COIS 2020H-Data Structures & Algorithms

Winter 2024

Assignment 3

Submission template.

YOUR NAME: Dikshith Reddy Macherla **STUDENT ID: 0789055**

1. Circular Array

Paste your code for the no-argument constructor.

```
// Initializes a priority queue with a default size of 20.
public PriorityQueue() : this(20) { }
```

Paste your code for the second constructor.

```
// Constructor that initializes a priority queue with a specified size.
// size: The size of the queue.
public PriorityQueue(int size)
{
    this.size = size; // Set the queue size.
    queue = new Node<T>[size]; // Initialize the queue array.
}
```

Paste your code for the Enqueue method.

```
// Adds a new node to the queue based on its priority.
// firstName, lastName: The first and last name of the patient.
// age: The age of the patient.
// priority: The emergency level (priority) of the patient.
public void Enqueue(T firstName, T lastName, int age, int priority)
{
    // Check if the queue is full.
    if ((rear + 1) % size == front)
    {
        Console.WriteLine("Queue is full.");
        return;
    }

    // Create a new node with the given information.
    Node<T> newNode = new Node<T>(firstName, lastName, age, priority);
    // If the queue is empty, initialize front and rear.
    if (front == -1)
    {
        front = rear = 0;
    }
    else
    {
        // Insert the new node in the correct position based on its priority.
        int i;
        for (i = rear; i != front; i = (i - 1 + size) % size)
        {
            if (priority > queue[i].EmergencyLevel)
            {
                queue[(i + 1) % size] = queue[i];
            }
            else
            {
                break;
            }
        }
        queue[(i + 1) % size] = newNode;
        rear = (rear + 1) % size;
    }
}
```

Paste your code for the Dequeue method.

```
// Removes and returns the node at the front of the queue.
// Returns the first name of the dequeued patient as an example.
public T Dequeue()
{
    // Check if the queue is empty.
    if (front == -1)
    {
        Console.WriteLine("Queue is empty.");
        return default(T);
    }

    // Retrieve the patient at the front.
    T item = queue[front].FirstName; // Example return value.

    // If this is the last item, reset the queue to empty.
    if (front == rear)
    {
        front = rear = -1;
    }
    else
    {
        front = (front + 1) % size; // Move front pointer forward.
    }
    return item;
}
```

Paste your code for the PrintAll method.

```
// Prints all nodes in the queue, showing their index, patient information, and emergency level.
public void PrintAll()
{
    // Check if the queue is empty.
    if (front == -1)
    {
        Console.WriteLine("Queue is empty.");
        return;
    }

    // Iterate through the queue and print each node's information.
    int i = front;
    while (i != rear)
    {
        Console.WriteLine($"Index: {i}, Patient: {queue[i].FirstName} {queue[i].LastName}, Age:
{queue[i].Age}, Emergency Level: {queue[i].EmergencyLevel}");
        i = (i + 1) % size;
    }
    // Print the last node's information.
    Console.WriteLine($"Index: {i}, Patient: {queue[i].FirstName} {queue[i].LastName}, Age:
{queue[i].Age}, Emergency Level: {queue[i].EmergencyLevel}");
}
```

Paste your code for the DeleteAll method.

```
// Deletes all nodes in the queue by resetting the front and rear pointers.
public void DeleteAll()
{
    front = rear = -1; // Reset the queue to be empty.
    Console.WriteLine("All patients have been deleted.");
}
```

2. Node Class and Main

Paste your code for the Node class here.

```
// Represents a node in the priority queue, storing patient information and emergency level.
public class Node<T>
```

```
{
   public T FirstName { get; set; } // Patient's first name.
   public T LastName { get; set; } // Patient's last name.
   public int Age { get; set; } // Patient's age.
   public int EmergencyLevel { get; set; } // Emergency level indicating the priority.

   // Constructor to initialize a node with patient information.
   public Node(T firstName, T lastName, int age,

   int emergencyLevel)
   {
      FirstName = firstName;
      LastName = lastName;
      Age = age;
      EmergencyLevel = emergencyLevel; // Set the emergency level (priority).
   }
}
```

