HACETTEPE UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

BBM-103 ASSIGNMENT 2 REPORT

UFUK AĞAYA - 2210356064

24.11.2022

CONTENTS

ANALYSIS

We are asked to extract data from the given text document and make some operations on this data and write it in a the destination text file.

It is necessary to check the first words by reading the given text file line by line. After the first words of the lines are determined, it is determined which operation will be performed according to that word. And the determined operation is finalized through the defined functions and printed to the target text file. This process is looped to repeat for each line. As a result, we get a result file that contains the results of the operations in the text file given at the beginning, line by line.

DESIGN

Reading of data:

- *By reading the text file taken from the user, the patient's data is read into the the data list.*

Repository of Patient Data*:*

- *It is the list where the data of the patients will be kept in order. It is intially empty, but addtions and substractions will occur as the input is read.*

Detection of the first words of the lines:

- *Each line of the given file is read one by one thanks to the loop. By seperating ',' and ' ', the first word of the line is determined and the action to be applied is decided.*

Redirecting to functions based on the first words of the lines:

- *If there is '**create**' at the beginnig of the line:*

The program assumes that the user wants to register a new patient. It checks if the name entered has been created before. If not, it records it. Adds the patient's datas to the data list.

- *If there is '**remove**' at the beginnig of the line:*

The program assumes that the user wants to delete the record of an existing patient. Checks if the given name is among the recorded patients. If it detects that the patient is recorded, it removes it from the list.

- *If there is '**probability**' at the beginning of the line:*

The program calculates the probability that the patient named after the word 'probability' has the disease. It returns a text with the patient's name, the name of the disease and the probability of having the disease.

- *If there is a **'recommendation'** at the beginning of the line:*

The program will make a calculate to generate a recommendation for the patient whose name is written after the word 'recommendation'. It will return a text with a recommendation an whether the patient should receive treatment or not, depending on ehether the probability (calculated in the case of probability) of the patient having the disease is greater than the treatment risk value.

- *If the line containts only **'list'**:*

*The program does not look for data after the word **'list'** in this case. As soon as the program reaches this step, it prints the patient data from data list to the target text file in table format.*

Printing the results of the procedure to the target file:

- *This is the printing of the data returned by the functions to the target file after function routing by line reading. This function is handled by the **write()** function, which is also line-oriented.*

PROGRAMMER CATALOGUE

- ***read():***

Reads the given input file, used for line control within the write function.

```
input_data = file.readlines()
return input_data
```

- ***write():***

To print the returns data to the target file, the input file is opened in write mode: open(input.txt, "w"), at the same time the read() function is used to read the data line by line, and the first words of the lines are used to direct the functions.

```
-----------------------------------------------------------------
with open('doctors_aid_outputs.txt', 'w') as f:
    for line in read():
        first = line.split(",")[0]
        if first.split(" ")[0] == "create":
           f.write(create_patient(line))
            f.write("\n")

         elif...:

        else:

           pass

-----------------------------------------------------------------
```

- ***create_patient(line):***

It is the function that the write function directs the system in create lines. It detects the patient name that comes after the word 'create' in the line, checks whether it is among the registered patients, if it is registered, it returns "patient x is already recorded.", if not, it saves the name to the names list, patient data to the patient_data_list list and returns "patient x is recorded.".

```
---------------------------------------------------------------------

name1 = line.split(",")[0]
name = name1.split(" ")[1]
if name in names:
    return "Patient {} cannot be recorded due to
duplication.".format(name)
else:
    names.append(name)
    patient_data_list.append(line.split(","))
    return "Patient {} is recorded.".format(name)

---------------------------------------------------------------------
```

