SISTER NIVEDITA UNIVERSITY

School of Engineering

Subject Name:	DATA STRUCTURES AND ALGORITHMS
Subject Code:	
Department:	B.TECH IN COMPUTER SCIENCE AND ENGINEERING

Name:	CHANDREYEE SHOME
Enrolment Number:	2212200001166
Registration Number:	220010415539
Semester:	2nd
Academic Year:	1st (2022 - 2023)

ASSIGNMENT 8

Q 1. Implement Priority Queue using Heap.

CODE:

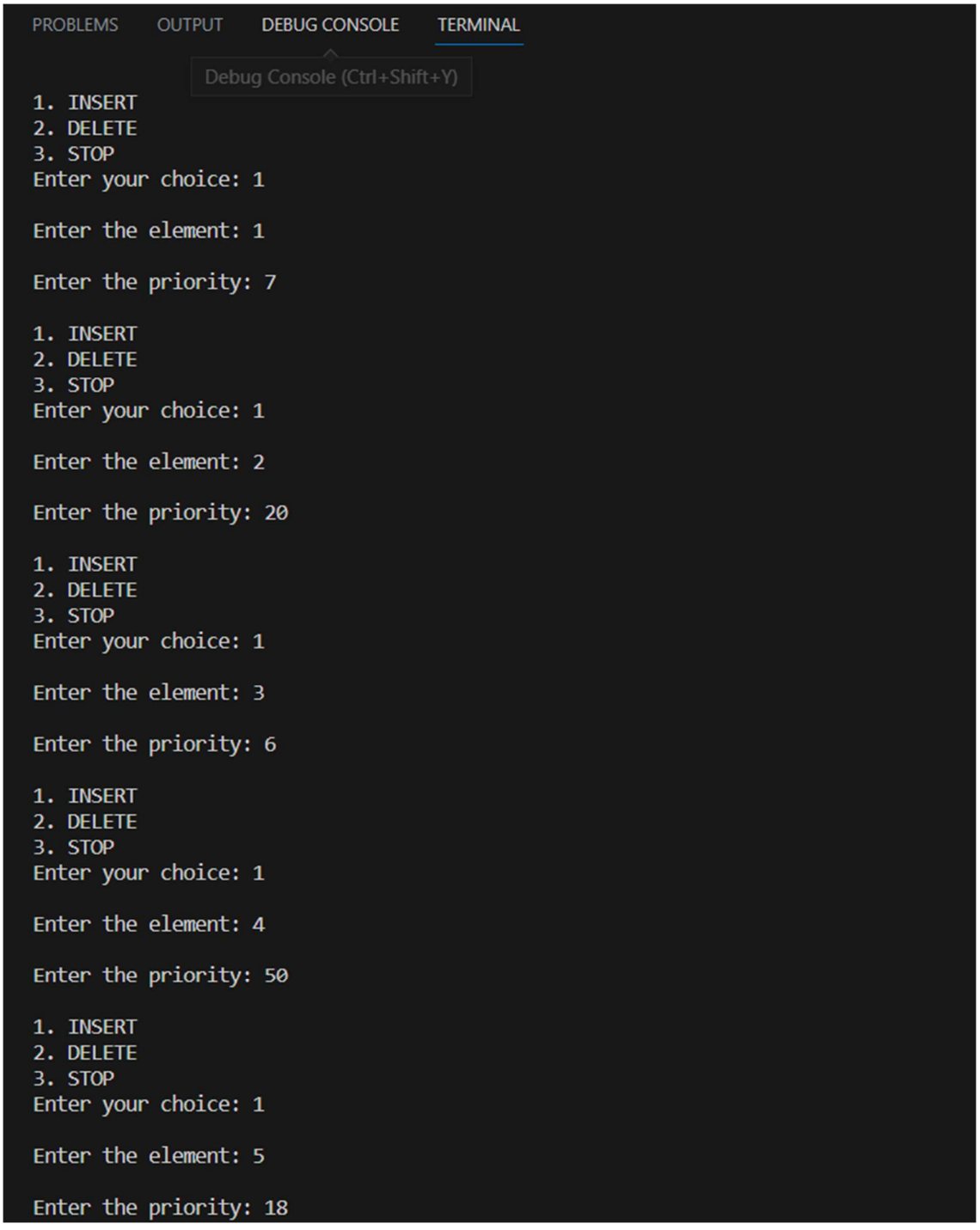
```
//PRIORITY QUEUE (HEAP)

//header files
#include<stdio.h>
#include<stdlib.h>

#define Max 20 //size of priority queue

//user defined structure Item
typedef struct Item
{
    int data;
    int priority;
}Item;

//user defined structure Queue
typedef struct Queue
{
    Item arr[Max];
    int rear;
}Q;

//function declaration
void init(Q *q);
Item createItem(Item Item, int data, int priority);
int isFull(Q *q);
int isEmpty(Q *q);
void swap(Item *item1, Item *item2);
void Insert_heap(Q *q, Item X);
Item Delete_heap(Q *q);

int main()
{
    Item item;
    Q queue;
    init(&queue);
    int data, priority, choice;
    system("cls");
    do
    {
        printf("\n1. INSERT\n2. DELETE\n3. STOP");
        printf("\nEnter your choice: ");
        scanf("%d",&choice);

        switch(choice)
```