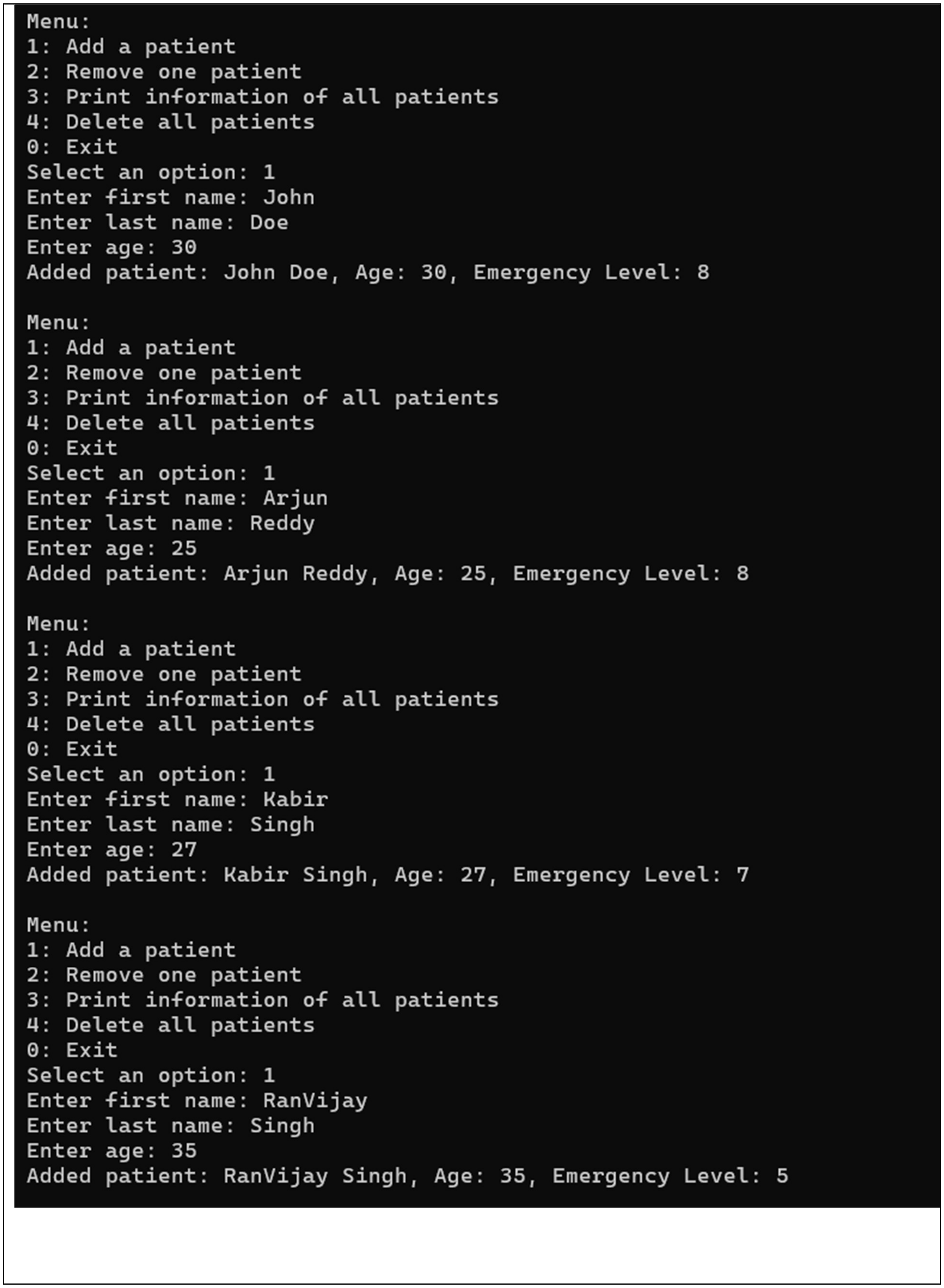
Paste your code for the Main method here.

```
// Entry point of the program.
static void Main(string[] args)
{
   PriorityQueue<string> queue = new PriorityQueue<string>(10); // Initialize the queue with size 10.

   // Loop for user interaction with the priority queue.
   bool continueRunning = true;
   while (continueRunning)
   {
      Console.WriteLine("\nSelect an option:");
      Console.WriteLine("1: Add a patient");
      Console.WriteLine("2: Remove one patient");
      Console.WriteLine("3: Print information of all patients");
      Console.WriteLine("4: Delete all patients");
      Console.WriteLine("0: Exit");
      string option = Console.ReadLine();

      switch (option)
      {
         case "1":
            AddPatient(queue);
            break;
         case "2":
            string removedPatient = queue.Dequeue();
            Console.WriteLine(removedPatient + " has been removed from the queue.");
            break;
         case "3":
            queue.PrintAll();
            break;
            case "4":
                queue.DeleteAll();
                break;
            case "0":
                continueRunning = false;
                break;
            default:
                Console.WriteLine("Invalid option, please try again.");
                break;
        }
    }

    // Optionally, print the queue before exiting.
    Console.WriteLine("Final state of the queue:");
    queue.PrintAll();
}
```

