Instructors: Tunca Doğan, Fuat Akal, Aydın Kaya
TA: Hayriye Çelikbilek
Course: BBM103 Fall 2022
Subject: Control Flow, Functions, Lists
Given: 10.11.2022
Due: 24.11.2022 23:00, to classroom.github.com

Assignment 2: Doctor's Aid

1. Introduction

A clinical decision support system (CDSS) is a health information technology that provides clinicians, staff, patients, or other individuals with knowledge and person-specific information, to help health and health care. CDSS encompasses a variety of tools to enhance decision-making in the clinical workflow. These tools include automatic alerts and reminders to care providers and patients, clinical guidelines, condition-specific order sets;', focused patient data reports and summaries, documentation templates, diagnostic support, and contextually relevant reference information, among other tools. Robert Hayward of the Centre has proposed a working definition for Health Evidence: "Clinical decision support systems link health observations with health knowledge to influence health choices by clinicians for improved health care". CDSSs constitute an essential topic in artificial intelligence in medicine.

This assignment focuses on a probability-based decision system called Doctor's Aid for clinicians.

2. Problem

Worldwide, an estimated 19.3 million new cancer cases and almost 10.0 million cancer deaths occurred in 2020. Female breast cancer has surpassed lung cancer as the most commonly diagnosed cancer, with an estimated 2.3 million new cases (11.7%). However, after prevention, early detection and treatment are possible via mammographic screening for breast cancer. Still, it has limitations, such as overdiagnosis and overtreatment [1]. Overdiagnosis is a critical issue in understanding the potential harms of screening for any cancer. In addition to causing harm through overtreatment, it labels a person as having cancer, with all of the psychological, financial, time, and opportunity costs that may entail, regardless of whether the cancer is treated or surveilled [2].

3. Doctor’s Aid Scenario

For this aid utility, patient name, diagnosis accuracy, patient’s disease name, disease incidence, treatment name, and the treatment risk probability will be given in an input file named “doctors_aid_inputs.txt” an example of this file is provided in the attachment. Note that the clinician does not have to give the commands in the same order. Each type of command can be used in any order in the input.

Command types in the input file are as follows:

1. Create a new Patient: The “create” command will add a new patient to the system. For example, the “create Hayriye, 0.999, Breast Cancer, 50/100000, Surgery, 0.40” command means that a patient named Hayriye has been diagnosed with breast cancer with a method that has a 99.9% accuracy ratio. The disease is seen in Turkey in 50 people in every 100,000 population; this phenomenon is known as incidence. Doctors suggest surgery as treatment, and the application risk of the surgery is calculated as 40%. The output of this command will be “Patient Hayriye is recorded.” For another case, the output will be “Patient Hayriye cannot be recorded due to duplication.”. You do not have to check for other cases. Please note that the input data are artificially produced. They are not indicating real-world scenarios.
2. Delete an existing Patient: The “remove” command will remove an existing patient from the system. For example, “remove Deniz” means the user wants to delete the patient named Deniz. The output of this command will either be “Patient Deniz is removed.” or (if there is no such patient) “Patient Deniz cannot be removed due to absence.”.
3. List All Patients with their Information: The “list” command will be used to list the complete information in the system. For example: “list”
   Note that there is no additional operand for this command.

Patient Name	Diagnosis Accuracy	Disease Name	Disease Incidence	Treatment Name	Treatment Risk
Hayriye	99.9%	Breast Cancer	50/100,000	Surgery	40%
Deniz	99.99%	Lung Cancer	40/100,000	Radiotherapy	65%
Toprak	98%	Prostate Cancer	21/100,000	Hormonotherapy	20%

4. Patient’s Probability of having the Disease: The “probability” command will be used to calculate and show/help the actual fatality probability of a patient. For example, the “probability Hayriye” command will have an output of “Patient Hayriye has a probability of **33%** of having breast cancer.” This calculation is made according to the information above in creating part (**Yes, it is true!**). (See also: FP). The other case is “Probability for Hayriye cannot be calculated due to absence.”
5. A Patient’s Recommendation for a Particular Treatment: The “recommendation” command will be used to give a system recommendation to a patient about whether to

have the treatment. As you might notice, the decision is fatally important! For example, the "recommendation Hayriye" command will result in either "System suggests Hayriye to have the treatment." or "System suggests Hayriye NOT to have the treatment.". The recommendation calculation will compare the patient's probability of having the disease and the treatment risk probability. The above command will result in "System suggests Hayriye NOT to have the treatment." in the output file because **40% > 33%**. This assignment will have no confusion, no equal probabilities, or no misspelled inputs. The other case will be "Recommendation for Hayriye cannot be calculated due to absence.". You do not have to check for input correctness. The only case you should check is when a patient's name is given (in any command) whether the patient exists in the system.

