CODELAB I

ASSESSMENT 2: Utility App

Tutor : Ms. Lavanya Mohan

Programming Fundamentals

|  |  |
| --- | --- |
| **Student’s Name:** | John Xander Deada |
| **Student Id** | 2023-327 |
| **Github Repository Name:** | Assessment 2\_CC |
| **Github Repository Link:** | <https://github.com/Solace12/Assessment-2_CC> |
| **YouTube Link** |  |

**BRIEF**

**The task that was given to us was create a vending machine program that uses python codes. This document will show my capabilities and knowledge in programming and python codes. In this documentation I will show and discuss each part of my vending machine code.**



**Specifications**

This code uses simple and basic skills of python programming

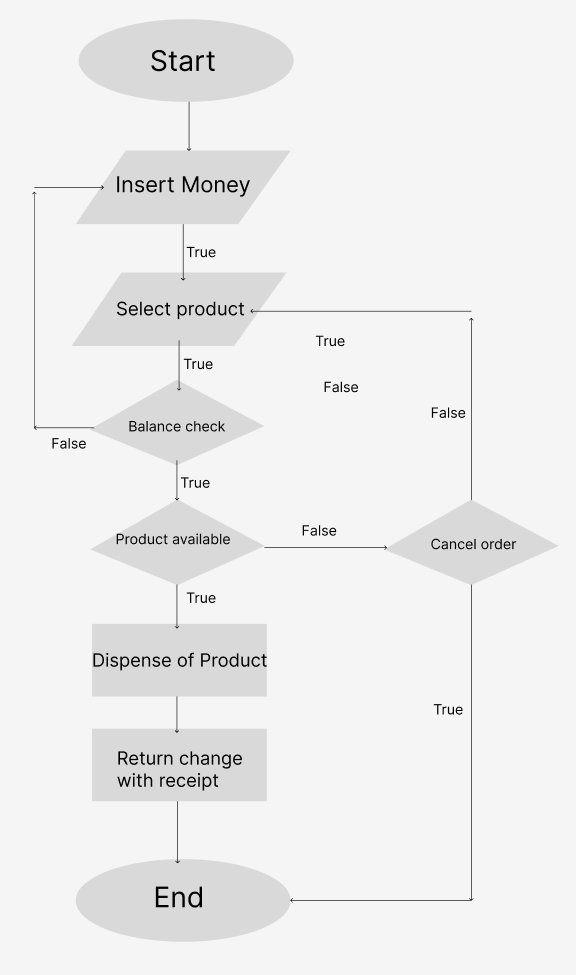
This code has the following:

* The Vending Machine logo
* The Available items in the vending machine
* The Price of each item and how many stocks it has
* Deposit of cash
* Selection of items
* Item dispensing
* Remaining balance
* Ask’s if you want to make another purchase
* Receipt system
* Return of the remaining balance
* A goodbye message

This vending machine requires the basic skills in programming where the user can insert money, pick the item they want to buy and a receipt feature.



