Documentation of the vending machine

Brief

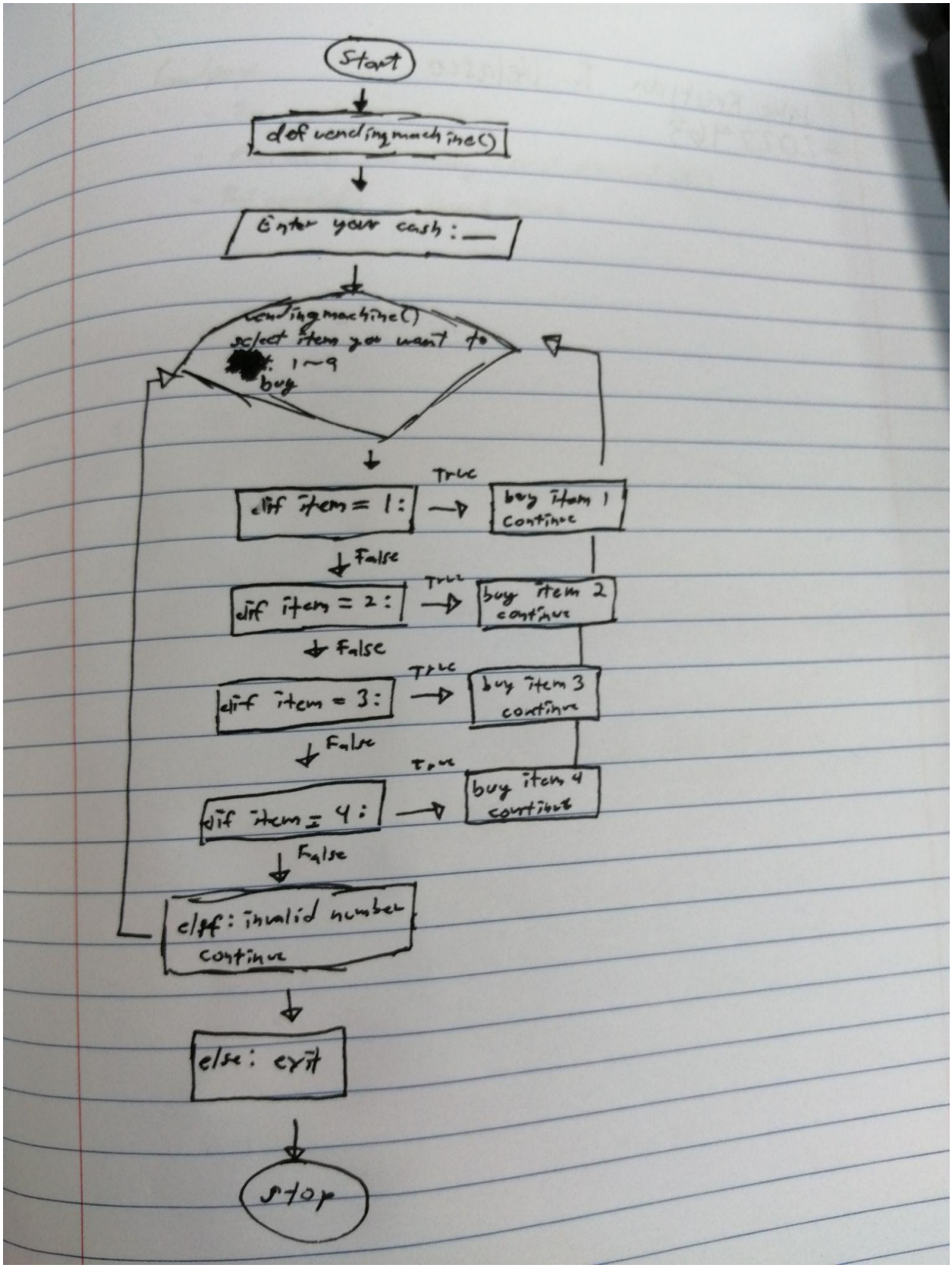
We have been asked to build a program that simulates the experience of buying a vending machine. This program requires us to make use of everything that we have currently learned about python programming. The vending machine works like this: It requires an input from the user for the amount of cash they could have to spend on the vending machine; They could choose what they want to buy; Be given the correct change when buying; And have limited stocks on the vending machine.

