Vending Machine: A program written in Python

[redacted] University

Brief

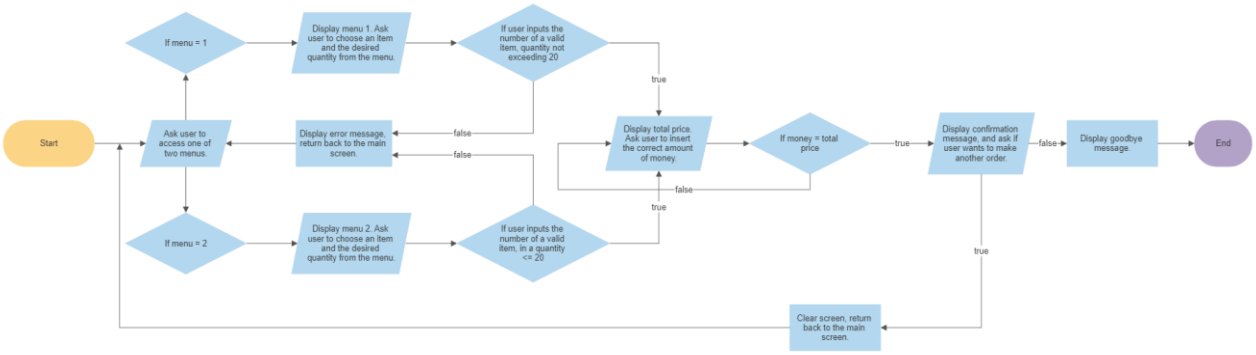
I, as a student of [redacted] in [redacted], have been tasked with creating a program that mimics a vending machine. Over the course of the semester, we have been taught to understand many of the basic programming concepts so that they will act as a secure foundation of our coding knowledge, and with that, to use those concepts as building blocks for one of the many ambitious projects we'll undertake. This vending machine is one of them.

Specification

This vending machine program was made using Python, the programming language that was taught to us. It required a set of specifications that would make it recognizable as a vending machine, and a heavy emphasis was placed on having a cohesive user interface that would look and function just like a vending machine in the real world.

- These specifications go over the basic functions, such as:
- Presenting a menu of snacks and drinks to the user and being able to receive their input.
- Performing money calculations so that the right amount is exchanged between the user and the machine.
- Displaying messages to highlight what it is doing.
- And other things that expand on these functions in detail.

Flowchart



This flowchart highlights the basic idea of every step of the process. From start to finish, the user is informed with what decisions they should make in a clear and concise way, and I find that using this chart has helped me understand my own idea on a simpler level.

Technical Description

The vending machine starts off with a main menu screen, with two sections (called submenus) that each contain a list of snacks and drinks. At the very beginning of the code, the datetime module was imported and used to display the current date and time. This is the only part of the program that is not inside a loop, which means that it will only appear once. That is, when the program is started.

The rest of the program, however, is contained inside a while loop, so that the user is given a chance to go back to the beginning by following choices that lead to breaks within the loop. There is only one instance in which the loop leads to a continue, reverting the user back to

the previous step.

The most apparent method that is prevalent in this program is the use of if-else statements. Almost every statement contains a message asking for an input, with the result of this input leading to the next statement. This is mainly how the vending machine communicates with the user.

Likewise, there is a lot of dialogue that is involved throughout the process that is meant to allow the user to communicate with the machine. This is accomplished through an appropriate use of print() and input() statements. It mostly accepts integers as valid arguments, since those integers correspond to the selections the user has to go through in the menus. Occasionally accepting letters as user input, the program also asks the user yes and no questions.

The program makes good use of variables to store and provide the information that it has.

The dictionary() function was used as a container for menu items, and the variables that stored those dictionaries were used to display those items on their respective submenus, and let the user access any item they wanted, along with its price, within them. The submenus especially took advantage of this so that money calculations can be performed with the price data that it provided in a relatively efficient way. The initial menu, on the other hand, was simple enough to be displayed by a single print message.

Finally, the vending machine uses basic arithmetic operators to manage the user's money. Condition checks that make sure the user was inputting valid numbers are made possible using comparison operators. The utility that variables can offer extends to this area as well with the use of string concatenation, which streamlines the coding process and makes it look more readable.

Critical Reflection

The nitty-gritty of bringing your program to life involves an element of considerable passion, especially for a project of this scale. Designing a vending machine that was going to perform various functions and keep track of lots of information was a challenging task for a beginner like me. Although the premise was simple and not very demanding in coding experience, leaving a lot of room for creativity, the complexity that I had to deal with while creating this program was ever so heightened by the level of detail that I had to work with. Such is the case for projects.

