Deborah Barndt

ITMD 513 Open Source Programming

Professor Dr. Sam

Final

4-29-19

'''

Deborah Barndt

4-29-19

HospitalStay.py

final: Patient Fees - using Graphical User Interface (GUI)

This program will create menu that allows the user to enter a type of surgery

and a type of medication, and check the patient out of the hospital. When the

patient checks out, the total charges will be displayed in a window format. The

GUI program will compute a patient's bill for a hospital stay. The different

components of the program are: the PatientAccount class, the Surgery class, and

the Pharmacy class.

The PatientAccount class will keep a total of the patient's charges, and will

also keep track of the number of days spent in the hospital. The group must

decide on the hospital's daily rate. The Surgery class will have stored within

it the charges for at least five types of surgery, and it can update the

charges variable of the PatientAccount class. The Pharmacy class will have

stored within it the price of at least five type of medication, and it can

update the charges variable PatientAccount class.

Written by Deborah Barndt.

'''

from tkinter import \*

from tkinter import ttk

from tkinter import messagebox

import re

from datetime import datetime

from datetime import date

# PatientAccount class that will keep a total of the patient's charges.

class PatientAccount:

# Constructor function for the patient's information such as their first name,

# last name, date of birth, hospital daily rate, days in the hospital, and the

# hospital charges.

def \_\_init\_\_(self):

self.\_\_patientFirstName = None

self.\_\_patientLastName = None

self.\_\_patientDOB = None

self.\_\_hospDailyRate = 3949.95

self.\_\_numDaysInHosp = 0

self.\_\_hospCharges = {}

# Function to get the patient's first name.

def getPatientFirstName(self):

return self.\_\_patientFirstName

# Function to set the patient's first name.

def setPatientFirstName(self, patientFirstName):

self.\_\_patientFirstName = patientFirstName

# Function to get the patient's last name.

def getPatientLastName(self):

return self.\_\_patientLastName

# Function to set the patient's last name.

def setPatientLastName(self, patientLastName):

self.\_\_patientLastName = patientLastName

# Function to get the patient's date of birth (DOB).

def getPatientDOB(self):

return self.\_\_patientDOB

# Function to set the patient's date of birth (DOB).

def setPatientDOB(self, patientDOB):

self.\_\_patientDOB = patientDOB

# Function to get the hospital's daily rate.

def getHospDailyRate(self):

return self.\_\_hospDailyRate

# Function to set the hospital's daily rate.

def setHospDailyRate(self, dailyRate):

self.\_\_hospDailyRate = dailyRate

# Function to get the number of days the patient has been in the hospital.

def getNumDaysInHosp(self):

return self.\_\_numDaysInHosp

# Function to set the number of days the patient has been in the hospital.

def setNumDaysInHosp(self, numDays):

self.\_\_numDaysInHosp = numDays

# Function to get the patient's charges for their hospital stay.

def getHospCharges(self):

return self.\_\_hospCharges

# Function to set the patient's charges for their hospital stay.

def setHospCharges(self, dailyRate, numDays, proCost, medCost):

self.\_\_hospTotal = (dailyRate \* numDays) + proCost + medCost

self.\_\_hospCharges = self.\_\_hospTotal

# Surgery class that will have stored the charges for at least five types of surgery.

class Surgery:

# Constructor function for the types of surgery procedures and prices.

def \_\_init\_\_(self):

self.\_\_procedures = {

'Appendectomy': 1500,

'Breast Biopsy': 500,

'Coronary Artery Bypass': 40000,

'Hip Replacement': 39299,

'Knee Replacement': 49500,

'Tonsillectomy': 8500,

'Thyroidectomy': 10824

}

self.\_\_procedures = None

self.\_\_price = 0

# Function to get the surgery procedures for the patient.

def getProcedures(self):

return self.\_\_procedures

# Function to set the surgery procedures for the patient.

def setProcedures(self, procedures):

self.\_\_procedures = procedures

# Pharmacy class that will have stored the price of at least five types of medication.

class Pharmacy:

#Constructor function for the types of medications and prices.