This is my iteration of the task.
Github link:

Specification (features):

1. It has a menu of drinks and items the user can buy.
2. It has a set of codes the user can input to choose any item they want.
3. It has a way of capturing the user's input value.
4. It has a way of returning the correct money left.
5. It has a message letting you know you purchased an item.
6. It has comments that explain what the program is doing.
7. It has a feature to buy multiple items.
8. It has a feature of an intelligence system for suggesting purchases.
9. It has the use of functions to improve the structure of your program.
10. And it has a stock system meaning the machine may run out of products

System flowchart


Start
def vendingmachine()
Enter your cash: ___
vendingmachine()
select item you want to
buy
True
elif item = 1:
buy item 1
continue
False
elif item = 2:
True
buy item 2
continue
False
elif item = 3:
True
buy item 3
continue
False
elif item = 4:
True
buy item 4
continue
False
elif: invalid number
continue
else: exit
stop

This is the flowchart of the vending machine. This is the draft of the program, so things are different in the final code, but the logic is the same. It starts with initializing the function, then asks for your cash. After that you will enter a number to buy an item. You can choose to exit but it's not written in the flowchart properly and the code will exit, and the program will stop.

Technical Description

Line 1~25:

```
def vendingmachine(var):
    print("|##############################################|")
    print("|#|                                |##############|")
    print("|#|  =====  ..--'''`  |~~``|    |##|````````|##|")
    print("|#|  |   |  \     |  :    |    |##| Exact  |##|")
    print("|#|  |___|   /___ |  | ___|    |##| Change |##|")
    print("|#|  /=__\  ./.__\   |/,__\    |##| Only   |##|")
    print("|#|  \__//   \__//    \__//    |##|________|##|")
    print("|#|============================|##############|")
    print("|#|````````````````````````````|##############|")
    print("|#| =._      +++     //////   |##############|")
    print("|#| \/  \     | |     \    \   |#|`````````|##|")
    print(f"|#|  \___\    |_|     /___ /   |#|  {user}      |##|")
    print("|#|  / __\\  /|_|\   // __\     |#| |1|2|3| |##|")
    print("|#|  \__//-  \|_//   -\__//    |#| |4|5|6| |##|")
    print("|#|============================|#| |7|8|9| |##|")
    print("|#|````````````````````````````|#| ``````` |##|")
    print("|#| ..--    _______   .--._.    |#|[=======]|##|")
    print("|#| \   \   |    |   |    |    |#|  _   _  |##|")
    print("|#|  \___\  : ___:   | ___|    |#| ||| ( ) |##|")
    print("|#|  / __\  |/ __\   // __\    |#| |||  `  |##|")
    print("|#|  \__//   \__//  /_\__//    |#|  ~      |##|")
    print("|#|============================|#|________|##|")
    print("|#|````````````````````````````|##############|")
    print("|#|||||||||||PUSH|||||||||||||####\|||||/###|")
def vendingmachinebase():
    print("|##############################################|")
```

I started with a function because this function will be reused in the program and I wanted to have a vending machine art while the vending machine program is running. The art was taken from a website, it is not made by me. This is the link to the website. All credits go to the original creators.

In line 13, the code isn't aligned with the rest of the function. This is because there's a bug that makes the output misaligned with the rest unless it is formatted the way in the screenshot. Another limitation of the art is that it cannot go beyond 2 digits as the variable. The vending machine requires an input of 1-9 and if you write anything else, it will exit or make you retry another number. Entering 10+ will make the vending machine art be once again misaligned with the rest of the art and I cannot fix it to make it so that it can support both inputs. So, I decided to remove the 10th option and made the 10th item an easter egg in the program.