Paste your code for the *AddPatient* method.

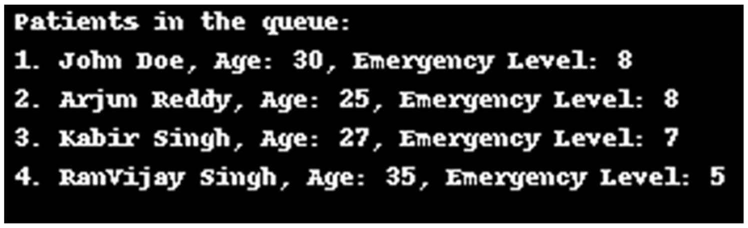
```
// Adds a new patient to the queue with user-provided information and a random emergency level.
static void AddPatient(PriorityQueue<string> queue)
{
    Console.Write("Enter first name: ");
    string firstName = Console.ReadLine();
    Console.Write("Enter last name: ");
    string lastName = Console.ReadLine();
    Console.Write("Enter age: ");
    int age;
    while (!int.TryParse(Console.ReadLine(), out age) || age < 0)
    {
        Console.Write("Invalid age, please enter a valid number: ");
    }

    // Generate a random emergency level between 1 and 10.
    Random rnd = new Random();
    int emergencyLevel = rnd.Next(1, 11);

    // Enqueue the new patient with the provided information and generated emergency level.
    queue.Enqueue(firstName, lastName, age, emergencyLevel);
    Console.WriteLine($"Added {firstName} {lastName} with emergency level {emergencyLevel} to
the queue.");
}
```

Add four patients to the Circular Array object and show the console output here [Provide a screenshot, **not** by pasting the text]



```
Menu:
1: Add a patient
2: Remove one patient
3: Print information of all patients
4: Delete all patients
0: Exit
Select an option: 1
Enter first name: John
Enter last name: Doe
Enter age: 30
Added patient: John Doe, Age: 30, Emergency Level: 8

Menu:
1: Add a patient
2: Remove one patient
3: Print information of all patients
4: Delete all patients
0: Exit
Select an option: 1
Enter first name: Arjun
Enter last name: Reddy
Enter age: 25
Added patient: Arjun Reddy, Age: 25, Emergency Level: 8

Menu:
1: Add a patient
2: Remove one patient
3: Print information of all patients
4: Delete all patients
0: Exit
Select an option: 1
Enter first name: Kabir
Enter last name: Singh
Enter age: 27
Added patient: Kabir Singh, Age: 27, Emergency Level: 7

Menu:
1: Add a patient
2: Remove one patient
3: Print information of all patients
4: Delete all patients
0: Exit
Select an option: 1
Enter first name: RanVijay
Enter last name: Singh
Enter age: 35
Added patient: RanVijay Singh, Age: 35, Emergency Level: 5
```

Show the output of "PrintAll" for the circular array.