The idea I had in mind when I was figuring out how to tackle the execution of this program, was to make something that was easy enough, for the sake of the end user and I, to use and understand. Not too easy, however, because it would result in inefficient, dull and uninteresting code, bringing out a less-than-ideal, potentially awkward experience. I know as a beginner that I didn't have to worry about thinking that way since I didn't have that many tools in my arsenal to begin with, but I thought it was worth the chance to make something beautiful anyway.

To expand on a point I previously mentioned, I built my vending machine program with simplicity in mind. For me, it was a one-track process wherein each step had a simple reason or action behind it, both inside the code and outside, where I wanted it to appear user-friendly and welcoming.

I would also like to mention that I touched on the names of those snacks with a bit of creativity, bringing about a hint of novelty to them. In this part, I briefly took inspiration from an imaginary time period in the far future where vending machines would look very slick and stylish with one big touchscreen serving as its interface, like any machine you'd expect to see in a sci-fi movie. All of these snacks are inspired by their real-world counterparts as well, in one way or another.

In terms of improvement, I knew that my program was going to leave a lot to be desired. There are many other functionalities that I could've added with my current skillset, and I see how I could write cleaner, more optimized code. In any case, I am still very happy with the outcome.

Appendix

```
import datetime
import os
import time
now = datetime.datetime.now()
t = now.strftime("%A, %B %d %Y")


v1 = {"1": 7, "2": 5, "3": 3, "4": 3, "5": 10}
v2 = {"1": 10, "2": 3, "3": 5, "4": 5, "5": 7}


def change(a, b):
    return a % b


print("\n")
print(t)
print("\nHello. I am a vending machine. It is currently",
now.strftime("%H:%M."))
while True:
    print("\n-- 1. Comforting Classics --\n-- 2. Futuristic Flavors --
\n")

    cat = input("Select your category of choice (1-2, q to quit): ")

    if cat == "1":
```

```
        print("\n1. Snacker - $", v1["1"], "\n2. KatKat - $", v1["2"],
"\n3. Marbles - $", v1["3"], "\n4. Mr. Pepper - $", v1["4"], "\n5. 1up
- $", v1["5"], "\n")

        ch = input("Enter the number of the item you'd like to
purchase (1-5): ")

        am = int(input("Specify the desired amount: "))

        for item in v1.keys():

            item = ch

            val = (v1[item])

            tt = val * am

            if ch in v1.keys() and am <= 20:

                print(f"That'll be ${tt} for {am}.")

                money = 0

                for money in range(tt):

                    money = int(input("\nPlease insert your money: "))

                    if money < tt:

                        print("Sorry. You did not insert enough funds
for this item. Please try again.\n")

                        continue

                    else:

                        chn = change(money, val)

                        print(f"Thank you. Your item has been
dispensed, and you will receive a change of ${chn}.")

                        retry = input("Would you like to make another
order? (y-n) ")

                        if retry == "n":

                            print("Okay. Goodbye.")

                            quit()

                        else:
```

```
                            print("Okay. Going back to the main
menu...")
                            time.sleep(2)



                            os.system('cls')
                            break
                    break
                else:
                    print("Sorry. The item you were looking for is
unavailable or in insufficient amounts.")
                    time.sleep(1)
                    os.system('cls')
                    break



    elif cat == "2":
        print("\n1. Intergalactic Icecream - $", v2["1"], "\n2.
Overnight Oats - $", v2["2"], "\n3. Flavourful Feta Cup - $", v2["3"],
"\n4. Cheesy Chips - $", v2["4"], "\n5. Tasty Thing - $", v2["5"],
"\n")
        ch = input("Enter the number of the item you'd like to
purchase (1-5): ")
        am = int(input("Specify the desired amount: "))
        for item in v2.keys():
            item = ch
            val = (v2[item])
            tt = val * am
            if ch in v2.keys() and am <= 20:
                print(f"That'll be ${tt} for {am}.")
                money = 0
                for money in range(tt):
```

```
                    money = int(input("\nPlease insert your money: "))
                    if money < tt:
                        print("Sorry. You did not insert enough funds
for this item. Please try again.\n")



                        continue
                    else:
                        chn = change(money, val)
                        print(f"Thank you. Your item has been
dispensed, and you will recieve a change of ${chn}.")
                        retry = input("Would you like to make another
order? (y-n) ")
                        if retry == "n":
                            print("Okay. Goodbye.")
                            quit()
                        else:
                            print("Okay. Going back to the main
menu...")
                            time.sleep(2)
                            os.system('cls')
                            break
                break
            else:
                print("Sorry. The item you were looking for is
unavailable or in insufficient amounts.")
                time.sleep(1)
                os.system('cls')
                break


    elif cat == "q":
        print("Have a nice day.")
```

```
        quit()

    else:


        print("Invalid answer. Please try again.")
        time.sleep(2)

        os.system('cls')
        continue
```