Line 26~50

```
26 def vendingmachinebase():
27     print("|##############################################|")
28     print("|#|                                          |##############|")
29     print("|#|  =====  ..--'''`  |~~``|   |##|````````|##|")
30     print("|#|  |   |  \     |  :    |   |##| Exact  |##|")
31     print("|#|  |___|   /___ |  | ___|   |##| Change |##|")
32     print("|#|  /=__\  ./.__\   |/,__\   |##| Only   |##|")
33     print("|#|  \__//   \__//    \__//   |##|________|##|")
34     print("|#|==========================|##############|")
35     print("|#|``````````````````````````|##############|")
36     print("|#| =._      +++     //////  |##############|")
37     print("|#| \/  \     | |     \    \  |#|`````````|##|")
38     print("|#|  \___\    |_|     /___ /  |#|         |##|")
39     print("|#|  / __\\  /|_|\   // __\    |#| |1|2|3| |##|")
40     print("|#|  \__//-  \|_//   -\__//   |#| |4|5|6| |##|")
41     print("|#|==========================|#| |7|8|9| |##|")
42     print("|#|``````````````````````````|#| ``````` |##|")
43     print("|#| ..--    _______   .--._.   |#|[=======]|##|")
44     print("|#| \   \   |    |   |    |   |#|  _   _  |##|")
45     print("|#|  \___\  : ___:   | ___|   |#| ||| ( ) |##|")
46     print("|#|  / __\  |/ __\   // __\   |#| |||  `  |##|")
47     print("|#|  \__//   \__//  /_\__//   |#|  ~      |##|")
48     print("|#|==========================|#|_________|##|")
49     print("|#|``````````````````````````|##############|")
50     print("|#|||||||||||PUSH|||||||||||||####\|||||/###|")
```

It looks very similar to the first function but what is the difference? The difference between the two is that this function doesn't require a variable to and there is no variable in the function as well. This is just made as a display in the program while the first function will be the one used throughout.

Line 51~53:

```
51  #user will input any amount of cash they want
52  cash= int(input("You walk to a vending to buy something. You checked your
        pockets and you have how much money?\n"))
53  print("You have",cash,"AED in your pockets.")
```

This variable required an INT data type and an input from the user. This variable will then store the value for the user's cash in the vending machine. It will then print out an output that you have X amount of cash in your pockets.

In one of the requirements it states that: The user should be able to input any amount of money and have the correct change returned.

This variable is for that and for flexibility. If the user wants to start with more, or less money, they can. Later in the program it will also show the correct remaining change.

Line 54~55:

```
53  print("You have",cash,"AED in your pockets.")
54  #To keep track of items bought
55  inventory=[]
```

The inventory list is to keep track of everything the user has bought and will be shown what's in the variable when exiting the vending machine.

Line 56~58:

```
56  stock1,stock2,stock3,stock4,stock5,stock6,stock7,stock8,stock9,stock10=10,10
        ,10,10,10,10,10,10,10,10
57  cashleft=cash
58  cost1,cost2,cost3,cost4,cost5,cost6,cost7,cost8,cost9,cost10=2,2,3,3,4,4,5,5,6
        ,6
```

Line 56 is initiating the stock's stock. They will all start out at 10 stocks. You can notice that there are 10 stocks instead of 9. This is because stock10 is an easter egg that can only be found if you buy the 9th item. Line 57 is initializing another variable, and this variable's value takes after the value of 'cash'. This is so that there's 2 variables, 1 is the variable that tracks what the user has entered. And the other variable tracks your cash left after buying in the vending machine. I made 2 variables because of an error.

This is how it looks like updated:

