**Overview**

This assignment is to be implemented using procedural programming. The overall program should process patients through a medical centre. This is not supposed to be a sensible simulation of such a centre, and does not comply with typical operating practices for such centres.

**General code notes**

These are some general rules about what you should and shouldn’t do.

1. Your assignment should be organised into:

(a) **A driver file** containing your main() function.

(b) A **header file** containing the prototypes for the functions you write.

(c) An **implementation file** containing the implementations of your functions

2. Your code must compile with the compilation instructions you provide in Readme.txt.

3. You are not allowed to use classes.

4. You can use **structs**, but without member functions.

5. You should include some exception handling in the case of data file unavailability.

6. Be consistent in your tabbing style.

7. Include sensible volumes of commenting.

8. Use appropriate variable names.

9. Don’t leak memory.

10. The main() function should make it clear what is going in, and shouldn’t be too large

**Run structure**

If your program were to be compiled into the executable MedCent, it must run as follows:

**$ ./MedCent Patients Output-file**

The Medical Centre runs on a strict quota system, and closes after seeing the number of Patients specified as the command argument, or when there are no Patients left to see.

There is otherwise no sense of time in this program, so it’s possible every patient could see the same doctor.

Errors should be reported to standard error. Runtime progress should be reported to standard out. A summary of what happens to each patient should be passed to whatever file is specified on the command line as the **Output-file**.

When you load the data files, you should output the content in a sensible format to make it clear that your input processing works.

* You should deal with the patients on a one by one basis
* Each patient will need to see a randomly chosen doctor of the same gender. If at any time there is no available doctor of the same gender, the patient leaves with their ailment unresolved.
* The doctors will attempt to diagnose the correct ailment, with the percentage chance of doing so being some sensible function of the doctor’s quality, the ailment degree, and the ailment determination complexity. You should specify this function in your Readme.txt file.
* If the doctor fails to identify the correct ailment, the doctor will guess at the ailment, equally likely to specify any ailment including the correct one.
* If a doctor determines that the patient has a particular ailment they will apply the treatment associated with that ailment.
* The chance of the treatment working should be some sensible function of the doctor’s quality and the ailment complexity. If it’s the wrong treatment, the chance of it working should be 25% of the chance when using the correct treatment. You should specify your success function in your Readme.txt file.
* Once given a treatment the patient leaves, whether the treatment is successful or not.

Sensible with respect to the functions above means that increasing or decreasing the arguments should change the chance of success in a common sense way, so a higher quality doctor, higher ailment degree and lower ailment determination complexity give better chances of success.

**Inputs**

Four data files will be provided. The general syntax of those files is described here. The four data files are as follows:

1. **Patients.txt**:

No more than 100 entries.

**Title,Name,Age,Gender,Ailment,Ailment degree.**

Example:

Ms,Alice Anteater,25,F,3,100.

Mr,Bob Badger,37,M,2,50.

Cpl,Carol Carrot,45,F,2,75.

,Dan Digger,30,M,1,20.

Dr,Ernie Edwards,50,M,3,75.

The Age, Ailment and Ailment degree are all integers, with the Ailment degree a percentage in the range 1 to 100. Gender is a single character. The title and name can be anything, although while the title can be empty there must be a name. The Ailment corresponds to the Ailments listed in the Ailments.txt file

The higher the Ailment degree the worse the case of that Ailment the patient has. The higher it is, the easier it is to recognise and the harder to treat.

1. **Doctors.txt**:

No more than 10 entries.

**Name,Gender,Quality.**

Boris Barn-Owl,M,77.

Ernie Edwards,M,60.

Frankie Fisher,M,75.

Geraldine Gardner,F,73.

Henrietta Helpful,F,80.

The name cannot be empty. Gender is a single character. The quality is a percentage in the range 1 to 100.

1. **Ailments.txt**:

No more than 20 entries.

Name,Ailment determination complexity,Ailment treatment complexity,Treatment.

Ailment determination complexity and Ailment treatment complexity are percentages in the range 1 to 100. The lower the Ailment determination complexity, the easier it is to recognise the ailment. The higher the Ailment treatment complexity. the harder it is to treat the ailment successfully. The Treatment corresponds to the Treatments listed in the Treatments.txt file.

Dizziness,20,70,1.

Headache,5,5,3.

Broken bone,5,25,2.

1. **Treatments.txt**:

No more than 10 entries.

**Name.**

Bed rest.

Plaster cast.

Panadol.

Antibiotics.

Miscelleneous Medicine.

If a given data file has an incorrectly formatted line, you should report the problem and ignore that line.

**Output**

Errors should be reported to standard error. Runtime progress, included the file loading output demo, should be reported to standard out.

A summary of what happens to each patient should be passed to whatever file is specified on the command line as the Output-file. This should include the **doctor seen, the ailment diagnosed, and the treatment,** and whether the treatment was successful.