- ***remove_patient(line):***

It is the function that the wrtie function directs the system to in remove lines. It detects the patient name after the word 'remove' in line, checks whether it is among the registered patients, if it is in the names list, removes the name from the list, likewise deletes the data from the patient_data_list list, and finally returns "patient x is removed.".

```
r = line.split(",")[0]
name_re_with_n = r.split(" ")[1]
name_re = name_re_with_n[:-1]
global name_x
name_x = name_re.replace("\n", "")
if name_x in names:
    global z
    z = int(names.index(name_x))
    names.pop(z)
    for i in patient_data_list:
        name = i[0].split(" ")[1]
        index_of_i = patient_data_list.index(i)
        if name == name_re:
            patient_data_list.pop(index_of_i)
        else:
            pass
    return "Patient {} is removed".format(name_x)
else:
    pass
```

- ***probability(name):***

It is the function directed by the write function on lines of the format 'probability' + patient_name, pulls data from the patient_data_list list, calculates the patient's risk of having cancer as a percentage in the variable 'prob' using **Bayes Theorem***. As a result, it prints output to the target file in the format 'Patient x has a probability of {prob} of having {disease name}'.

→ Getting datas:

```
if name in names:
    state = "create" + " " + name
    patient_data_list_inner = []
    for inner_list in patient_data_list:
        for item in inner_list:
            patient_data_list_inner.append(item)
    name_i = patient_data_list_inner.index(state)
    global acc
    acc = patient_data_list_inner[name_i + 1]
    disease_name = str(patient_data_list_inner[name_i + 2])
    disease_incidence_1 = str(patient_data_list_inner[name_i + 3])
```

***Bayes Theorem**: A *theorem describing how the conditional probability of each of a set of possible causes for a given observed outcome can be computed from knowledge of the probability of each cause and the conditional probability of the outcome of each cause.*

→ Calculating Probability:

```
-------------------------------------------------------------------------------

global numerator
nominator = disease_incidence_1.split("/")[0]
global denominator
denominator = disease_incidence_1.split("/")[1]
global prob
prob_num = float(acc) * int(nominator)
de_1 = float(denominator)-float(nominator)
de_2 = 1-float(acc)
prob_de = (de_1*de_2)+(float(acc)*float(nominator))
prob = prob_num / prob_de

-------------------------------------------------------------------------------
```

→ Making the return sentence with conditionals:

```
-----------------------------------------------------------------------------------------

prob_str = str(prob)
    decimal_control = prob_str[3] + prob_str[4]
    if decimal_control == "00":
        prob_percent = "{:.0%}".format(float(prob))
        return "Patient {0} has a probability of {1} of having
{2}.".format(name, prob_percent, disease_name)
    else:
        prob_percent = "{:.2%}".format(float(prob))
        return "Patient {0} has a probability of {1} of having
{2}.".format(name, str(prob_percent), disease_name)
else:
    return "Probability for {0} cannot be calculated due to
absence.".format(name)

-----------------------------------------------------------------------------------------
```

- ***probability_r(name):***

It is not one of the functions where the write function directs the system. It differs from probability() in that it gives the risk calculation directly in numbers, not in text format. It is designed to be used in the probability calculation in the recommendation() function

- ***recommendation(line):***

It is the function directed by the write function on lines starting with 'recommendation'. It detects the patient name after the word 'recommendation', takes the treatment_risk value from the patient_data_list,and calls the probability_r() function to get the patient's risk value of having the disease. If the patient's risk value is greater than the treatment risk, the sentence 'System suggest {patient name} to have the treatment.' is written to the target file, if the patient's risk value is not greater than the treatment risk, the sentence 'System suggest {patient name} to not have the treatment.