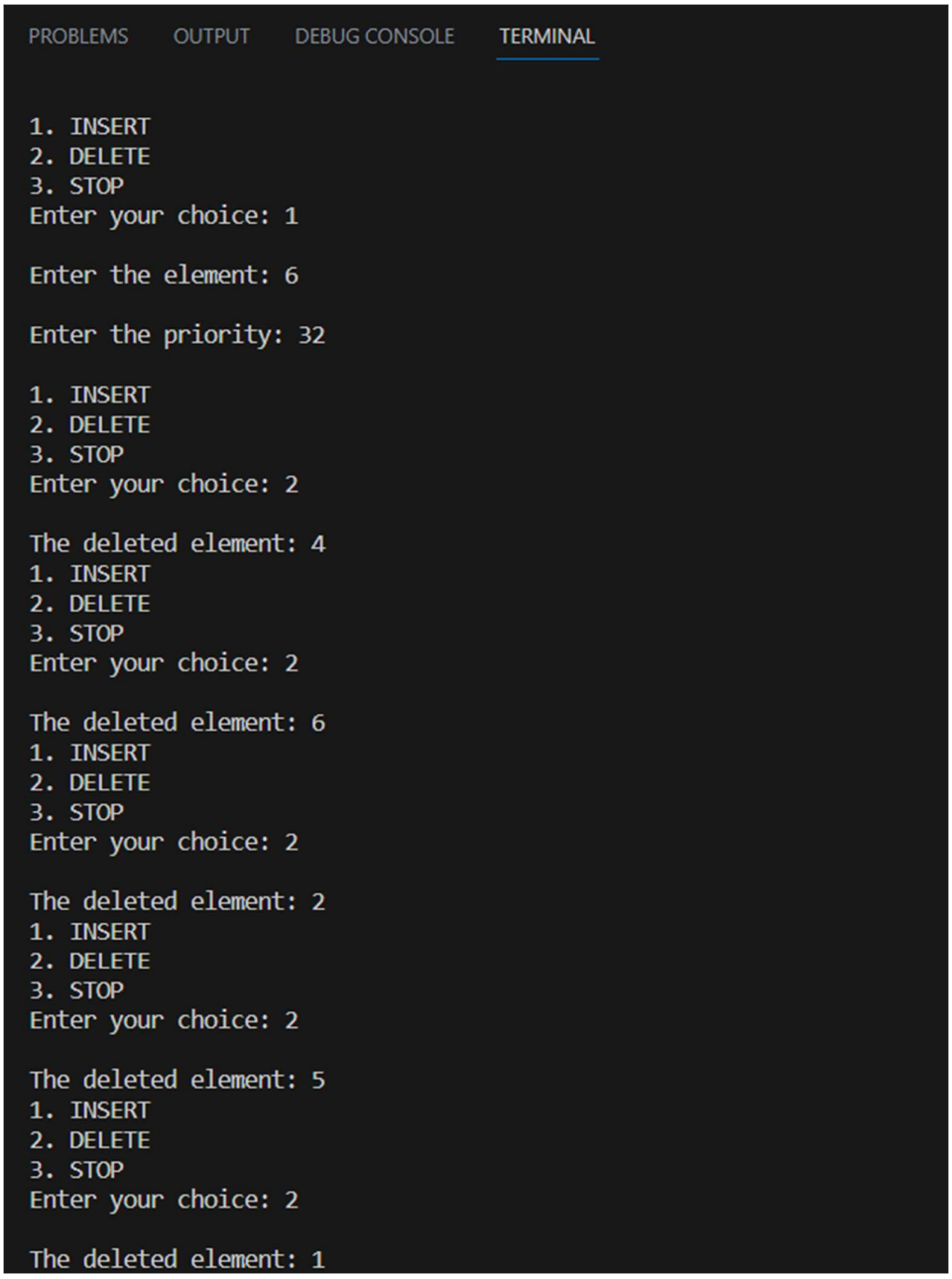
```
        {
            case 1:
                    printf("\nEnter the element: ");
                    scanf("%d",&data);
                    printf("\nEnter the priority: ");
                    scanf("%d",&priority);
                    item = createItem(item, data, priority);
                    Insert_heap(&queue, item);
                    break;

            case 2:
                    if(queue.rear != -1)
                    {
                        item = Delete_heap(&queue);
                        printf("\nThe deleted element: %d",item.data);
                    }
                    else
                    {
                        printf("\nQueue Empty\n");
                    }
                    break;
            case 3:
                    exit(0);
            default:
                    printf("\nWrong Choice!!\n");
                    break;
        }
    }while (choice != 3);

}

//function to initialize queue
void init(Q *q)
{
    q->rear = -1;
}

//function to create item
Item createItem(Item Item, int data, int priority)
{
    Item.data = data;
    Item.priority = priority;
    return Item;
}

//function to chcek if queue is full
int isFull(Q *q)
{
```

