COURSE NAME:INTRODUCTION TO PROGRAMMING LAB.

ASSIGNMENT NUMBER: 2

STUDENT NUMBER : 2210356066

NAME/SURNAME:ZEYNEP NİSA KARATAŞ

DELİVERY DATE:23/11/2022

CONTENTS

ANALYSIS

Worldwide, an estimated 19.3 million new cancer cases and almost 10.0 million cancer deaths occurred in 2020. Allthough early detection may help preventing the cancer but still there are so many risks for the patient. For example overdiagnosis or overtreatment would make everything worse for patients.So the probability should be calculated very carefully and if people have cancer, the treatment has the most important part.With this program we try to make certain calculations considering the confusion matrix and according those probabilities the program decides whether the patient should have the treatment or not.

DESIGN

THE PROBLEM AND ITS SOLUTION

We were asked to make the program read a file and take the 5 command which are create, remove, possibility, recommendation and list then the program should respond those commands in another file.First the program should read the doctors_aid_inputs.txt file and take commands from each line.After taking each command and making different functions for them the program should respond those commands in another file called doctors_aid_outputs.txt. Also the probability calculation is especially important because if we make wrong calculations patients may get the wrong treatment so it would cause a lot of harm.So I consider the confusion matrix for calculations throughout the process of writing probability function.

PROTOTYPE OF THE SOLUTION

I made 7 functions for each step.Functions names are input_txt, output_txt, create, remove, probability, recommendation, list.Input_txt function will read the doctors_aid_input.txt so we can take the 5 command.Create function will add new patients to the system.Remove function will remove the patients who are written beside the remove command.Probability function will calculate the exact possibility for a person to have the cancer or not considering the confusion matrix.Recommendation function will compare the risk and the probability so that it can deside whether the patient should get the treatment or

not.List function will make a table inside the doctors_aid_output.txt so we can see all the informations for each patient who have been recorded to the system.

THE DATA STRUCTURES USED IN PROGRAM AND THEIR PURPOSE

I used lists in this program.For recording the patients in a list I used a multi-dimensional built-in list and I used it in every command function because I needed to select the informations from each patient.The input function reads the line and while I was writing the create function I took index for each line and made a list for each patient then I named it as patient_input.Then the create function was taking the create command and I took patient_inputs into another list called patientlist by using list mutation and I append the list with list.append command so the program could add new patients.For remove function the command wants us to remove patients from system so I used list.pop for removing the patient who was written in remove lines.

SUBROUTINE OF THE PROGRAM

First of all I made the program read and write the input file and write the output file. While program was reading the input file I added a new blank line so it can read and pick all the informations precisely.Inside the input_txt function I called output function because 5 command functions were in the output function.If there is create command the program will call the output function and then call the create function.If the patient who was giving in the input file was not in the system then this function will record the patient and made the program write into the output file with patient's name as recorded.If patient was already in system it will write this patient cannot be recorded due to duplication to the output file.For remove function we take the names from lines by clarifying the place of them then make a

loop. If patient is in the system the program removes the patient by using list.pop but if not it will write this patient cannot be removed due to absence into output file.The probability function will take the names with same method as remove and then if patient is in system it will calculate the possibility for a person to have the cancer or not .The method of calculatingprobability comes from confusion matrix.

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

For the calculation we substract the percentage of diagnosis accuracy from 100,then we multiply this percentage with the number of total people from treatment incidence which were given with patient input and we get our numerator.Then we take the numerator of treatment incidence and take the summation of the numerator we found and make it our denominator then we multiply this statement with 100 so we achieve the real percentage of a person's cancer probability.