def \_\_init\_\_(self):

self.\_\_medications = {

'Amoxil': 16.94,

'Deltasone': 113,

'Levoxyl': 81,

'Lipitor': 165,

'Neurontin': 241,

'Vicodin': 114.85,

'Zestril': 1257

}

# Function to get the medications for the patient.

def getMedications(self):

return self.\_\_medications

# Function to set the medications for the patient.

def setMedications(self, medications):

self.\_\_medications = medications

# Function to contain the surgery objects.

def surgeryCost(surgery, newPrice):

procedure = [

'Appendectomy',

'Breast Biopsy',

'Coronary Artery Bypass',

'Hip Replacement',

'Knee Replacement',

'Tonsillectomy',

'Thyroidectomy'

]

price = [1500, 500, 40000, 39299, 49500, 8500, 10824]

if(surgery == procedure[0]):

newPrice = price[0]

elif(surgery == procedure[1]):

newPrice = price[1]

elif(surgery == procedure[2]):

newPrice = price[2]

elif(surgery == procedure[3]):

newPrice = price[3]

elif(surgery == procedure[4]):

newPrice = price[4]

elif(surgery == procedure[5]):

newPrice = price[5]

elif(surgery == procedure[6]):

newPrice == price[6]

return newPrice

# Function to contain the pharmacy objects.

def pharmacyCost(pharmacy, newMedPrice):

medications = [

'Amoxil',

'Deltasone',

'Levoxyl',

'Lipitor',

'Neurontin',

'Vicodin',

'Zestril'

]

price = [16.94, 113, 81, 165, 241, 114.85, 1257]

if(pharmacy == medications[0]):

newMedPrice = price[0]

elif(pharmacy == medications[1]):

newMedPrice = price[1]

elif(pharmacy == medications[2]):

newMedPrice = price[2]

elif pharmacy == medications[3]:

newMedPrice = price[3]

elif pharmacy == medications[4]:

newMedPrice = price[4]

elif pharmacy == medications[5]:

newMedPrice = price[5]

elif pharmacy == medications[6]:

newMedPrice = price[6]

return newMedPrice

# Function to calculate the hospital charges.

def calcCharge():

newPrice = 0

newMedPrice = 0

patient = PatientAccount()

patientsurgery = Surgery()

patientpharmacy = Pharmacy()

firstname = str(fNameText.get(1.0, END))

lastname = str(lNameText.get(1.0, END))

#dob = datetime.date(dobText.get(1.0, END))

dob = str(dobText.get(1.0, END))

try:

numDays = float(daysText.get(1.0, END))

except ValueError as e:

messagebox.showerror("Error", "Number of days in hospital must be a number.")

return

'''try:

datetime.strftime(dob, "%m/%d/%Y")

except ValueError as err:

messagebox.showerror("Error", "Date of birth format must be MM/DD/YYYY.")

return

'''

# Check the validation of the user input.

if(firstname.strip() == ''):

messagebox.showerror('Error', 'First name cannot be empty.')

if(lastname.strip() == ''):

messagebox.showerror('Error', 'Last name cannot be empty.')

if(dob.strip() == ''):

messagebox.showerror('Error', 'Date of birth cannot be empty.')

if(numDays < 0):

messagebox.showerror('Error', 'Cannot enter negative amount of days.')

else:

patient.setPatientFirstName(firstname)

patient.setPatientLastName(lastname)

patient.setPatientDOB(dob)

patient.setNumDaysInHosp(numDays)

dailyRate = float('{0:2f}'.format(patient.getHospDailyRate()))

totalRate = dailyRate \* numDays

surgery = str(surgeryList.get())

patientsurgery.setProcedures(surgery)

pharmacy = str(pharmacyList.get())

patientpharmacy.setMedications(pharmacy)

proCost = float('{0:2f}'.format(surgeryCost(surgery, newPrice)))

medCost = float('{0:2f}'.format(pharmacyCost(pharmacy, newMedPrice)))

patient.setHospCharges(dailyRate, numDays, proCost, medCost)

totalBill = float('{0:2f}'.format(patient.getHospCharges()))