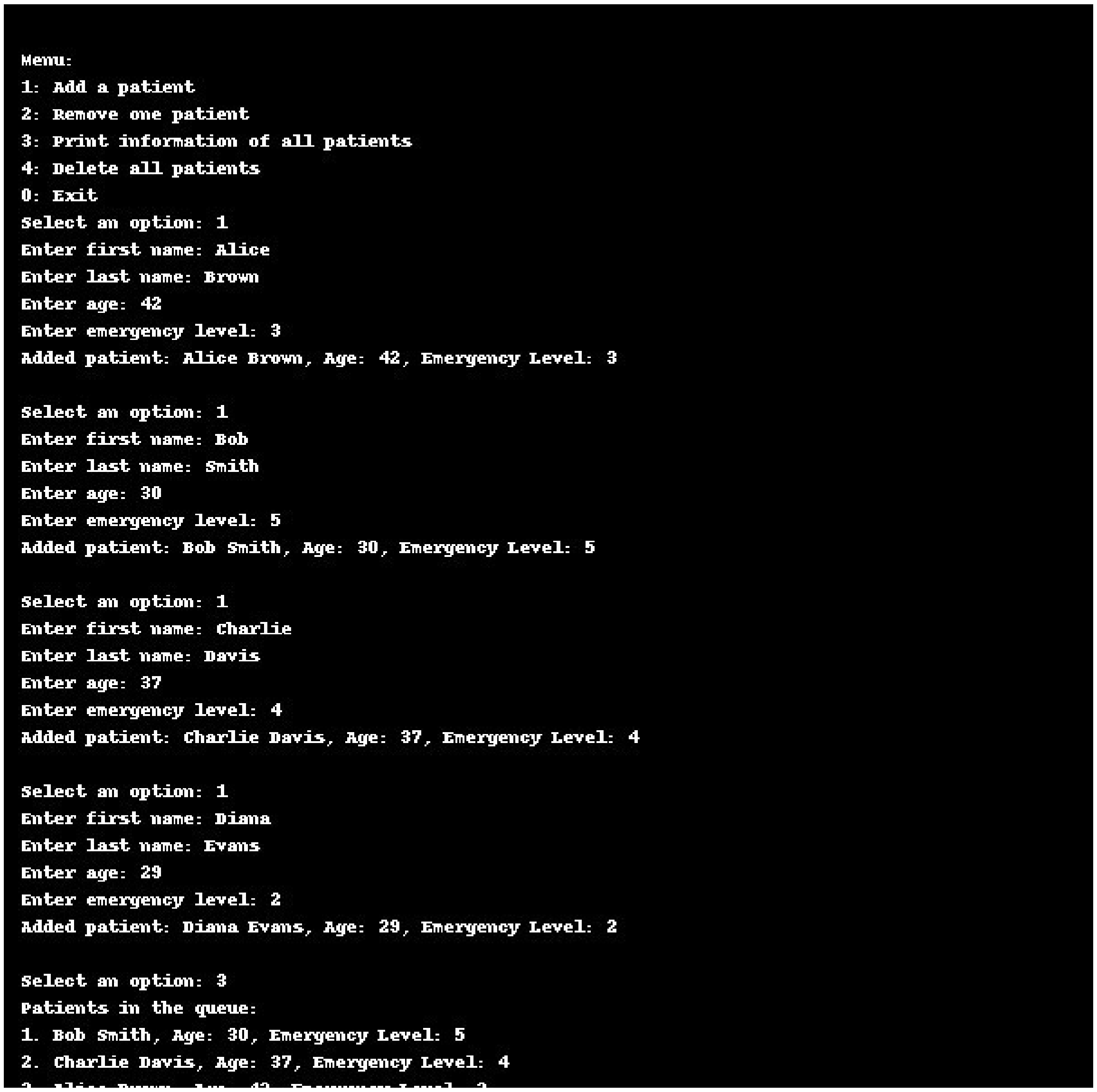
```
Patients in the queue:
1. John Doe, Age: 30, Emergency Level: 8
2. Arjun Reddy, Age: 25, Emergency Level: 8
3. Kabir Singh, Age: 27, Emergency Level: 7
4. RanVijay Singh, Age: 35, Emergency Level: 5
```

3. Testing

After completing the above part, create a new object from the Circular Array with size 4, run the program, and do the following from the console.