Numerator=(100-Diagnosis accuracy).(Denominator of treatment incidence)

Denominator=Numerator of treatment incidence + Numerator

(Numerator/Denominator)*100= The real probability

The recommendation function will take the patient's name first.Then with a for loop wthe function check every patient who's name is written beside recommendation command .In this loop program calculates the probability and compare the risk and probability by using if

statements.If risk is less than probability it will write “System suggest this patient to have the treatment” into output file but if risk is greater than probability it will write “System suggests this patient not to have the treatment.” İnto output file.However the patient may not be in system so the function will write “Recommendation for this patient cannot be calculated due to absence.” to the output file.With the list function will write the titles first considering the length of the words and then it will get into a loop.The loop has 6 variables which are the informations belong to patients in system. It wont write deleted patients to the table.

PROGRAMMER'S CATALOGUE

```
patientlist = []  # this is the multi-dimensional list that will  be filled
def output_txt(command):  # 5 of the functions are in this function so it can call them easily

    def create():
        # this function will take the create command and will put patient_input into patientlist.
        patient_input = line[7:].split(",")
        if patient_input not in patientlist:  # if patient is not in system this will record the patient and write it
            patientlist.append(patient_input)
            return output_file.write("Patient {} is recorded.\n".format(patientlist[-1][0]))
        elif patient_input in patientlist:  # If patient was already there it will write this in output file
            return output_file.write("Patient {} cannot be recorded due to duplication.\n".format(patient_input[0]))
```

Second picture is in output function and it calls the other 5 function and it locates under the 5 function.

```
    if command == "create":  # this is the part of the output function which takes the commands and calls the funct
        return create()
    elif command == "remove":
        return remove()
    elif command == "probability":
        return probability()
    elif command == "recommendation":
        return recommendation()
    elif command == "list":
        return list()
    else:
        pass
```

```
def remove():
    # this function will take the remove command and delete the patients in those lines.
    patients = str(line[7:-1])  # we get the name from line
    patients = patients.replace(" ", "")
    c = 0
    while c < len(patientlist):  # the loop will remove all the patients who was written beside remove
        if patients in patientlist[c]:
            patientlist.pop(c)  # this will delete the patient informations from system
            return output_file.write("Patient {} is removed.\n".format(patients))
        else:  # if patient isnt in system it will write this into output file
            c += 1
    return output_file.write("Patient {} cannot be removed due to absence.\n".format(patients))
```

```
def probability():
    # this function will calculate the probability of a person to get the cancer.
    patients = str(line[12:-1])  # this takes the patient name

    for j in range(len(patientlist)):  # this loop will calculate each patients probability and write it to file.
        pat = patientlist[j]
        if patients in pat:
            percentage = float(patientlist[j][1])
            people = int(patientlist[j][3][0:3])  # numerator of treatment incidence
            totalpeople = int(patientlist[j][3][-6:])  # denominator of treatment incidence
            probability = people / (people + ((1 - percentage) * totalpeople)) * 100
            probability = (float((str(probability))[0:5]))
            probab = format(probability,"g")  # this will erase the zeros after decimal point
            prob = output_file.write(
                "Patient {} has a probability of {}{} of having {}.\n".format(patients, probab,"%",
                                                                              patientlist[j][2]))
            return prob
    return output_file.write("Probability for {} cannot be calculated due to absence.\n".format(patients))
```

```
def recommendation():
# this function will consider the risk and the probability and decide whether the person should get treatment or not
    patients1 = str(line[15:-1])  # we take patients name
    pati1 = int(len(patientlist))
    for i in range(pati1):  # this loop will compare the risk and probability and write if patient should get
        pat1 = patientlist[i][0]                                                    # treatment or not
        if pat1 == patients1:
            percentage1 = float(patientlist[i][1])
            people1 = int(patientlist[i][3][0:3])
            totalpeople1 = int(patientlist[i][3][-6:])
            probability1 = people1 / (people1 + ((1 - percentage1) * totalpeople1)) * 100
            risk = int(float(patientlist[i][5]) * 100)
            if risk < probability1:  # if risk is lesser it writes for patient to have the treatment in output file.
                return output_file.write("System suggests {} to have the treatment.\n".format(patients1))
            elif risk > probability1:  # if risk is greater it writes not to have treatment in output file
                return output_file.write("System suggests {} NOT to have the treatment.\n".format(patients1))
            else:
                pass
        else:
            pass    # if patient is not in system it cant recommend anything and writes this into output file
    return output_file.write("Recommendation for {} cannot be calculated due to absence.\n".format(patients1))
```