# Display area for patient checkout and display in a pop-up messagebox.

messagebox.showinfo('Mercy Hospital Total Bill', 'First Name: ' +

patient.getPatientFirstName() + 'Last Name: ' +

patient.getPatientLastName() + 'Date of Birth: ' +

patient.getPatientDOB() + 'Rate: $' +

str('%.2f' % totalRate) + '\n' + patientsurgery.getProcedures() + ' $' +

str('%.2f' % proCost) + '\n' + patientpharmacy.getMedications() + ' $' +

str('%.2f' % medCost) + '\nTotal Bill: $' + str('%.2f' % totalBill))

# Create an instance for window to display.

window = Tk()

# Create the GUI window title, dimensions, and color.

window.title('Mercy Hospital Patient Fees')

window.geometry('400x260')

window.config(background='#9DBAFF')

window.iconbitmap('hospitalcross.ico')

window.option\_add('\*Dialog.msg.font', 'Franklin Book 16')

# Labels for the Patient Account information.

fNameLabel = Label(window, text = 'First Name: ', background='#9DBAFF', font=('Franklin Book', 10))

fNameLabel.grid(row = 0, column = 0, sticky = W+E)

lNameLabel = Label(window, text = 'Last Name: ', background='#9DBAFF', font=('Franklin Book', 10))

lNameLabel.grid(row = 1, column = 0, sticky = W+E)

dobLabel = Label(window, text = 'Date of Birth: ', background='#9DBAFF', font=('Franklin Book', 10))

dobLabel.grid(row = 5, column = 0, sticky = W+E)

surgeryLabel = Label(window, text = 'Surgery: ', background='#9DBAFF', font=('Franklin Book', 10))

surgeryLabel.grid(row = 6, column = 0, sticky = W+E)

pharmacyLabel = Label(window, text = 'Medication: ', background='#9DBAFF', font=('Franklin Book', 10))

pharmacyLabel.grid(row = 7, column = 0, sticky = W+E)

daysLabel = Label(window, text = 'Days in the Hospital:', background='#9DBAFF', font=('Franklin Book', 10))

daysLabel.grid(row = 8, column = 0, sticky = W+E)

# Text fields for the user input.

fNameText = Text(window, width = 15, height = 1, font=('Franklin Book', 10))

fNameText.grid(row = 0, column = 1, padx = 5, pady = 5, ipady = 2, sticky = W+E)

lNameText = Text(window, width = 15, height = 1, font=('Franklin Book', 10))

lNameText.grid(row = 1, column = 1, padx = 5, pady = 5, ipady = 2, sticky = W+E)

dobText = Text(window, width = 15, height = 1, font=('Franklin Book', 10))

dobText.grid(row = 5, column = 1, padx = 5, pady = 5, ipady = 2, sticky = W+E)

daysText = Text(window, width = 15, height = 1, font=('Franklin Book', 10))

daysText.grid(row = 8, column = 1, padx = 5, pady = 5, ipady = 2, sticky = W+E)

# Drop down menu for the list of surgeries.

surgeryChoice = StringVar()

surgeryList = ttk.Combobox(window, textvariable = surgeryChoice, font=('Franklin Book', 10))

surgeryList['values'] = ('Appendectomy', 'Breast Biopsy', 'Coronary Artery Bypass', 'Hip Replacement',

'Knee Replacement', 'Tonsillectomy', 'Thyroidectomy')

surgeryList.current(0)

surgeryList.grid(row = 6, column = 1, padx = 5, pady = 5, ipady = 2, sticky = W)

# Drop down menu for the list medications.

pharmacyChoice = StringVar()

pharmacyList = ttk.Combobox(window, textvariable = pharmacyChoice, font=('Franklin Book', 10))

pharmacyList['values'] = ('Amoxil', 'Deltasone', 'Levoxyl', 'Lipitor',

'Neurontin', 'Vicodin', 'Zestril')

pharmacyList.current(0)

pharmacyList.grid(row = 7, column = 1, padx = 5, pady = 5, ipady = 2, sticky = W)

# Checkout button to check out patient from the hospital.

calcBtn = Button(window, text = 'Check Out', font=('Franklin Book', 10), width = 10, command = calcCharge)

calcBtn.grid(row = 9, column = 1, padx = 5, pady = 3, sticky = W+E)

# Create an event loop to display the GUI.

window.mainloop()

Output Result:

















