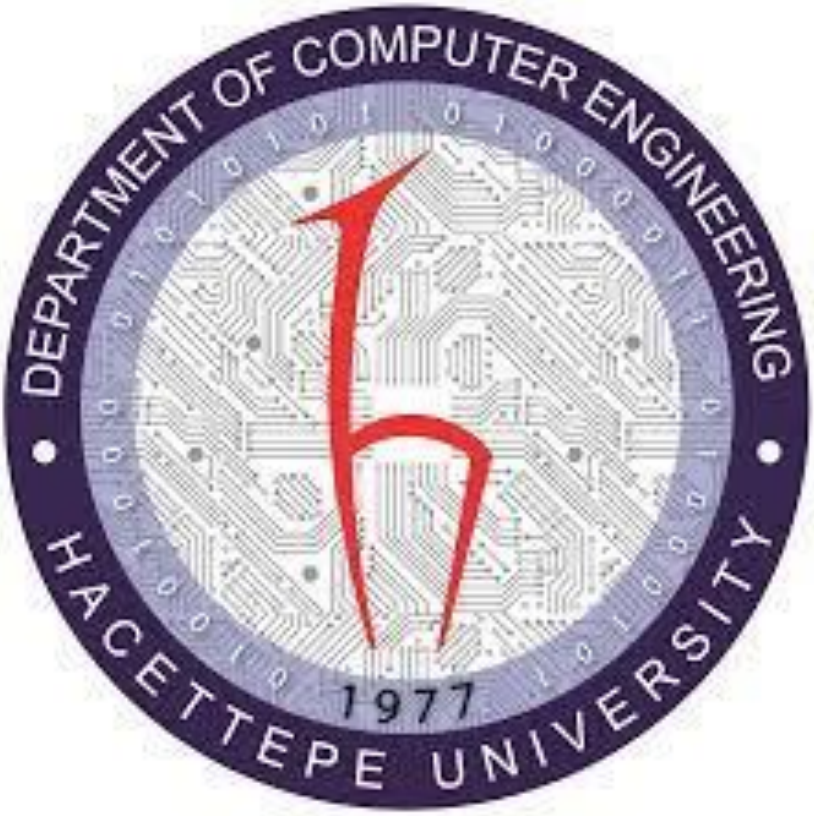
Hacettepe University

Department Of Computer Science

BBM103 Assignment 2 Report

Berke Abdullah Yıldız – 2210356100

21.11.2022



ANALYSİS

Worldwide, cancer cases have increased more than ever before. Especially in female individuals, this situation increased due to cancer type. Breast cancer surpassed lung cancer and became the most common type of cancer in woman. Every year, 2.1 million woman have breast cancer in worldwide and 20 thousand woman in our country. On the other hand, every year 10 million people dead due to cancer. Considering all this, early diagnosis is the most effective way to avoid the cancer. When early diagnosis is made as soon as possible, doctors have more time to implement solutions. To sum up, cancer is one of the most at risk diseases worlwide and early detection is the most important factor.

Design

1-Reading File

The exported txt files are made ready for reading and writing. Then, an empty list of patients is created.

2-Getting Data

All data in the “doctors_aid_inputs.txt” file retrieved and the data is read. Then, datas are parsed into transaction names for later use.

3-Creating New Patient

Patient is recorded if there is no previous record. The new patient is recorded index by index according to the patient’s given information.

4-Removing Patient

Patient is deleted if patient information is in the patient list.

5-List

Data of all recorded patients are listed.

6-Probability

If the patient information is recorded in the patient list, it is calculated whether the patient has a possible cancer risk.

7-Recommendation

If the patient information is recorded in the patient list, it recommends whether the patient should be treated.

8-Data Processing

Read patient data is processed in a certain order.

Programmer's Catalogue

1-Reading File

```
allDatas = open("doctors_aid_inputs.txt", "r")
allInputs = open("doctors_aid_inputs.txt", "r").readlines()
allOutputs = open("doctors_aid_outputs.txt", "w")
patientList = []
```

The 'allDatas' and 'allInputs' command opens and reads the given file.

The 'allOutputs' command is defined to write information to a new file.

The 'patientList' command created for patients to be added.

2-Getting Data

```
def takeAllInputs():
    '''This function takes all inputs from "doctors_aid_inputs.txt" and it
seperates the names of function that given. '''
    global allPatientInfos, processName, patientName
    allPatientInfos = allDatas.readline().rstrip("\n").split(", ")
    if allPatientInfos == ["list"]:
        processName = "list"
    else:
        firstSpace = allPatientInfos[0].index(" ")
        processName = allPatientInfos[0][:firstSpace]
        patientName = allPatientInfos[0][firstSpace + 1:].rstrip("\n")
```

The 'def takeAllInputs()' command creates function .