Also, outputs of each command's step will be provided to an output file named "doctors_aid_outputs.txt".

4. Implementation

Using a **multi-dimensional built-in python list** as the data structure for the data in the doctor's aid is **obligatory.** For example, example_patient_data_list = [["Hayriye", 0.999, "Breast Cancer" "50/100000" "Surgery" 0.40], ["Deniz", 0.9999, "Lung Cancer" "40/100000" "Radiotherapy" 0.65]] represents the complete information for a particular system. Necessary printing operations can be held besides this multi-dimensional list. Deleted patients will be removed from this list. Names of the patients will be unique, and they can be used as an identifier for the search operations. Also, you should develop a habit of writing explanatory comments during your development for clarity.

You will need to implement at least (1) a recommendation function, (2) a disease probability function, (3) a listing function, (4) a creating new patient function, (5) a removing patient function, (6) a reading the input file function and (7) a saving to the output file function within this assignment implementation.

5. What to include in the report:

- Cover (course name, assignment number, title, student number and name-surname, delivery date)
- Analysis (describe and explain the problem in your own words,
- Design (Write your thoughts on the solution to the problem,
  - for each solution/sub-solution/subroutine/function;
    - write the prototype of that subroutine and explain what it does,
    - write the data structures (arrays) used in that subprogram, explain their purpose,
    - write the algorithm (pseudocode) of that program/subroutine.)

- Programmer's Catalogue (write down the time you spent analyzing, designing, implementing, testing, and reporting; attach the code and explain the reusability of your program/subprogram for other programmers, show description of possible function calls.)
- User Catalogue (program's user manual/tutorial, restrictions on the program)

Some example reports are provided as attachments.

6. Grading

In evaluating the assignment, efficiency and compliance with the structural programming principles will be considered in addition to the correct and complete operation of the program. It is essential to use several necessary functions because, in computer science, it is vital to break down a problem into considerable sub-problems.

Why Do We Write Functions?

1. They allow us to conceive of our program as a bunch of sub-steps. (Each sub-step can be its function. When any program seems too hard, break the overall program into sub-steps!)
2. They allow us to reuse code instead of rewriting it.
3. Functions allow us to keep our variable namespace clean (local variables only "live" as long as the function does). In other words, function_1 can use a variable called I, and function_2 can also use a variable called I, and there is no confusion. Each variable I only exists when the computer is executing the given function.
4. Functions allow us to test small parts of our program in isolation from the rest. This aid is especially true in interpreted languages, such as Python and Matlab, but it can be helpful in C, Java, and ActionScript.

Accordingly, the scoring is as follows:

Evaluation	Points	Evaluate Yourself / Guess Grading
Indented and Readable Codes	5	…
Using Meaningful Naming	5	…
Using Explanatory Comments	5	…
Efficiency (avoiding unnecessary actions)	5	…
Function Usage	25	…
Correctness	35	…
Report	20	…
There are several negative evaluations	…	…

7. Example Program Input/Output Scenario:

doctors_aid_inputs.txt

```
create Hayriye, 0.999, Breast Cancer, 50/100000, Surgery, 0.40
create Deniz, 0.9999, Lung Cancer, 40/100000, Radiotherapy, 0.50
create Ateş, 0.99, Thyroid Cancer, 16/100000, Chemotherapy, 0.02
probability Hayriye
recommendation Ateş
create Toprak, 0.98, Prostate Cancer, 21/100000, Hormonotherapy, 0.20
create Hypatia, 0.9975, Stomach Cancer, 15/100000, Immunotherapy, 0.04
recommendation Hypatia
create Pakiz, 0.9997, Colon Cancer, 14/100000, Targeted Therapy, 0.30
list
remove Ateş
probability Ateş
recommendation Su
create Su, 0.98, Breast Cancer, 50/100000, Chemotherapy, 0.20
recommendation Su
list
probability Deniz
```