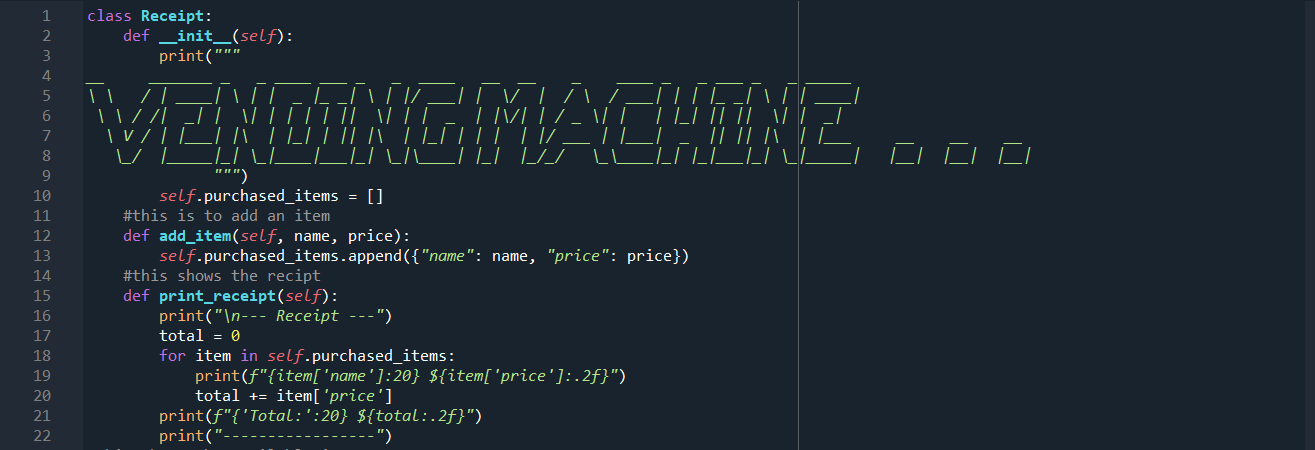
**System Flowchart**

****

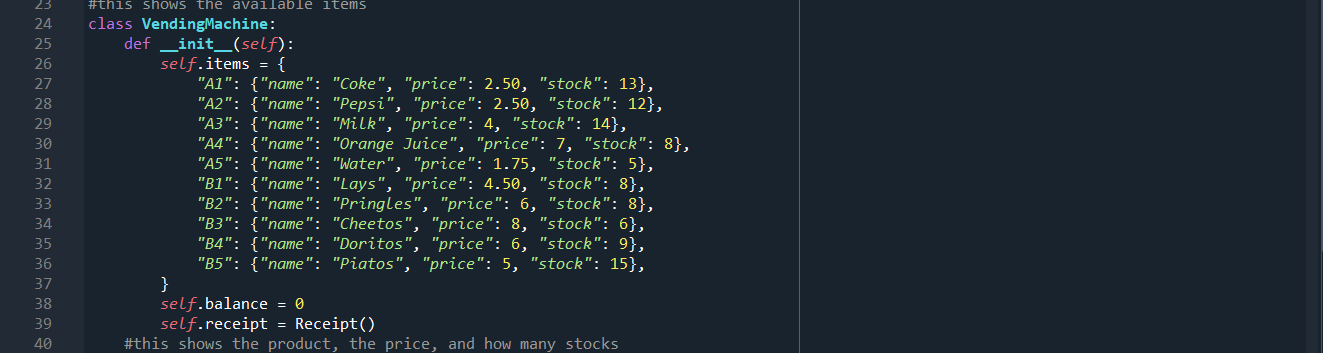
**This flowchart shows the order of the vending machine where the user inserts the money, select the product, checks of the balance is sufficient or insufficient, checks if the product is available or not, dispense of product and return of change with receipt.**

