INTRODUCTION TO PROGRAMMING –

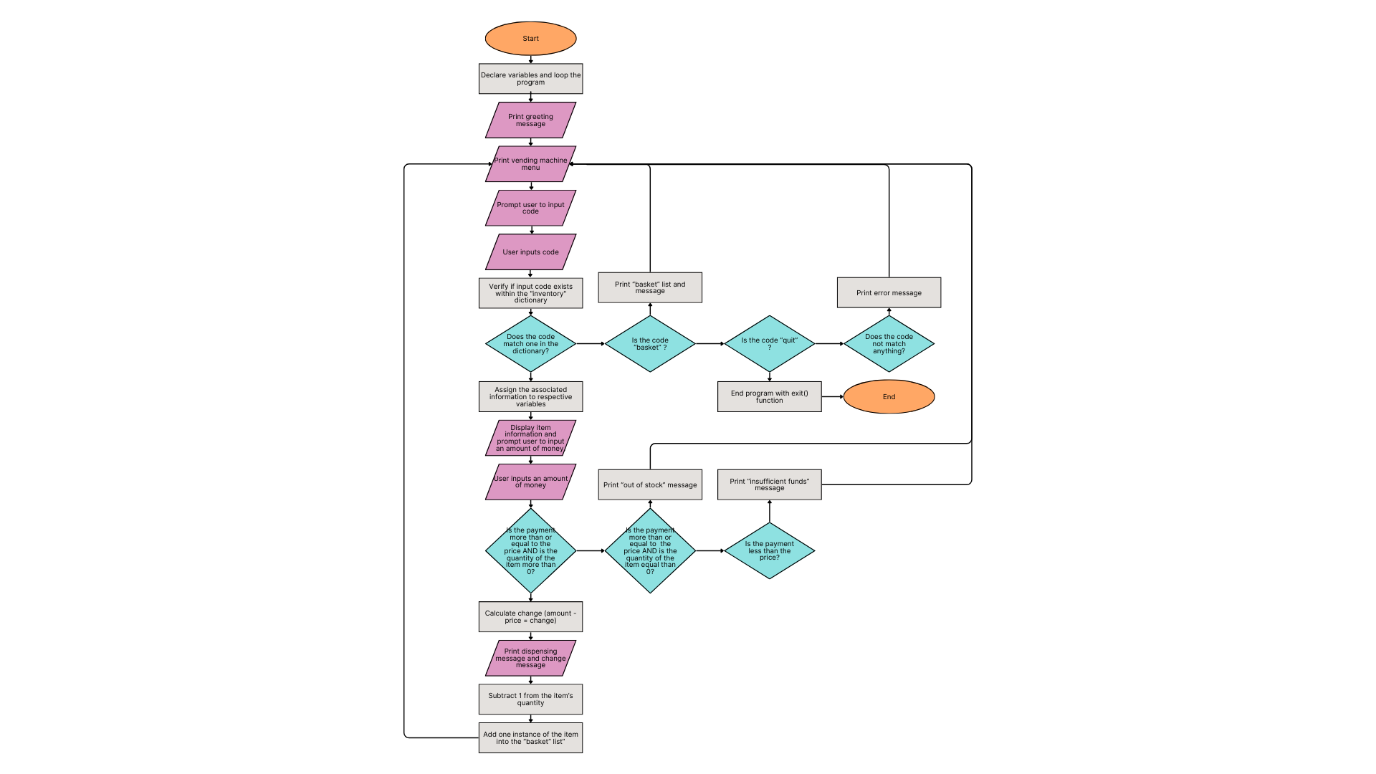
2ND SUMMATIVE ASSESSMENT: UTILITY APP

By Marcopolo John P. Deocampo

**Specifications**

The project this document will cover is a utility app designed to function similarly to a vending machine. This vending machine application is created entirely with the Python coding language, as well as must include these features listed below:  
- A menu consisting of snacks and drinks; The number and range of items is optional.  
- A set of codes that will allow a user to select a product of their choice.  
- A system to manage money in terms of input and change.  
- A message informing the user that their product has been dispensed.  
- A message informing the user of the amount of change they will receive.  
Additionally, optional features may be added to the application in order to attain extra marks.