1) Add four patients. You can give any value to their names and ages.
2) PrintAll

Show the output here, including the menu printed to the user and the selected option.

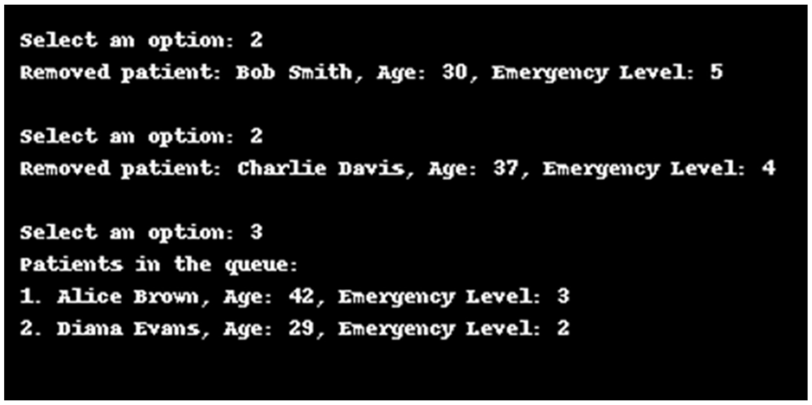


```
Menu:
1: Add a patient
2: Remove one patient
3: Print information of all patients
4: Delete all patients
0: Exit
Select an option: 1
Enter first name: Alice
Enter last name: Brown
Enter age: 42
Enter emergency level: 3
Added patient: Alice Brown, Age: 42, Emergency Level: 3

Select an option: 1
Enter first name: Bob
Enter last name: Smith
Enter age: 30
Enter emergency level: 5
Added patient: Bob Smith, Age: 30, Emergency Level: 5

Select an option: 1
Enter first name: Charlie
Enter last name: Davis
Enter age: 37
Enter emergency level: 4
Added patient: Charlie Davis, Age: 37, Emergency Level: 4

Select an option: 1
Enter first name: Diana
Enter last name: Evans
Enter age: 29
Enter emergency level: 2
Added patient: Diana Evans, Age: 29, Emergency Level: 2

Select an option: 3
Patients in the queue:
1. Bob Smith, Age: 30, Emergency Level: 5
2. Charlie Davis, Age: 37, Emergency Level: 4
```

3) Dequeue
4) Dequeue again

5) PrintAll

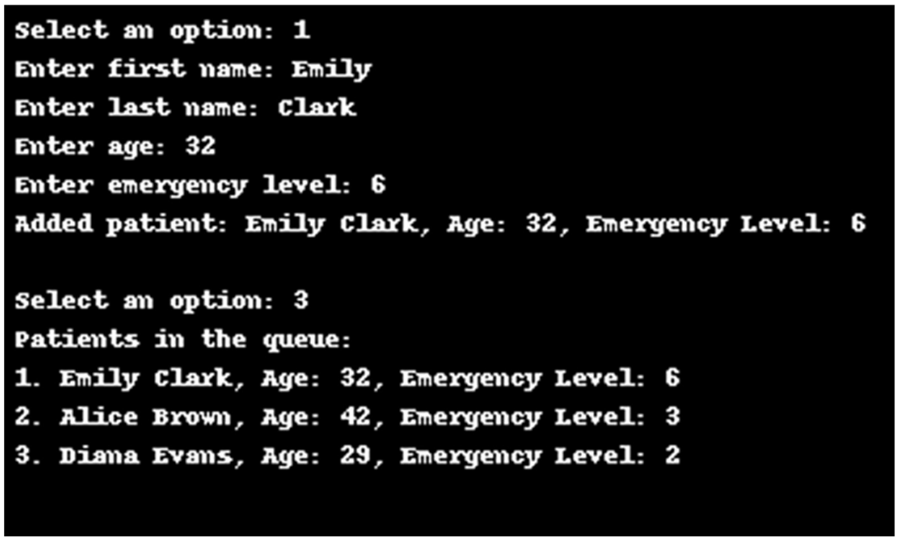
Show the output here, including the menu printed to the user and the selected option.



```
Select an option: 2
Removed patient: Bob Smith, Age: 30, Emergency Level: 5

Select an option: 2
Removed patient: Charlie Davis, Age: 37, Emergency Level: 4

Select an option: 3
Patients in the queue:
1. Alice Brown, Age: 42, Emergency Level: 3
2. Diana Evans, Age: 29, Emergency Level: 2
```