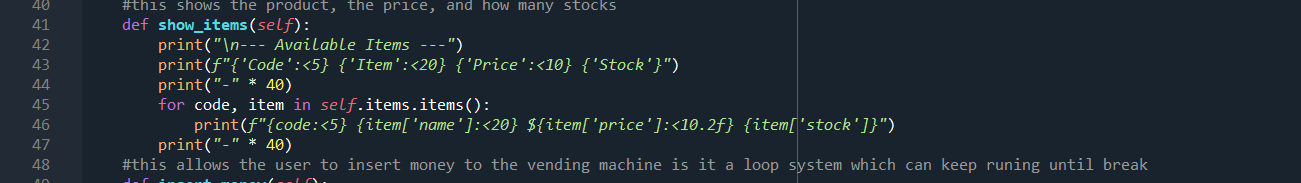


**Technical Description**

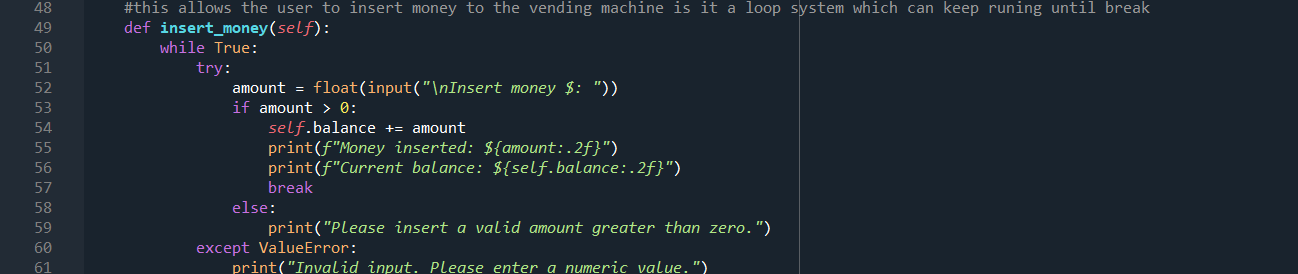
****

**This line of dictionary is a design to add another item to the menu with its name and price also prints the receipt**

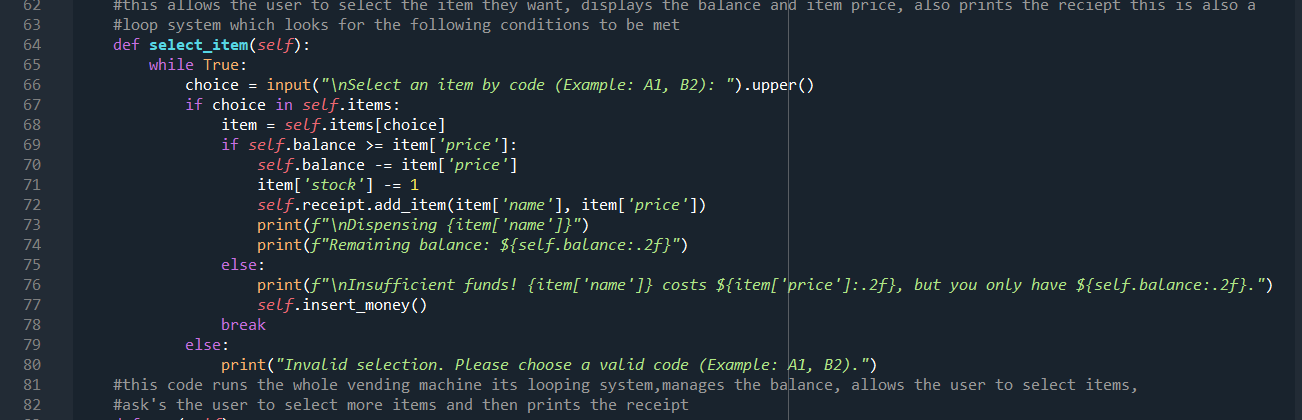
**this function shows the available items in the vending machine**

****

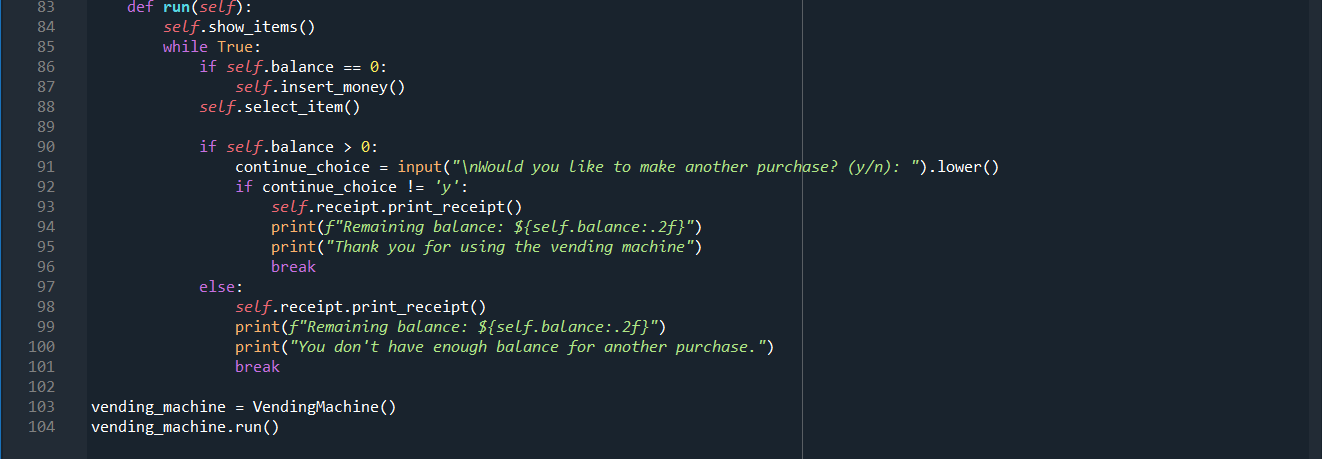
**This function shows the product, price, and how many stocks it has**

