

**Recep Tayyip Erdogan University**

**Faculty of Engineering and Architecture**

**Computer Engineering**

CE103- Algorithms and Programming I

**Make-up Exam**

**Fall Semester, 2020-2021**

| Instructor | Asst. Prof. Dr. Uğur CORUH |
| --- | --- |
| Contact Information | [ugur.coruh@erdogan.edu.tr](mailto:ugur.coruh@erdogan.edu.tr) |
| Google Classroom Code | **ouw44uk** |
| Publish Date | **11.02.2021** |
| Due Date | **19.02.2021 17:00** |

**Complete the following homework requirements, prepare them in the format given in the link below until the deadline and time, and upload them to the related assignment in the classroom.**

<https://drive.google.com/file/d/1yqSXZZ3346iIqotb_e_yzaryfxEXE0fR/view?usp=sharing>

**Grades:**

| Problem-1 | 60 points |
| --- | --- |
| Problem-2 | 40 points |
| **Total** | **100** points |

*You will develop the following examples in C language. The examples with C ++ will not be accepted.*

NOTE: In the code, add the following information with printf. Put the description of each application in the problem part.

*int main(void)*

*{*

*printf("Build Time: %s %s\n", \_\_DATE\_\_, \_\_TIME\_\_);*

*printf("Owner: Name Surname\n");*

*printf("Student ID: 11111111\n");*

*printf("Course: CE-103\n");*

*printf("Homework: 1\n");*

*printf("Problem: “Printing the Text Entered on the Screen in Reverse \n");*

*... your codes...*

*}*

**Problem-1**: *Hospital Priority Queue (60)*

In this application you will develop a **C application** for a hospital. Patients will register hospital for several pains. In the following table you have pain scores.

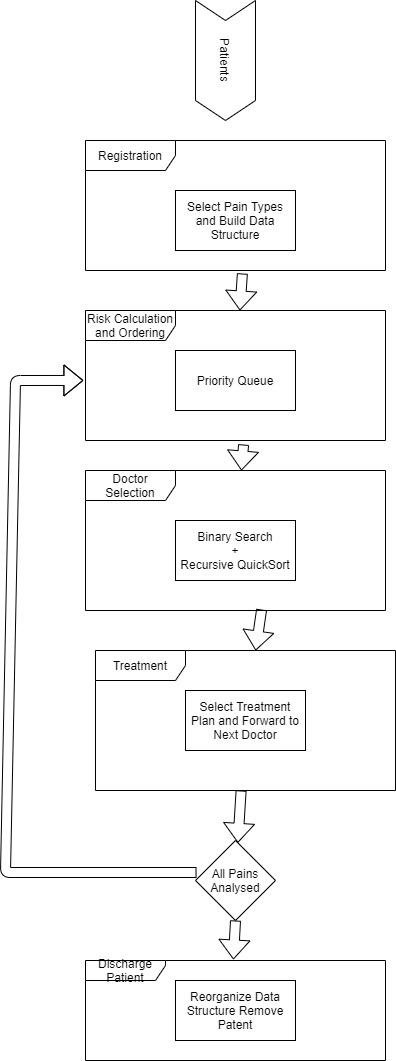
| # | Pain | Score Constant |
| --- | --- | --- |
| 1 | Combined or Medically Unexplained Pain | 90 |
| 2 | Persistent and recurring headache | 80 |
| 3 | Severe pain in the chest, throat, jaw, shoulder, arm or abdomen | 70 |
| 4 | Pain between the lower back or shoulder blades | 50 |
| 5 | Severe Abdominal Pain | 30 |
| 6 | Calf Pain | 20 |
| 7 | Burning Feet or Legs | 5 |

A patient can have multiple pains then you will calculate priority score (pain score) with following formula

This pain score will be weight for priority queue. Also you have **7 doctor for each pain**. This simulation will run for **100 patient** sequentially. There will be a doctor seleciton process also. If doctor not available you will copy this patient to **another waiting queue** and you will move to next patient for doctor selection. 100 patient pain will be randomly set and shoud provide multiple pains each patient should have at least 2 pain.

If doctor found for patient and treatment set that mean pain removed from patient then main waiting priority queue will be reorganized for next doctor selection. If patient all pains are treated then this patient will be discharged and priority queue will be reorganized. This flow will be done untill all patients treated.

You will print log for every steps.

****

**Problem-2**: *Quicksort Analysis (40)*

In this application, you will develop a C application for quicksort performance measurement. You will develop, measure, and compare the following algorithms.

Step-1: Create random input n size

Step-2: Use quicksort algorithms below, measure times and store for graph

Step-3: Increase n size and skip to step-1 (do this untill n<10000)

Step-4: Draw measurements on the console to compare results <https://c-for-dummies.com/blog/?p=831>

* Quicksort with Hoare Partitioning

QUICKSORT (A, p, r)

if p < r then

q ← H-PARTITION(A, p, r)

QUICKSORT(A, p, q − 1)

QUICKSORT(A, q +1, r)

* Quicksort with Lomuto Partitioning

QUICKSORT (A, p, r)

if p < r then

q ← L-PARTITION(A, p, r)

QUICKSORT(A, p, q − 1)

QUICKSORT(A, q +1, r)

* Quicksort with Left Hoare and Right Lomuto Partioning

QUICKSORT (A, p, r)

if p < r then

q ← H-PARTITION(A, p, r)

QUICKSORT\_L(A, p, q − 1)

QUICKSORT\_R(A, q +1, r)

-------------------------

QUICKSORT\_L (A, p, r)

if p < r then

q ← H-PARTITION(A, p, r)

QUICKSORT\_L(A, p, q − 1)

QUICKSORT\_L(A, q +1, r)

-------------------------

QUICKSORT\_R(A, p, r)

if p < r then

q ← H-PARTITION(A, p, r)

QUICKSORT\_R(A, p, q − 1)

QUICKSORT\_R(A, q +1, r)

* Quicksort with One Step Hoare and Next Step Lomuto Partitioning

flag default = false

---------

QUICKSORT (A, p, r, flag)

if p < r then

if flag == true

q ← H-PARTITION(A, p, r)

flag = false

else

q ← H-PARTITION(A, p, r)

flag = true

endif

QUICKSORT(A, p, q − 1,flag)

QUICKSORT(A, q +1, r,flag)

**Note : Please provide source code of applications with your reports**