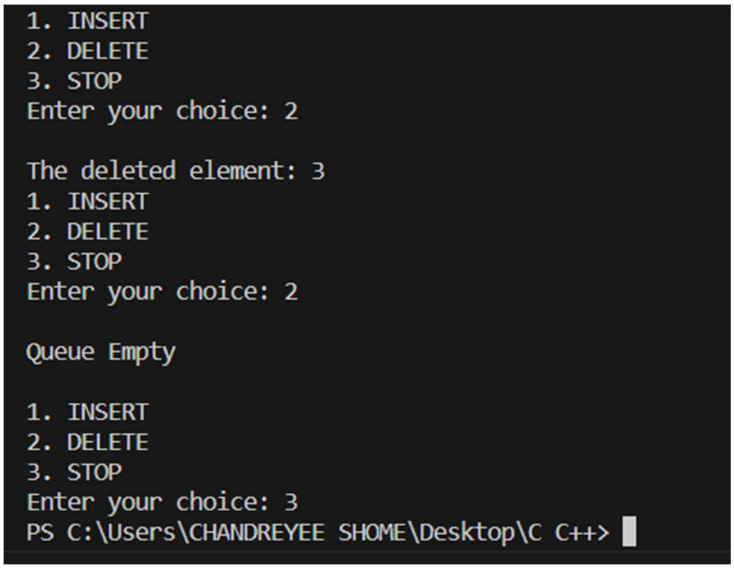
```
    return (q->rear == Max-1);
}

//function to check if queue is empty
int isEmpty(Q *q)
{
    return (q->rear == -1);
}

//function to swap
void swap(Item *item1, Item *item2)
{
    Item temp;
    temp = *item1;
    *item1 = *item2;
    *item2 = temp;
}

//function to insert item in queue
void Insert_heap(Q *q, Item X)
{
    int parent, child;
    if(!isFull(q))
    {
        q->arr[++q->rear] = X;
        child = q->rear;

        do
        {
            parent = (child-1)/2;
            if(q->arr[child].priority > q->arr[parent].priority)
            {
                swap(&q->arr[child], &q->arr[parent]);
            }
            else
            {
                break;
            }
            child = parent;
        }while(child != 0);
    }
}

//function to delete item from queue
Item Delete_heap(Q *q)
{
    int parent, child;
    Item z;
```

```
    if(!isEmpty(q))
    {
        parent = child = 0;
        z = q->arr[0];
        q->arr[0] = q->arr[q->rear];
        q->rear--;
        do
        {
            child = 2*parent +1;
            if(child > q->rear)
            {
                break;
            }
            if(child < q->rear && q->arr[child].priority < q->arr[child +
1].priority)
            {
                child = child + 1;
            }
            if(q->arr[child].priority > q->arr[parent].priority)
            {
                swap(&q->arr[parent], &q->arr[child]);
                parent = child;
            }
            else
            {
                break;
            }
        }while(child <= q->rear);
        return z;
    }
}
```

OUTPUT:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Debug Console (Ctrl+Shift+Y)
1. INSERT
2. DELETE
3. STOP
Enter your choice: 1
Enter the element: 1
Enter the priority: 7
1. INSERT
2. DELETE
3. STOP
Enter your choice: 1
Enter the element: 2
Enter the priority: 20
1. INSERT
2. DELETE
3. STOP
Enter your choice: 1
Enter the element: 3
Enter the priority: 6
1. INSERT
2. DELETE
3. STOP
Enter your choice: 1
Enter the element: 4
Enter the priority: 50
1. INSERT
2. DELETE
3. STOP
Enter your choice: 1
Enter the element: 5
Enter the priority: 18


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1. INSERT
2. DELETE
3. STOP
Enter your choice: 1
Enter the element: 6
Enter the priority: 32
1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
The deleted element: 4
1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
The deleted element: 6
1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
The deleted element: 2
1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
The deleted element: 5
1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
The deleted element: 1


1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
The deleted element: 3
1. INSERT
2. DELETE
3. STOP
Enter your choice: 2
Queue Empty
1. INSERT
2. DELETE
3. STOP
Enter your choice: 3
PS C:\Users\CHANDREYEE SHOME\Desktop\C C++>