****

**This function allows the user to insert money to the vending machine**

****

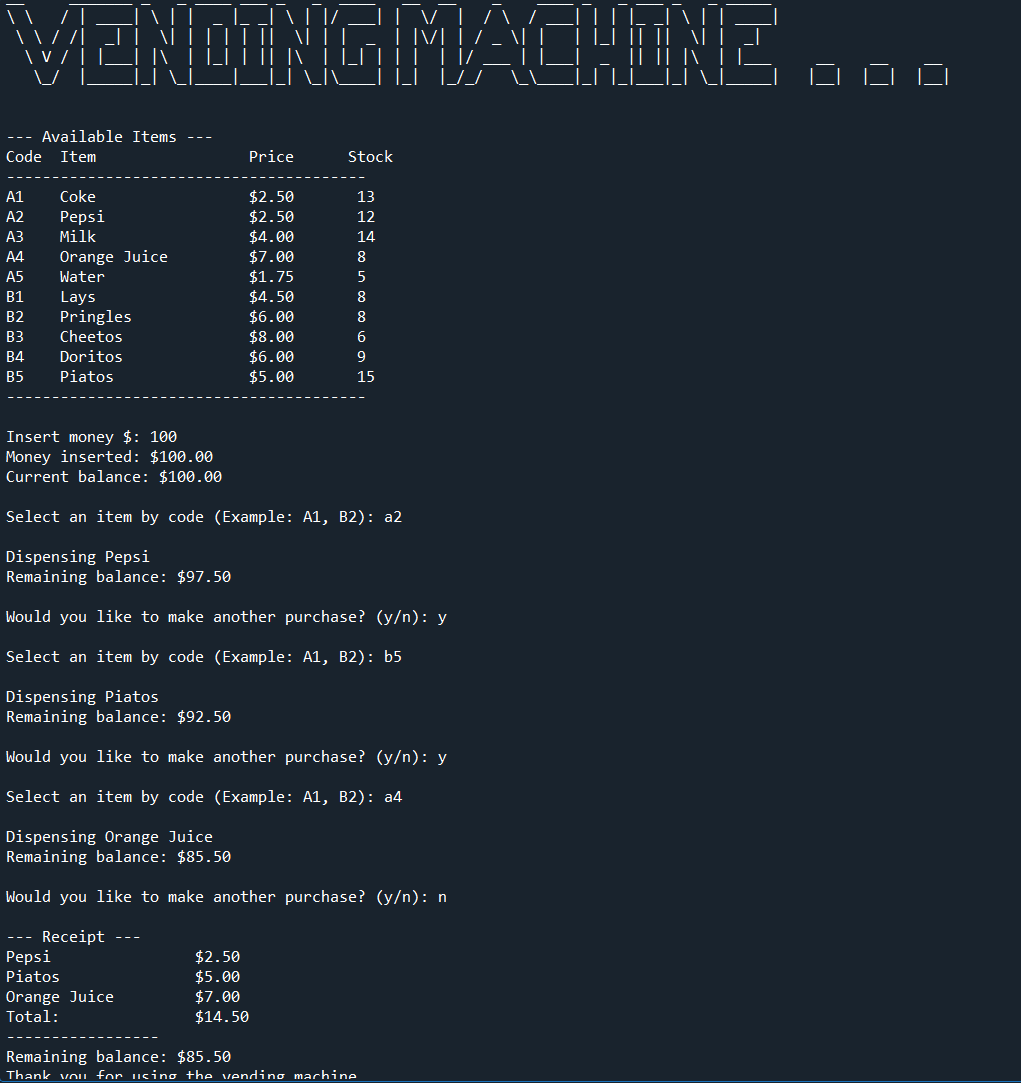
**This function allows the user to select an item from the vending machine which also checks the balance if its sufficient or not.**

****

**This function runs the whole vending machine system how it loops, manages the balance , allows the user to select items, ask’s the user if they want to select more products.5**



**Output**

****



**Critical Reflection**

In making the vending machine it really helped me improve my knowledge of python programming. This improved my critical thinking skills which is really important in python developing. I’ve learned about functions which can make the code easier to understand and more versatile, flexible, easy to manage, and which can reduce memory space. With the use of comment lines, it allows the developer to have a good coding format which can help writing code easier to understand and read, in developing it is a dream of coders and developers to make their code have a good format which can benefit in making the code easy to read and understand. I’ve also learned how to properly code which made me confident in my coding skills and can help me in the future. Overall, there are some functions that I want to add in this vending machine project but due to the lack of knowledge it made it impossible for me to add more functions in this assessment. This assessment made me realize that you don’t only need to know how to write codes but also understand them. It may have took some time and brain power to complete this project but I’ve had fun in programming a vending machine it really tested my way of thinking and improved my skills furthermore I’ve learned more features which can be also added in the next project.