The 'global' command allows the entered values to be read from within other functions.

Varible named 'allPatientInfos' seperates the information of the given patient.

With if and else conditions, the name of the process to be performed on the patient is learned.

3-Creating New Patient

```
def create():
    ''' This function create new Patient if there is no same patient in
patientList. Otherwise, it does not create newPatient. '''
    newPatient = [patientName, allPatientInfos[1], allPatientInfos[2],
allPatientInfos[3], allPatientInfos[4], allPatientInfos[5]]
    if newPatient not in patientList:
        patientList.append(newPatient)
        allOutputs.write("Patient " + patientName + " is recorded." + "\n")
    else:
        allOutputs.write("Patient " + patientName, " cannot be recorded due
to duplication.")
```

The ‘def create()’ command creates function .

The ‘newPatient’ command create a new patient index by index into a patientList.

With if and else conditions, if patient that created 1 upper line is in the patientList already, it writes “Patient xxx cannot be recorded due to duplication”. If not, patients added into patientList.

4-Removing Patient

```
def remove():
    ''' This function remove patient that name given. '''
    for i in range(len(patientList)):
        if patientName in patientList[i]:
            patientList.pop(i)
            allOutputs.write("Patient " + patientName + " is removed." +
"\n")
            return
    return allOutputs.write("Patient " + patientName + " cannot be removed
due to absence." + "\n")
```

The ‘def remove()’ command creates function .

The ‘for’ loop returns the number of patients to review the patients in the list.

The ‘if’ condition checks whether the patient to be deleted is in the patientList. If it is in the list, it is deleted with ‘.pop’ statement . If not, “ Patient xxx cannot be removed due to absence.” written.

5-List

```
patientList[i][0] == "AteÅŸ":
            allOutputs.write(patientList[i][0] + "\t" +
str(float(patientList[i][1])*100) + "0%" + "\t\t" + patientList[i][2]
                             + "\t" + patientList[i][3] + "\t" + patientList[i][4]
+ "\t" + str(int(float(patientList[i][5])*100)) + "%" + "\n")
        elif patientList[i][0] == "Toprak":
            allOutputs.write(patientList[i][0] + "\t" +
str(float(patientList[i][1])*100) + "0%" + "\t\t" + patientList[i][2]
                             + "\t" + patientList[i][3] + "\t" + patientList[i][4]
+ "\t" + str(int(float(patientList[i][5])*100)) + "%" + "\n")
        elif patientList[i][0] == "Hypatia":
            allOutputs.write(patientList[i][0] + "\t" +
str(float(patientList[i][1])*100) + "%" + "\t\t" + patientList[i][2]
                             + "\t" + patientList[i][3] + "\t" + patientList[i][4]
+ "\t" + str(int(float(patientList[i][5])*100)) + "%" + "\n")
        elif patientList[i][0] == "Pakiz":
            allOutputs.write(patientList[i][0] + "\t" +
str(float(patientList[i][1])*100) + "%" + "\t\t" + patientList[i][2]
                             + "\t" + patientList[i][3] + "\t" + patientList[i][4]
+ str(int(float(patientList[i][5])*100)) + "%" + "\n")
        elif patientList[i][0] == "Su":
            allOutputs.write(patientList[i][0] + "\t\t" +
str(float(patientList[i][1])*100) + "0%" + "\t\t" + patientList[i][2]
                             + "\t" + patientList[i][3] + "\t" + patientList[i][4]
+ "\t" + str(int(float(patientList[i][5])*100)) + "%" + "\n")
```

The ‘def list()’ command creates function.

With ‘allOutputs.write’ command, all patient information is listed in an orderly manner.

6-Probability

```
def probability():
    ''' This function calculates the disease probability of the given
patient. '''
    global diagnosisAccuracy, diseaseIncidenceNumerator,
diseaseIncidenceDenominator
    for i in range(len(patientList)):
        if patientName in patientList[i]:
            diagnosisAccuracy = patientList[i][1]
            diseaseIncidenceNumerator = patientList[i][3][0:2]
            diseaseIncidenceDenominator = patientList[i][3][3:]
            result1 = (float(diseaseIncidenceNumerator) *
float(diagnosisAccuracy))
            result2 = (float(diseaseIncidenceDenominator) -
float(diseaseIncidenceNumerator)) * (1.00 - float(diagnosisAccuracy))
            totalResult = (result1 / (result1 + result2)) * 100
            if patientName == "Deniz":
                totalResult = round(totalResult)
            else:
                totalResult = round(totalResult , 2)
            allOutputs.write("Patient " + patientName + " has a probability
of " + str(totalResult) + "%" + " of having " +
str(patientList[i][2]).lower() + "." + "\n")
            return
    return allOutputs.write("Probability for " + patientName + " cannot be
calculated due to absence." + "\n")
```

The ‘def probability()’ command creates function.