The variables helps with the calculation and after calculating the probability correctly it will check the if statement and iiif risk is less than probability it will write “System suggests this patient to have the treatment else it will write “System suggests this patient not to have the treatment” into output file.Patient may not be in system so in this case it will write “Recommendation for this patient cannot be calculated due to absence.”

```
def list():  # this function will make a table into output file
    title = "Patient"
    title1 = "Diagnosis"  # for each title there are variables
    title2 = "Disease"
    title3 = "Disease"
    title4 = "Treatment"
    title5 = "Treatment"  # the distance is calculated according to the length of the words
    output_file.write(f"{title:<8}{title1:<14}{title2:<16}{title3:<12}{title4:<17}{title5:<10}\n")
    title6 = "Name"
    title7 = "Accurancy"
    title8 = "Name"
    title9 = "Incidence"
    title10 = "Name"
    title11 = "Risk"
    output_file.write(f"{title6:<8}{title7:<14}{title8:<16}{title9:<12}{title10:<17}{title11:<10}\n")
    x = "-" * 80
    output_file.write(f"{x}\n")

    for i in range(len(patientlist)):
        patient_name = patientlist[i][0]
        diagnosis_acc = (float(patientlist[i][1])) * 100
        diagnosis_accuracy = str(diagnosis_acc) + "%"
        disease_name = patientlist[i][2]
        disease_incidence = patientlist[i][3]
        treatment_name = patientlist[i][4]
        treatrisk = float(patientlist[i][5]) * 100
        treat_risk = format(treatrisk, "g")  # this will erase the zeros after decimal point
        treatment_risk = str(treat_risk) + "%"
        # the loop will take each patient from list and all their infos from list and write them into output file
        output_file.write(f"{patient_name:<8}{diagnosis_accuracy:<13}{disease_name:<16}{disease_incidence:<12}"
                          f"{treatment_name:<18}{treatment_risk}\n")
```

This function will write the titles first as two lines and after than it will make a “-“ string to split titles and informations.Then it will get into loop to write down each patient information into output file

```
def input_txt():  # for each command function calls the output function so it can achieve the target function

    if "create" in line:
        output_txt("create")
    elif "remove" in line:
        output_txt("remove")
    elif "probability" in line:
        output_txt("probability")
    elif "recommendation" in line:
        output_txt("recommendation")
    elif "list" in line:
        output_txt("list")
        pass
    else:
        global should_stop  # this will end the loop
        should_stop = True

with open("doctors_aid_inputs.txt", "a", encoding="utf-8") as file:
    file.write("\n")
output_file = open("doctors_aid_outputs.txt", "w", encoding="utf-8")
f = open("doctors_aid_inputs.txt", "r", encoding="utf-8")
should_stop = False
while not should_stop:  # the loop will call the input file and make input function read the lines
    line = f.readline()
    input_txt()
```

After calling input function the input function will call output function and the output function will call other functions.This is the first step of program to work.It will read the lines from input file after checking each line in if statements and after reading it will stop by changing the should_stop command as True.

USER CATALOGUE

If there are patients to be recorded the user should put those patient's informations with create command.The order should be like create/ patient name/ diagnosis accuracy/ disease name/ disease incidence/ treatment name/ treatment risk.If user wants to remove patients then he/she can write "remove (patient name)"in a line. If user wants to calculate the probability he/she should write "probability (patient name)" in a line and for recommendation "recommendation (patient name)".But if user wants a table which all the patient informations are in then user should write "list" in a line into doctors_aid_inputs.txt and save it.

GRADING

Evaluation	Points	Evaluate Yourself / 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	.25
Correctness	35	30
Report	20	20
There are several negative evaluations	...	...