6) Add a new patient
7) PrintAll

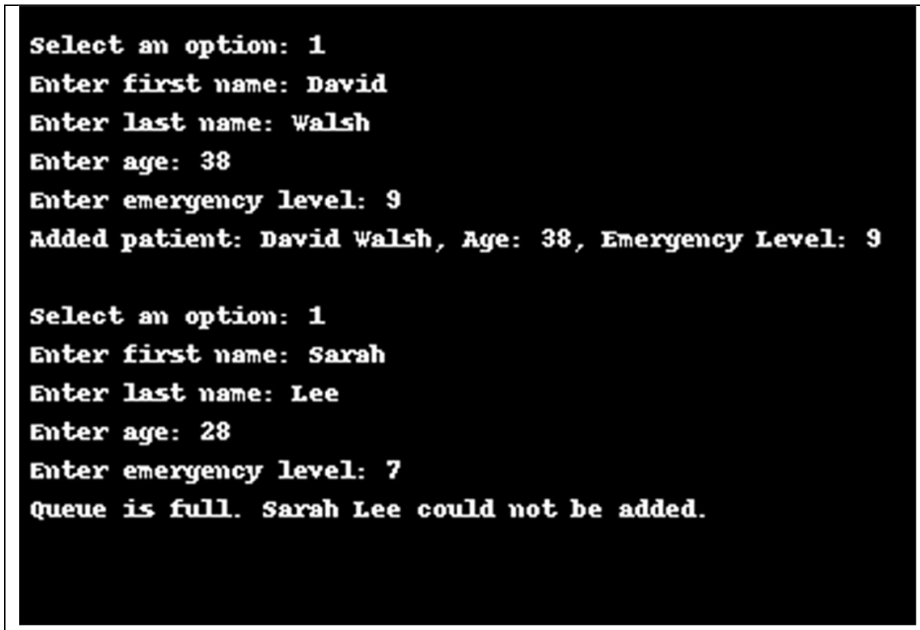
Show the output here, including the menu printed to the user and the selected option.



```
Select an option: 1
Enter first name: Emily
Enter last name: Clark
Enter age: 32
Enter emergency level: 6
Added patient: Emily Clark, Age: 32, Emergency Level: 6

Select an option: 3
Patients in the queue:
1. Emily Clark, Age: 32, Emergency Level: 6
2. Alice Brown, Age: 42, Emergency Level: 3
3. Diana Evans, Age: 29, Emergency Level: 2
```

8) Add new patient
9) Add one more patient

Show the output here, including the menu printed to the user and the selected option.



```
Select an option: 1
Enter first name: David
Enter last name: Walsh
Enter age: 38
Enter emergency level: 9
Added patient: David Walsh, Age: 38, Emergency Level: 9

Select an option: 1
Enter first name: Sarah
Enter last name: Lee
Enter age: 28
Enter emergency level: 7
Queue is full. Sarah Lee could not be added.
```

GOOD LUCK