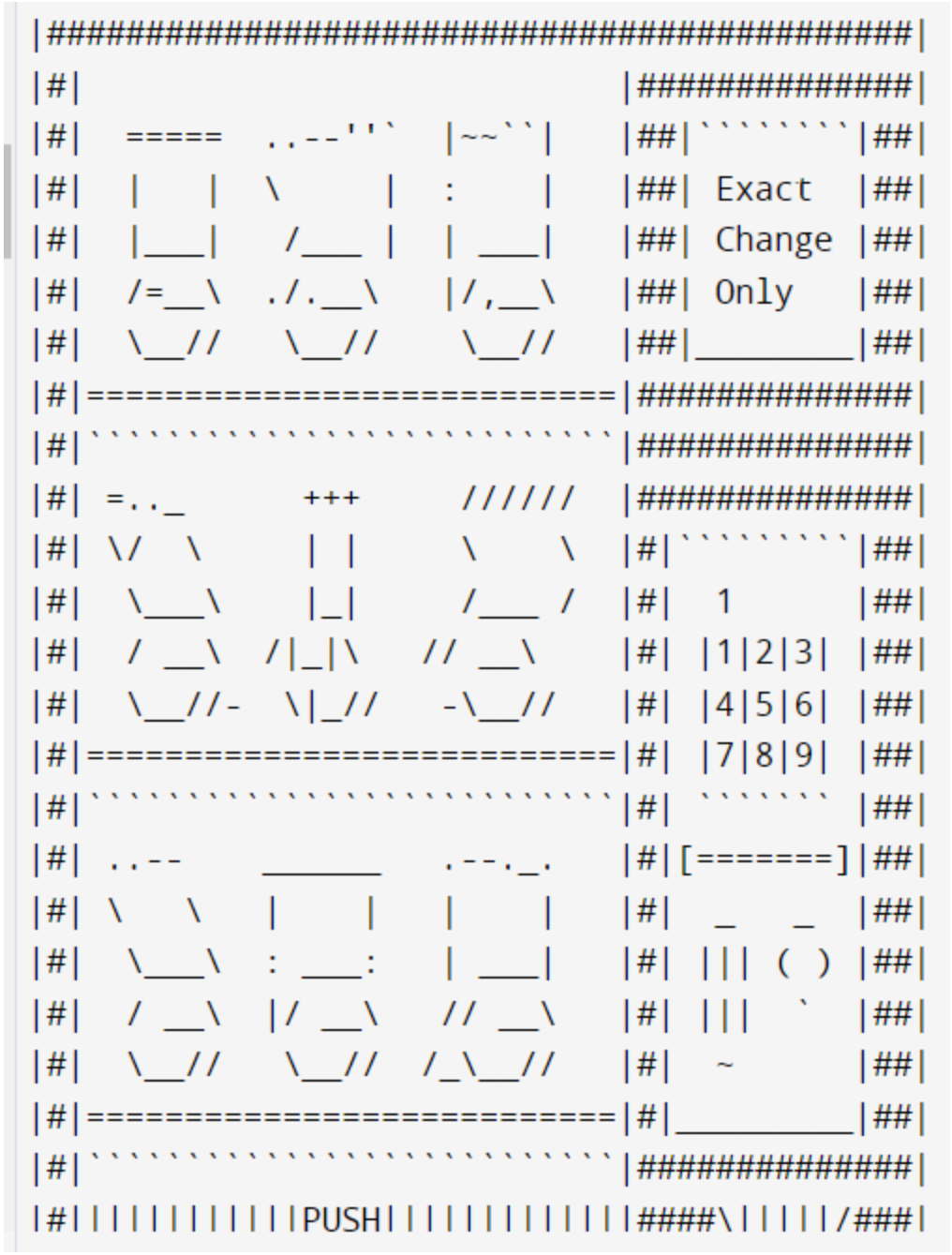
```
55  inventory=[]
56  stock1,stock2,stock3,stock4,stock5,stock6,stock7,stock8,stock9,stock10=10,10
        ,10,10,10,10,10,10,10,10
57  cost1,cost2,cost3,cost4,cost5,cost6,cost7,cost8,cost9,cost10=2,2,3,3,4,4,5,5,6
        ,6
58  vendingmachinebase()
```

I have removed the ‘cashleft’ variable. Line 57 now is where the costs of the products are listed. Line 58 is when the 2nd vending machine function is called. This is to just display the vending machine art.

Line 59~61:

```
59  while cash>=cost1 or cash>=cost3 or cash>=cost5 or cash>=cost7 or cash>=cost9
60      user=int(input("You look and see the menu: 1-Bread(2AED): 2-Water(2AED): 3
            -Chocolate(3AED): 4-Hot Drink(3AED): 5-Chips(4AED): 6-Tea(4AED): 7
            -Sandwich(5AED): 8-Soda(5AED): 9-Energy Drinks(6AED). The vending
            machine requires a number:\n"))
61      vendingmachine(user)
```

Line 59 checks if the cash value the user entered can buy the lowest priced item and if it is true, it will then ask for an input to be stored in the ‘user’ variable. In the print message, it displays the menu of the items that can be bought along with the cost of these respective items. Then, a function called ‘vendingmachine’ is called with the variable ‘user’. This will display:



As you can see from the picture, there is ‘1’ on the vending machine where there would normally be none in the original ASCII art. This is because I made it display the number the user has input when buying an item from the vending machine. This makes the program more immersible.