**Appendix**

class Receipt:

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

print("""

\_\_ \_\_\_\_\_\_\_ \_ \_ \_\_\_\_ \_\_\_ \_ \_ \_\_\_\_ \_\_ \_\_ \_ \_\_\_\_ \_ \_ \_\_\_ \_ \_ \_\_\_\_\_

\ \ / | \_\_\_\_| \ | | \_ |\_ \_| \ | |/ \_\_\_| | \/ | / \ / \_\_\_| | | |\_ \_| \ | | \_\_\_\_|

\ \ / /| \_| | \| | | | | || \| | | \_ | |\/| | / \_ \| | | |\_| || || \| | \_|

\ V / | |\_\_\_| |\ | |\_| | || |\ | |\_| | | | | |/ \_\_\_ | |\_\_\_| \_ || || |\ | |\_\_\_ \_\_ \_\_ \_\_

\\_/ |\_\_\_\_\_|\_| \\_|\_\_\_\_|\_\_\_|\_| \\_|\\_\_\_\_| |\_| |\_/\_/ \\_\\_\_\_\_|\_| |\_|\_\_\_|\_| \\_|\_\_\_\_\_| |\_\_| |\_\_| |\_\_|

""")

self.purchased\_items = []

#this is to add an item

def add\_item(self, name, price):

self.purchased\_items.append({"name": name, "price": price})

#this shows the recipt

def print\_receipt(self):

print("\n--- Receipt ---")

total = 0

for item in self.purchased\_items:

print(f"{item['name']:20} ${item['price']:.2f}")

total += item['price']

print(f"{'Total:':20} ${total:.2f}")

print("-----------------")

#this shows the available items

class VendingMachine:

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

self.items = {

"A1": {"name": "Coke", "price": 2.50, "stock": 13},

"A2": {"name": "Pepsi", "price": 2.50, "stock": 12},

"A3": {"name": "Milk", "price": 4, "stock": 14},

"A4": {"name": "Orange Juice", "price": 7, "stock": 8},

"A5": {"name": "Water", "price": 1.75, "stock": 5},

"B1": {"name": "Lays", "price": 4.50, "stock": 8},

"B2": {"name": "Pringles", "price": 6, "stock": 8},

"B3": {"name": "Cheetos", "price": 8, "stock": 6},

"B4": {"name": "Doritos", "price": 6, "stock": 9},

"B5": {"name": "Piatos", "price": 5, "stock": 15},

}

self.balance = 0

self.receipt = Receipt()

#this shows the product, the price, and how many stocks

def show\_items(self):

print("\n--- Available Items ---")

print(f"{'Code':<5} {'Item':<20} {'Price':<10} {'Stock'}")

print("-" \* 40)

for code, item in self.items.items():

print(f"{code:<5} {item['name']:<20} ${item['price']:<10.2f} {item['stock']}")

print("-" \* 40)

#this allows the user to insert money to the vending machine is it a loop system which can keep runing until break

def insert\_money(self):

while True:

try:

amount = float(input("\nInsert money $: "))

if amount > 0:

self.balance += amount

print(f"Money inserted: ${amount:.2f}")

print(f"Current balance: ${self.balance:.2f}")

break

else:

print("Please insert a valid amount greater than zero.")

except ValueError:

print("Invalid input. Please enter a numeric value.")

#this allows the user to select the item they want, displays the balance and item price, also prints the reciept this is also a

#loop system which looks for the following conditions to be met

def select\_item(self):

while True:

choice = input("\nSelect an item by code (Example: A1, B2): ").upper()

if choice in self.items:

item = self.items[choice]

if self.balance >= item['price']:

self.balance -= item['price']

item['stock'] -= 1

self.receipt.add\_item(item['name'], item['price'])

print(f"\nDispensing {item['name']}")

print(f"Remaining balance: ${self.balance:.2f}")

else:

print(f"\nInsufficient funds! {item['name']} costs ${item['price']:.2f}, but you only have ${self.balance:.2f}.")

self.insert\_money()

break

else:

print("Invalid selection. Please choose a valid code (Example: A1, B2).")

#this code runs the whole vending machine its looping system,manages the balance, allows the user to select items,

#ask's the user to select more items and then prints the receipt

def run(self):

self.show\_items()

while True:

if self.balance == 0:

self.insert\_money()

self.select\_item()

if self.balance > 0:

continue\_choice = input("\nWould you like to make another purchase? (y/n): ").lower()

if continue\_choice != 'y':

self.receipt.print\_receipt()

print(f"Remaining balance: ${self.balance:.2f}")

print("Thank you for using the vending machine")

break

else:

self.receipt.print\_receipt()

print(f"Remaining balance: ${self.balance:.2f}")

print("You don't have enough balance for another purchase.")

break

vending\_machine = VendingMachine()

vending\_machine.run()



**Reference List**