#Making a Ferragamo shopping list

# 1. display, and append the inventory count ( display item together and individually)

# 2. calcuate the billing item

# 3. display the discounted percentage of the items

# 4. able to record the total sales generated

import os

import csv

dir\_path = os.path.dirname(os.path.realpath(\_\_file\_\_))

filepath = ''

#To open the CSV file and read it

for root, dirs, files in os.walk(dir\_path):

for file in files:

if file == "inventories.csv":

filepath += root +'\\' + str(file)

f = open(filepath,'r')

readr = csv.reader(f)

#The data in the csv file will be transfered into python as a dictionary item

itemDict = {}

for row in readr:

itemDict[row[0]]= [row[1],row[2],row[3],row[4],row[5]]

# for this, the main key of the dictionary will be the model number, and he values will be stored in a list of model name, price, quantity, colour, and size

def showitems(item, feature):

# This will be the main function for other function that requires a display of items

return('Model Number : ' + str(item) + ' | name : ' + str(feature[0])+ ' | colour : ' + str(feature[3]) + '| size : ' +str(feature[4]) + '| Qty : ' + str(feature[2]))

def displayItems(list):

# This function is used to display item in a format of model name, colour and size only. This remove other features that customer are not interested in

product\_type = '6.5'

# For this function to work, I've decided to print all the models but only the size 4 items, as all the shoe has a size 6.5 in common, this will only show 1 size for each model, preventing any duplicates in the list

print('Item listing \n ------------------')

for item, feature in list.items():

if product\_type == feature[4]:

print(showitems(item, feature).replace('Model Number : ' + str(item),'').replace('| size : ' +str(feature[4]) + '| Qty : ' + str(feature[2]),'') + ' | price : $' + str(feature[1]))

print('------------------')

def displayitem\_no(list):

#function display items by its model number, this list will only have 1 item showing

while True:

product\_type = str(input('enter the model number you like to view: '))

if product\_type == 'end':

break

else:

for item, feature in list.items():

if product\_type == item:

print(showitems(item,feature).replace('Model Number : ' + str(item), ''))

def displayname(list):

#This display item by its name, usually, this will be the most common way of navigating through the inventories

while True:

product\_type = str(input('enter the name you like to view: '))

if product\_type == 'end':

break

else:

for item, feature in list.items():

if product\_type == feature[0]:

print(showitems(item, feature).replace(' | name : ' + str(feature[0]), ''))

def displaysize(list):

# This display all the models by the size the user type in. This function is useful for consumers that only have a special size that they can wear. Hence searching by size, allow employees to pull out all the models of the consumers size for the consumers

while True:

product\_type = str(input('enter the size you like to view: '))

if product\_type == 'end':

break

else:

for item, feature in list.items():

if product\_type == feature[4]:

print(showitems(item, feature).replace('| size : ' +str(feature[4]), ''))

def displaycolour(list):

# Consumers have different taste and prefernce, some would only like to buy shoes of their favourite colour. With this function, it allows user to list all the inventories that's the same colour as what the user has inputted

while True:

product\_type = str(input('enter the colour you like to view: '))

if product\_type == 'end':

break

else:

for item, feature in list.items():

if product\_type == feature[3]:

print(showitems(item, feature).replace(' | colour : ' + str(feature[3]), ''))

def displaystock(list):

#This is the main function for displaying all the stocks, if user were to choose a specific method of listing the inentories, other function will operate.

viewing\_type = str(input('Choose what would you like to view (model number / name / size / colour / or all): '))

if viewing\_type == 'all':

print('Stock Count list \n ------------------')

for item, feature in list.items():

print(showitems(item, feature))

print('-----------------------')

elif viewing\_type == 'model number':

displayitem\_no(list)

elif viewing\_type == 'name':

displayname(list)

elif viewing\_type == 'size':

displaysize(list)

elif viewing\_type == 'colour':

displaycolour(list)

def billing(list):

# This is the billing function that charges the consumers for what they have purchased

total\_bill = 0

while True:

displayItems(itemDict)

model\_type = str(input('model name: '))

colour\_type = str(input('colour: '))

if model\_type == 'end':

break

else:

for item, feature in itemDict.items():

if model\_type == feature[0] and colour\_type == feature[3]:

print(showitems(item, feature).replace(' | name : ' + str(feature[0])+ ' | colour : ' + str(feature[3]),'') + ' | price : $' + str(feature[1]))

user\_item = str(input('Please key in the model number: '))

# Most stores will usually scan the item model when processing a transaction, as the model number includes the model type, the size and the colour

if user\_item in list:

try:

no\_item = int(input('Enter the number of item purchased: '))

except ValueError:

print('Please enter a number')

else:

quantity = list[user\_item][2]

quantity = int(quantity)

# This will remove update the quantity left automatically, so when a consumer purhcase X amount of item, it will minus X amount from the inventory list

if no\_item > int(quantity):

print('Sorry, insufficient amount')

else:

quantity -= no\_item

chosen\_item = list.get(user\_item)

total\_bill += float(chosen\_item[1])\*no\_item

# It calcualates and total up the inventory purchased and it will be shown at the end

list[user\_item][2] = str(quantity)

proceed = str(input('Would you like to process another transaction? (yes/no) : '))

if proceed == 'no':

print("--------------- \nTotal bill is : ${}\n--------------".format(round(total\_bill,3)))

break

else:

print('Item not found')

while True:

# The main operation of this program, it is the start and the end of the program.

activity = str(input('Would you like to view the stock or make a bill (view/bill): '))

if activity == 'end':

print('Thank you and have a nice day')

f.close()

break

else:

if activity == 'view':

displaystock(itemDict)

elif activity == 'bill':

billing(itemDict)