Line 62~78:

```
62      #check what number has been entered
63      if user==1:
64          #reduces stock by 1
65          stock1=stock1-1
66          if stock1>=0:
67              #reduces cash with the cost
68              cash=cash-cost1
69              print("You bought bread for",cost1,"AED. You have",cash,"left.
                    There's",stock1," bread left. This pairs well with water for"
                    ,cost2,"AED.")
70              #adds item to inventory
71              inventory.append("bread")
72              #vending machine suggesting another item.
73              user2=input("Would you like to buy water? (y/n)")
74              if user2=='y':
75                  cash=cash-cost2
76                  stock2=stock2-1
77                  print(f"You bought water for {cost2} AED. You have {cash} left
                        . The amount of stock remaining is: {stock2}.")
78                  inventory.append("water")
```

Line 63 checks what number has the user entered in the menu of the vending machine. After, the stock of that item reduces by 1. If the stock is 0 or greater than 0, it will purchase the stock by reducing the amount of cash and by adding it to inventory. After adding the item to the inventory, the vending machine suggests an item to pair with the item bought. It will ask if you want to buy the suggested item to be stored in the variable ‘user2’.

The code has been updated to where it will check if there is a stock left of the suggested item. It now looks like this:

```
62      if user==1:
63          if stock1>=1:
64              stock1=stock1-1
65              cash=cash-cost1
66              print("You bought bread for",cost1,"AED. You have",cash,"left.
                    There's",stock1," bread left.")
67              inventory.append("bread")
68              if stock2>=1:
69                  print("This pairs well with water for",cost2,"AED.")
70                  user2=input("Would you like to buy water? (y/n)")
71                  if user2=='y':
72                      cash=cash-cost2
73                      stock2=stock2-1
74                      print(f"You bought water for {cost2} AED. You have {cash}
                            left. The amount of stock remaining is: {stock2}.")
75                      inventory.append("water")
76                      user3=input("Would you like to leave? (yes/no)")
77                      if user3=='yes':
78                          break
79                      elif user3=='no':
80                          continue
81                  elif user2=='n':
82                      continue
83              else:
84                  continue
85      else:
```

And the rest of the code looks similar after that. With the biggest difference being the line 62 being changed from: if user==1: to: elif user==9.

Line 290~294

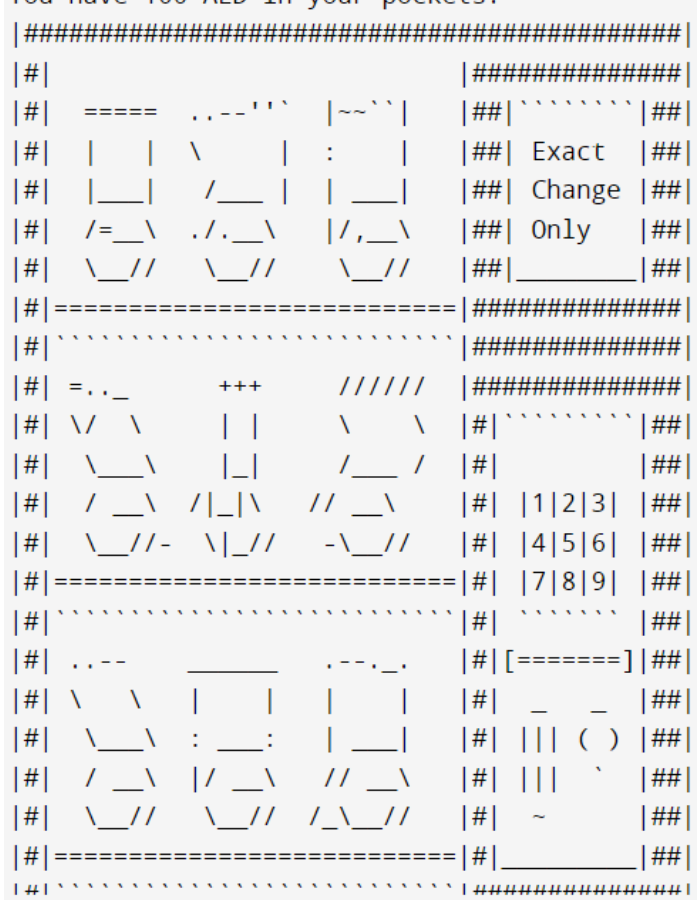
```
290     else:
291         print("Invalid number.")
292         continue
293 print("You left with",cash,"AED in your pockets. And the things you bought are
        :",inventory)
294
```