→ Making an inner list to use:

```
-------------------------------------------------------------------------
line1 = line.replace('\n', '')
name_r = line1.split(" ")[1]
state = "create" + " " + name_r
patient_data_list_inner = []
for inner_list in patient_data_list:
    for item in inner_list:
        patient_data_list_inner.append(item)

-------------------------------------------------------------------------
```

→ Making return sentences:

```
if not name_r in names:
    return "Recommendation for {} cannot be calculated due to
absence.".format(name_r)
else:
    name_i = patient_data_list_inner.index(state)
    treatment_risk = patient_data_list_inner[name_i + 5]
    treatment_risk = treatment_risk.replace('\n', '')
    treatment_risk = float(treatment_risk)
    prob = probability_r(name_r)
    if line1.split(" ")[0] == "recommendation":
        if prob > treatment_risk:
            return "System suggests {} to have the
treatment.".format(name_r)
        else:
            return "System suggests {} NOT to have the
treatment.".format(name_r)
    else:
        return "Recommendation for {] cannot be calculated due to
absence.".format(name_r)
```

- ***listing():***

It is the first of the functions directed by the write() function on lines where the line consists of the word 'list'. It is not affected by any data in the line, it takes no parameters. There is only one output possibility. The heading of the list to be printed is included in this function, the reason for separating it from the listing_2() function, which holds patient data and takes parameters, is to create a more understandable and readable layout.

```
heading =
f'{"Patient":8}{"Diagnosis":12}{"Disease":16}{"Disease":12}{"Treatme
nt":16}{"Treatment":16}'
heading_2 =
f'{"Name":8}{"Accuracy":12}{"Name":16}{"Incidence":12}{"Name":16}{"R
isk":16}'
return heading + "\n" + heading_2 + "\n" + "-"*73
```

- ***listing_2():***

It is the second of the functions directed by the write() function on lines where the line consists of the word 'list'. It does not take any parameters. It gets the required data from patient_data_list. Since patient_data_list retrieves the data at the time it is called, the recorded and deleted patients are in the desired order. With the Alyne method, the data is printed to the target file one after the other and in an organized structure.

```
line_1 = []
for inner_list in patient_data_list:
    line_1.append(inner_list)
data_set = ""
for i in line_1:
    is_create = i[0].split(" ")[0]
    if is_create == "create":
        name = i[0].split(" ")[1]
        percentage_t = "{0:.2%}".format(float(i[5]))
        list_float = i[1]
        percentage_d = "{0:.2%}".format(float(list_float))
        info =
f'{name:8}{percentage_d:11}{i[2]:16}{i[3]:12}{i[4]:17}{percentage_t:
16}'
        data_set += info + "\n"
    else:
        pass
data_set = data_set[:-1]
return data_set
```

USER CATALOGUE

First, the user must prepare the data according to the following conditions:

- In order for the program to give results with 100% accuracy, each operation must be written on separate lines and one line space must be left at the end of the Input file.

- If a patient is to be recorded, the word 'create' should be written first at the beginning of the line and then the patient's information should be written with ', ' between each information and in the order specified below. If the name entered is already registered, return will be 'Patient x is already recorded'.

- If you want to delete a recorded patient, the word 'remove' should be entered first at the beginning of the line, followed by the name of the patient to be removed.

- If you want to calculate the Diagnosis Accuracy of the registered patient, the word 'probability' should be written at the beginning of the line, then enter the name of the patient whose Diagnosis Accuracy is to be calculated.

- If you want to get a recommendation from the system for a recorded patient, enter 'probability' at the beginning of the line, followed by the name of the patient whose system recommendation is to be calculated. If the patient name entered is recorded, the output of the program will be 'System suggests patient x to have treatment' or 'System suggests patient x not to have treatment'. If the patient name entered is not recorded, the output will be 'Recommendation for {} cannot be calculated due to absence'.

- If you want to show the information of all recorded patients in a table, just type 'list' in the line. In this way, patients who are recorded to the system with the 'create' method without patients who are unrecorded from the system with the 'remove' method will be printed to the target file in table format with all their data.

GUESS GRADING

Evaluation	Points	Guess Grading
Indented and Readable Codes	5	5
Using Meaningful Naming	5	5
Using Explanatory Comments	5	5
Efficiency (avoiding unnecessary actions)	5	3
Function Usage	25	22
Correctness	35	35
Report	20	17
Total	100	92