doctors_aid_outputs.txt

```
Patient Hayriye is recorded.
Patient Deniz is recorded.
Patient Ateş is recorded.
0.33
System suggests Ateş NOT to have the treatment.
Patient Toprak is recorded.
Patient Hypatia is recorded.
System suggests Hypatia to have the treatment.
Patient Pakiz is recorded.
Patient     Diagnosis   Disease                 Disease     Treatment               Treatment
Name        Accuracy    Name                    Incidence   Name                    Risk
---------------------------------------------------------------------------------------------
Hayriye     99.9%       Breast Cancer           50/100000   Surgery                 40%
Deniz       99.99%      Lung Cancer             40/100000   Radiotherapy            50%
Ateş        99%         Thyroid Cancer          16/100000   Chemotherapy            2%
Toprak      98%         Prostate Cancer         21/100000   Hormonotherapy          20%
Hypatia     99.75%      Stomach Cancer          15/100000   Immunotherapy           4%
Pakiz       99.97%      Colon Cancer            14/100000   Targeted Therapy        30%
Patient Ateş is removed.
Probability for Ateş cannot be calculated due to absence.
Recommendation for Su cannot be calculated due to absence.
Patient Su is saved.
System suggests Su to have the treatment.
Patient     Diagnosis   Disease                 Disease     Treatment               Treatment
Name        Accuracy    Name                    Incidence   Name                    Risk
---------------------------------------------------------------------------------------------
Hayriye     99.90%      Breast Cancer           50/100000   Surgery                 40%
Deniz       99.99%      Lung Cancer             40/100000   Radiotherapy            50%
Toprak      98.00%      Prostate Cancer         21/100000   Hormonotherapy          20%
Hypatia     99.75%      Stomach Cancer          15/100000   Immunotherapy           4%
Pakiz       99.97%      Colon Cancer            14/100000   Targeted Therapy        30%
Su          98.00%      Breast Cancer           50/100000   Chemotherapy            20%
0.80
```

*Disease incidence values are up to date for Turkey, according to Globocan 2018. Everything else is imaginary.
*Input and output files are provided in the attachment for further examination. The table has a specific number of tabs for each data instance.

Please note that,

- ✓ Example input and output files are provided in the attachment for the detailed examination. Note that **punctuation** and **white spaces** in both files are **fatally important**. You may lose points because of punctuation or white spaces.
- ✓ If your program **fails to read** some of the inputs, your whole assignment could crash at the beginning!: This **indicates no grading**.

8. Notes

- **Do not use input() function.**
- Do not miss the submission deadline.
- Save all your work until the assignment is graded.
- The assignment must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating. Show any references or quotes for academic honesty.
- Write READABLE codes and their COMMENTS.
- You can ask your questions via Piazza and you are supposed to be aware of everything discussed on Piazza.
- You will use online Github Classroom system to submit your work.
- No other submission method (email or etc.) will be accepted. Do not submit any file via e-mail related with this assignment.
- You need to name your assignment as **"Assignment2.py"** and gather your work into a **single python file.**
- Do not forget to fill your grading table with your expected marks, and attach the table inside your report file. **The report is obligatory "Assignment2.pdf"**.
- Do not hesitate to ask questions to your TA. TA's office hour will be by appointment.
- Do not forget to **write down your name and number inside both the python (as comment) and pdf documents**.
- You should write, run and **test** your code in your local environment before uploading.
- **File hierarchy**
  - <Inside your commit without a folder>
  - Assignment2.py
  - Assignment2.pdf report
  - doctors_aid_inputs.txt
  - doctors_aid_outputs.txt
- **Sample Run**

  > python3 Assignment2.py

9. Policy

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out abstractly. Discussions about a particular solution to a problem (either in actual code or pseudocode) will not be tolerated. In short, turning in someone else's work (from the internet), in whole or part, as your own will be considered a violation of academic integrity. Please note that the former condition also holds for the material found on the web, as everything on the web has been written by someone else.

References for the Academic Integrity (AI):

https://academicintegrity.ucsd.edu/AI-Handbook-for-UCSD_2019.pdf,
academicintegrity.org/resources/facts-and-statistics