This is the last few lines of the code. Line 290 checks if the number entered is between 1~9. If it isn't it will print that message and it continues in the loop. Line 293 prints how much money you have left and the things you have bought while buying in the vending machine. This message shows when you leave the vending machine.

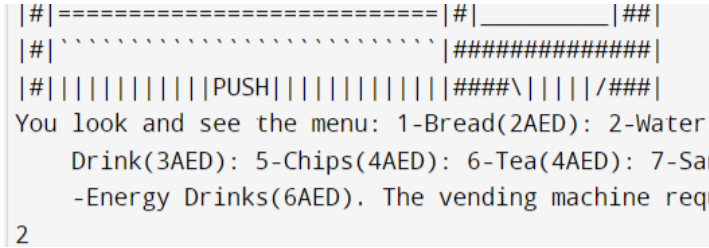
Mechanics

This section will be used to explain the game mechanics of the vending machine project. First, it narrates a story of the user walking up to a vending machine and checking how much money you have. You are then prompted to enter any amount of money and will be stored in the ‘cash’ variable.

```
You walk to a vending to buy something. You checked your pockets and you have how
    much money?
100
You have 100 AED in your pockets.
```



After that, you look at the numbers and see their prices and what item they are selling, you are then prompted to buy an item by entering a number.



```
You look and see the menu: 1-Bread(2AED): 2-Water(2AED): 3-Chocolate(3AED): 4-Hot
    Drink(3AED): 5-Chips(4AED): 6-Tea(4AED): 7-Sandwich(5AED): 8-Soda(5AED): 9
    -Energy Drinks(6AED). The vending machine requires a number:
2
```

The item is then transferred to your inventory, which you can check after leaving the vending machine. Then, the vending machine suggests an item to go along with the item you just bought.

```
|#|  \__//   \__//  /_\__//   |#|  ~      |##|
|#|============================|#|_________|##|
|#|`````````````````````````````|###############|
|#|||||||||||PUSH|||||||||||||####\|||||/###|
You bought water for 2 AED. You have 98 left. There's 9 water left.
This pairs well with bread for 2 AED.
```

Answering yes will let you buy the item, while answering no will leave the vending machine, showing how much money you have left, and the list of items you have bought.

For the purposes of this section, I will show you what happened if you wanted to buy the suggested item. It will look like this:

```
You bought water for 2 AED. You have 98 left. There's 9 water left.
This pairs well with bread for 2 AED.
Would you like to buy bread? (y/n)y
You bought bread for 2 AED. You have 96 left. The amount of stock remaining is: 9.
Would you like to leave? (yes/no)yes
You left with 96 AED in your pockets. And the things you bought are: ['water',
    'bread']
>
```

After buying the suggested item, you are then asked to leave. If you choose to stay, the menu will pop up again and you can purchase different items or the same ones. If you choose to leave, it will show how much cash you have left and the list of the things you bought, and the program terminates from there.

Critical Reflection

I think coding this program was definitely tedious. I started creating this program since January 4th, 2023. It is now January 7th, 2023. It took me less than a week to create this program, and it is super tiresome to continue creating this program. This is because it has 290+ lines of code and I didn't bother turning the core mechanics into a function. Having them turned into a function would definitely make this program have less lines and easier. I didn't do it because I'm running out of time and this paper is due on January 8th, 2023. The program as it is is passable. You definitely feel like you're buying off of a vending machine because of the ASCII art. I envisioned what the program would look like in my head first, then tried copying how I saw it in code. I think overall it turned out how I thought about it. Any small error might destroy the code but I fixed it multiple times now.