|  |  |
| --- | --- |
|  | **DEPARTMENT OF COMPUTER ENGINEERING** |

**Experiment No. 07**

|  |  |
| --- | --- |
| Semester | S.E. Semester III – Computer Engineering |
| Subject | Data Structures Lab (CSL301) |
| Subject Professor In-charge | Prof. Swapnil S. Sonawane |
| Assisting Teachers | Prof. Swapnil S. Sonawane |

|  |  |
| --- | --- |
| Student Name | Trisha Shah |
| Roll Number | 20102A0004 |

**Title:**

Implement Priority Queue ADT using array.

**Objective:**

Students will be able to explain various data structures, related terminologies and its types.

**Explanation:**

A priority queue is a special type of queue in which each element is associated with a priority value. And, elements are served on the basis of their priority. That is, higher priority elements are served first.

However, if elements with the same priority occur, they are served according to their order in the queue.

Generally, the value of the element itself is considered for assigning the priority. For example,

The element with the highest value is considered the highest priority element. However, in other cases, we can assume the element with the lowest value as the highest priority element.

We can also set priorities according to our needs.

Program Code:

#include<stdio.h>

#define N 10

typedef struct queue

{

int a[N];

int front,rear;

}queue;

void enqueue(queue \*q,int x)

{

if(q->rear==N-1)

{

printf("\nQueue Overflow");

}

else

{

q->rear++;

q->a[q->rear]=x;

}

}

int isempty(queue \*q)

{

if(q->front==q->rear+1)

return 1;

else

return 0;

}

int dequeue(queue \*q)

{

int max,p,i;

if(isempty(q))

{

return -1;

}

else

{

max=q->a[q->front];

p=q->front;

for(i=q->front+1;i<=q->rear;i++)

{

if(max<q->a[i])

{

max=q->a[i];

p=i;

}

}

for(i=p;i<=q->rear;i++)//Shifting

{

q->a[i]=q->a[i+1];

}

q->rear--;

return max;

}

}

void display(queue \*q)

{

int i;

if(isempty(q))

{

printf("\nQueue is empty...");

}

else

{

for(i=q->front;i<=q->rear;i++)

{

printf("\n%d",q->a[i]);

}

}

}

int main()

{

int ch,x;

queue q;

q.front=0;

q.rear=-1;

while(1)

{

printf("\nMenu:\n1-ENQUEUE\n2-DEQUEUE\n3-DISPLAY\n4-EXIT\nEnter Choice=");

scanf("%d",&ch);

if(ch==4)

break;

switch(ch)

{

case 1:

{

printf("\nEnter element to be inserted=");

scanf("%d",&x);

enqueue(&q,x);

}

break;

case 2:

{

x=dequeue(&q);

if(x==-1)

{

printf("\nQueue Underflow...");

}

else

{

printf("\nDeleted Element=%d",x);

}

}

break;

case 3:

{

display(&q);

}

break;

default:

{

printf("\nInvalid Choice...");

}

}

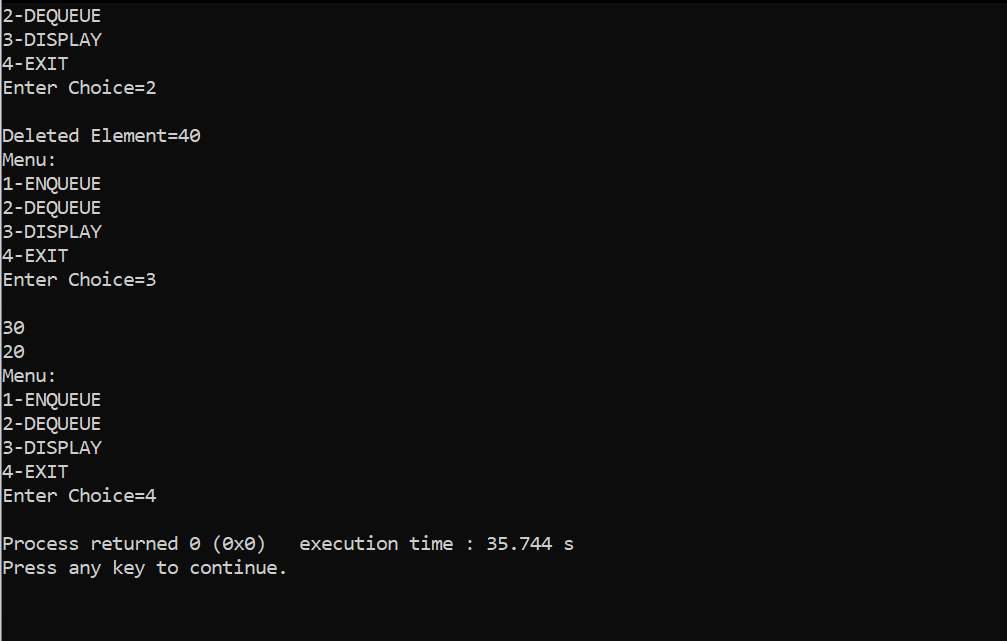
}

return 0;

}

**Output:**

****

****

**Conclusion:**

Through this experiment, students were able to successfully implement a priority queue.