array

Algorithms

1. Algorithm for enqueue

Input: An Array implementation of Priority Queue (P_Q[SIZE]), with front pointing to the first element and rear pointing to the last element in and an element E to be inserted into the queue, with a priority P

Output: The Priority Queue with the element E inserted at the end

Data Structure: Priority Queue

Steps:

Step 1: if(rear == SIZE) then

Step 1: print("The queue is full insertion not possible")

Step 2: exit(1)

Step 2: else

Step 1: if(rear == -1) then

Step 1: front ++

Step 2: EndIf

Step 3: ++rear

Step 4: Q[rear].elem = E

Step 5: Q[rear].priority = P

Step 3: EndIf

2. Algorithm for dequeue

Input: An Array implementation of Queue (Q[SIZE]), with front pointing to the first element and rear pointing to the last element in the queue.

Output: The element E which has the lowest priority is removed from the priority queue

Steps

Step 1: if(front == -1) then

Step 1: print("The Queue is empty")

Step 2: exit(1)

Step 2: else

Step 1: ptr = front

Step 2: lowestPriority = Q[front].priority

Step 2: while(ptr <= rear)

Step 1: if(Q[ptr].priority < lowestPriority) then

Step 1: lowestPriority = Q[ptr].priority

Step 2: pos = ptr

Step 2: endif

Step 3: ptr++

Step 3: endWhile

Step 4: E = Q[pos].elem

Step 5: While(pos > front) do

Step 1: pos--

Step 2: Q[pos+1] = Q[pos]

```

        Step 6: EndWhile
        Step 7:if(front==rear) then
            Step 1: front=-1
            Step 2: rear = -1
        Step 8:else
            fornt = front +1
        Step 9: endif
    Step 3: endif

```

Description of the Algorithm:

In this algorithm the time complexity of insertion is $O(1)$ while deletion is $O(n)$.

Program code:

```

/* Priority Queue implemetation using dynamic array
 * Done By : Rohit Karuankaran
 * */

#include <stdlib.h>
#include <stdio.h>
#define SIZE 32

typedef struct priority_queue
{
    int **Q;
    int size;
    int front;
    int rear;
}pqueue;

void initQueue(pqueue *q)
{
    q->size = SIZE;
    q->Q = (int**) malloc(q->size*sizeof(int*));

    for (int i = 0;i<q->size;i++)
        q->Q[i] = (int*)malloc(2*sizeof(int));

    q->front = -1;
    q->rear = -1;
}

void delQueue(pqueue *q)
{
    for(int i =0;i<q->size;i++)
        free(q->Q[i]);
    free(q->Q);
}

void enQueue(pqueue *q,int elem,int p)
{
    if(q->rear>=q->size)

```

```

{
    printf("The Queue is full Inseriton not possible\n");
    delQueue(q);
    exit(1);
}
else
{
    if(q->front==-1)
    {
        q->front=q->front+1;
    }
    q->rear = q->rear+1;
    q->Q[q->rear][0] = elem;
    q->Q[q->rear][1] = p;
    return;
}
}

int deQueue(pqueue *q)
{
    if(q->front == -1)
    {
        printf("QUEUE IS EMPTY THERE IS NO ELEMENT TO DELETE\n");
        return -1;
    }

    else
    {
        int ptr = q->front;
        int pos =ptr;
        int priority = q->Q[q->front][1];
        while(ptr<=q->rear)
        {
            if(q->Q[ptr][1]<priority)
            {
                priority = q->Q[ptr][1];
                pos = ptr;
            }
            ptr++;
        }

        int elem = q->Q[pos][0];

        if(pos !=q->front)
        {
            while(pos>q->front)
            {
                pos--;
                q->Q[pos+1][0] =q->Q[pos][0];
                q->Q[pos+1][1] =q->Q[pos][1];
            }
        }

        if(q->front==q->rear)
        {

```

```

        q->rear = -1;
        q->front = -1;
    }
    else{
        q->front += 1;
    }
    return elem;
}
}

void displayQueue(pqueue *q)
{
    int i = q->front;
    if(q->front == -1)
    {
        printf("EMPTY");
        return;
    }
    while(i >= 0 && i <= q->rear)
    {
        printf("%d ", q->Q[i][0]);
        i++;
    }
}

int main()
{
    pqueue *myQueue = (pqueue*) malloc(sizeof(pqueue));

    int RUN = 1;
    int elem;
    int priority;
    int choice;
    initQueue(myQueue);
    while(RUN)
    {
        printf("=====\n");
        printf("          Menu\n");
        printf("=====\n\n");
        printf("1.Enter into the queue\n");
        printf("2.Remove from the queue\n");
        printf("3.Display the queue\n");
        printf("4.Exit\n");
        printf("Enter your choice : ");

        scanf("%d%c",&choice);
        switch(choice)
        {
            case 1: printf("Enter the element you want to enter into the Queue :
");
                    scanf("%d%c",&elem);
                    printf("Enter the priority of the element : ");
                    scanf("%d%c",&priority);
                    enqueue(myQueue,elem,priority);
                    break;

```

```

        case 2: elem = deQueue(myQueue);
                printf("The element remove is :%d\n",elem);
                break;

        case 3: printf("The Queue is: ");
                displayQueue(myQueue);
                printf("\n");
                break;
        case 4: RUN = 0;
                break;
        default: printf("Enter a valid input\n\n");
    }
}

/*
insert(myQueue,32);
insert(myQueue,21);
displayQueue(myQueue);
*/
delQueue(myQueue);
printf("\nExiting.....\n");
}

```

Result: The Program compiled successfully and the desired output was obtained.

Sample input/Output:

```
rohit@iris ~/Programing/C/CSL201/2020-11-05
└─> gcc -Wall priority_queue.c -o priority_queue.o
rohit@iris ~/Programing/C/CSL201/2020-11-05
└─> ./priority_queue.o
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 1
Enter the element you want to enter into the Queue : 12
Enter the priority of the element : 4
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 1
Enter the element you want to enter into the Queue : 0
Enter the priority of the element : 0
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 2
The element remove is :0
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 1
Enter the element you want to enter into the Queue : 34
Enter the priority of the element : 1
```

```
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 2
The element remove is :34
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 2
The element remove is :12
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 2
QUEUE IS EMPTY THERE IS NO ELEMENT TO DELETE
The element remove is :-1
=====
Menu
=====
1.Enter into the queue
2.Remove from the queue
3.Display the queue
4.Exit
Enter your choice : 4
Exiting.....
rohit@iris ~/Programing/C/CSL201/2020-11-05
└─> █
```