```csharp
using System;

namespace coursework
{
    class PriorityQueue
    {
        int size;
        int front, rear, maxSize;
        int[,] queue;
        public PriorityQueue(int maxSize)
        {
            size = 0;
            front = 0;
            rear = -1;
            this.maxSize = maxSize;
            queue = new int[maxSize, 2];
        }
        /// <summary>
        /// add item to queue
        /// </summary>
        /// <param name="item"></param>
        /// <param name="priority"></param>
        public void enQueue(int item, int priority)
        {
            //check if item is in queue, if yes update the distance rather
              than adding item
            if (!isFull())
            {
                size = size + 1;
                int i = rear;
                while (i >= front)
                {
                    if (priority < queue[i, 1])
                    {
                        queue[i + 1, 0] = queue[i, 0];
                        queue[i + 1, 1] = queue[i, 1];
                    }
                    else { break; }
                    i--;
                }
                queue[i + 1, 0] = item;
                queue[i + 1, 1] = priority;
                rear++;
            }
            else
            {
                Console.WriteLine("Queue is full. {0} could not be added.",
                  item);
            }
        }
        /// <summary>
        /// reorders the queue as item priorities are altered
        /// </summary>
        /// <param name="item"></param>
        /// <param name="priority"></param>
        public void UpdateQueue(int item, int priority)
        {
            for (int i = front; i <= rear; i++)
            {
                if (queue[i, 0] == item)
                {
                    queue[i, 1] = priority;
                    BubbleSort();
                }
            }
        }
        /// <summary>
        /// return first item from the queue
        /// </summary>
        /// <returns></returns>
        public int deQueue()
        {
            size = size - 1;
            int item = queue[front, 0];
            if (front != maxSize - 1)
            {
                front = front + 1;
            }
            else
            {
                front = 0;
            }
            return item;
        }

        public bool isFull()
        {
            if (size == maxSize || rear == maxSize - 1)
            {
                Console.WriteLine("Queue is full\n");
                return true;
            }
            else
            {
                return false;
            }
        }
        public bool isEmpty()
        {
            if (size == 0)
            {
                return true;
            }
            else
            {
                return false;
            }
        }
        /// <summary>
        /// bubble sort algorithm
        /// </summary>
        public void BubbleSort()
        {
            int[] temp = new int[2];
            for (int j = front; j < rear; j++)
            {
                for (int i = front; i < rear; i++)
                {
                    if (queue[i, 1] > queue[i + 1, 1])
                    {
                        for (int x = 0; x < 2; x++)
                        {
                            temp[x] = queue[i, x];
                            queue[i, x] = queue[i + 1, x];
                            queue[i + 1, x] = temp[x];
                        }
                    }
                }
            }
        }
    }
}

```