**Flowchart**



To briefly explain the content of this flowchart, the program begins by declaring the necessary variables for the process, sets to program into an “infinite” loop, and then prints out a greeting message, followed by the actual application. The application itself functions like a normal vending machine: displaying products, selecting products, inserting money, product inventory, and the dispensing and change messages are what one can expect of this program. However, this application has two unique functions. When prompted to insert a code for a product, the user may type the words “basket” or “quit” to prompt separate program loops: “basket” makes the program print a list of the products the user has purchased during the session, and “quit” is how the program is manually stopped. Alternatively, a “basket” list is also displayed upon “quitting” the program, along with a goodbye message.

**Technical Description and Walkthrough**

<https://youtu.be/20iZyiqorfA>

**Critical Reflection**

Overall, I believe that I have applied my knowledge well into making this application. The program is made to mimic the functions of a vending machine, and in these respects, I believe I have made it function as it should. However, a vending machine should realistically operate only once per order, and while it would be ideal to have the program do the same, it wouldn’t allow for prolonged usage or testing: two things I felt were lacking in this program. In the end, I decided on looping the program so that I could integrate new features that would make the user experience more interesting.  
Among the two unique features I added to the application, the “basket” function was my favourite addition. The idea of an “Add to Cart” feature added to a vending machine was interesting to me, but in the end, I decided to just have a “receipt” of sorts be available to the user while they use the program, as well as after they’ve finished their session.  
On the other hand, the way that this program is set into a loop is a bit awkward. As previously mentioned, the program would go into a state where it would “infinitely” loop, and the user would have to manually break the loop. This function had been integrated into the code in a way that I was not satisfied with, and I believe that it could have been more refined with better application of loop concepts.  
Finally, I think that I could have improved the user experience for the application by adding more convenience into the program. In this iteration, if the user wishes to buy multiple instances of a product, they will have to buy individual instances over multiple loops. This would be much better if an amount option were to be included, which could be easily integrated using input codes.

**Appendix**

#Variable Declaration

inventory = {"A01":{"item":"Chocolate Bar","price":2,"stock":10},

         "A02":{"item":"Protein Bar","price":5,"stock":10},

         "A03":{"item":"Beef Jerky","price":5,"stock":10},

         "A04":{"item":"Candy Strips","price":2,"stock":10},

         "B01":{"item":"Assorted Nuts","price":3,"stock":10},

         "B02":{"item":"Cheesy Chips","price":3,"stock":10},

         "B03":{"item":"Pretzel Sticks","price":3,"stock":10},

         "B04":{"item":"Gummy Bears","price":3,"stock":10},

         "C01":{"item":"Water Bottle","price":1,"stock":10},

         "C02":{"item":"Soda Can","price":2,"stock":10},

         "C03":{"item":"Iced Tea","price":2,"stock":10},

         "C04":{"item":"Juice bag","price":2,"stock":10}

}

loop = True #Turns the program into an infinite loop

basket = [] #Stores the items that the user has purchased

#Greeting

print("Welcome to the Simple Vending Machine!\nMy job is to serve.\n\n")

#Menu Display

def menu():

    print("Wanna buy something?\nTIP: type 'basket' to see what you've bought, and 'quit' to exit the program.")

    #^Additional instructions for other features

    for code, info in inventory.items():

        print(f"{code} > Item: {info["item"]} | Price: {info["price"]} AED | {info["stock"]} in stock.")\

    #^Displays the menu

#Vending machine program

def vending():

    code = input(str("ENTER CODE: "))

    if code in inventory :

        item = inventory[code]["item"]

        price = inventory[code]["price"]

        quantity = inventory[code]["stock"]

        #^Assigning values from the dictionary to variables for convenience

        print(item.upper(),"\nPRICE:",price,"AED")

        payment = float(input("Please insert your money. AED "))

        #Asking the user for money as a float input

        if payment >= price and quantity > 0:

            change = payment - price

            print("\nDispensing the "+item+", please collect your change of",change,"AED.\n\n")

            inventory[code]['stock']-=1

            basket.insert(0,item)

        #^Calculating for change, stock changes, and adding items to the "basket"

        elif payment >= price and quantity == 0:

            print("Oops! We've run out of that item!\n")

        else:

            print("Oops! Insufficient Funds!\n")

    elif code == "quit":

        print("\nThank you for using the Simple Vending Machine!\nDuring this session, you bought: "

              +str(basket))

        exit()

        #^End message and breaking the loop by stopping the program

    elif code == "basket":

        print("\nHere's what you've bought so far:\n"+str(basket)+"\n")

        #^Displays the user's "basket"

    else:

        print("Oops! We've encountered an error!\n")

#MAIN LOOP

while loop == True:

 menu()

 vending()