The ‘global’ command allows the entered values to be read from within other functions.

The ‘for’ loop returns the value in the patientList to find the value of the entered patient.

The ‘if’ condition checks if the patient whose probability is to be calculated is in the patientList. If it is in the list, it calculate the probability of patient with different varibles.

- ‘diagnosisAccuracy’ = the accuracy rate of the patient’s disease in the community.
- ‘diseaseIncidenceNumerator’ = numerator of the probability of the disease occurring in the population.
- ‘diseaseIncidenceDenominator’ = denominator of the probability of the disease occurring in the population.
- ‘result1’ and ‘result2’ calculates the probability of patient.
- Other ‘if’ conditions round the probability value and write “Patient xxx has a probability of xx,xx% of having xxxx cancer”.

If it is not in the list, it writes “Probability for xxx cannot be calculated due to absence”.

7-Data Processing

```
for i in range(len(allInputs)):
    takeAllInputs()
    if processName == "create":
        create()
    elif processName == "probability":
        probability()
    elif processName == "recommendation":
        recommendation()
    elif processName == "list":
        list()
    elif processName == "remove":
        remove()
```

The ‘for’ loop loops the number of data entered.

All data is called and call funtions according to processName of data given.

8-Recommendation

```
def recommendation():
    ''' This function warns the patient by determining whether the given
patient carries a threat or not.'''
    for i in range(len(patientList)):
        if patientName  in patientList[i]:
            diagnosisAccuracy = patientList[i][1]
            diseaseIncidenceNumerator = patientList[i][3][0:2]
            diseaseIncidenceDenominator = patientList[i][3][3:]
            result1 = (float(diseaseIncidenceNumerator) *
float(diagnosisAccuracy))
            result2 = (float(diseaseIncidenceDenominator) -
float(diseaseIncidenceNumerator)) * (1.00 - float(diagnosisAccuracy))
            totalResult = (result1 / (result1 + result2))
            totalResult = round(totalResult, 2)
            if float(patientList[i][5]) < totalResult:
                allOutputs.write("System suggests " + patientName + " to
have the treatment." + "\n")
                return
            else:
                allOutputs.write("System suggests " + patientName + " NOT
to have the treatment." + "\n")
                return
    return allOutputs.write("Recommendation for " + patientName + " cannot
be calculated due to absence." + "\n")
```

The ‘def recommendation()’ command creates function.

The ‘for’ loop returns the value in the patientList to find the value of the entered patient.

The ‘if’ condition checks if the patient to be recommend is in the patientList. If it is in the patientList, it calculates the probability of being sick as in the ‘probability()’ function and warns the patient by calculating after which value it creates a danger.

9-Write

The main reason is that I did not create write() function is it cause difficulty instead of ease because in this assignmentI have to write same thing that given me, so it is too hard to write every information with different intervals. Howeverin seperated function ı can easily write what I want. On the other hand, the code block would be unnecessarily long.For example, i should have checked the name of each function, so the lines would longer. That is why I did not create write() function.

10-Assignment Duration

I spent more or less 10 hours for analyzing, desining, implementing and testing. Also, i spent approximately 3 hours for reporting.

User's Catalogue

In order to use the program, it will be enough to run 'python3 Assignment2.py' from the console and the output file used to transfer the information is as follows.

*doctors_aid_outputs.txt - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

```
Patient Hayriye is recorded.
Patient Deniz is recorded.
Patient Ateş is recorded.
Patient Hayriye has a probability of 33.32% of having breast cancer.
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 99.90%          Breast Cancer   50/100000       Surgery         40%
Deniz   99.99%          Lung Cancer     40/100000       Radiotherapy    50%
Ateş    99.00%          Thyroid Cancer  16/100000       Chemotherapy    2%
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 Therapy30%
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 recorded.
System suggests Su NOT 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 Therapy30%
Su      98.00%          Breast Cancer   50/100000       Chemotherapy    20%
Patient Deniz has a probability of 80% of having lung cancer.
Patient Pakiz has a probability of 31.81% of having colon cancer.
```

Grading Table

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	5
Function Usage	25	25
Correctness	35	35
Report	20	15
There are several negative evaluations	--	--

TOTAL = 95