Vending Machine Logic Programming: Return Smallest Amount of Notes

Author: Wong Yew Lee

Email: wongyewlee-wm19@student.tarc.edu.my

Github: https://github.com/wyewlee

Assumption:

1. Assuming that ***the vending machine sells only one type of can drink.***
2. Assuming that ***the can drink cost only RM1 each.***
3. Assuming that ***the vending machine can only process one can per transaction.***
4. Assuming that ***the vending machine accept one money note only.***
5. Assuming that ***the vending machine return the balances in notes only.***
6. Assuming that ***the denomination of notes is limited to RM100, RM50, RM20, RM10, RM5, RM1***

Python Version

In [1]:

```
from platform import python_version

print(python_version())
```

```
3.8.10
```

Python Code

In [9]:

```
def insertNotes(): #ask for inserted note from user input
    print('Note Inserted? Format: xx.xx')
    x = input()
    print('\nInserted Note: RM',x)
    return x

def checkInsertedNotes(x): #checking on inserted note
    available_notes = [100,50,20,10,5,1] #available denomination of notes

    x = x.split('.') #preprocess and string

    if len(x) != 2:
        raise Exception('Please use the format xx.xx')

    if int(x[0]) not in available_notes: #if inserted note is not in the avaialble denominations
        raise Exception('Not a existing notes.')

    if x[1] == '00': #check if there is cents being input by user
        return True
    else: #assuming vending machine accept only notes
        raise Exception('Inserted note should not have cents.')

def processNotes(x):  #necessary processing on inserted amount
    x = x.split('.') #preprocess data
    x = int(x[0]) - 1 #assuming each can is RM1, deduct RM1 from the inserted amount

    available_notes = [100,50,20,10,5,1]
    result = dict({100:0, 50:0, 20:0, 10:0, 5:0, 1:0}) #empty dictionary

    for n in range(0,len(available_notes)):
        if x >= available_notes[n]:
            count = x // available_notes[n]
            x = x - (count*available_notes[n])

            payload = {available_notes[n]:count}
            result.update(payload) #update count to dictionary
    return result #return data in dictionary format


def returnNotes(x): #toString Data of dictionary format
    print("\n*Total Returning Notes:*")
    total = 0
    for key,value in x.items():
        print('RM',key,'\t: ', value)
        total = total + (key*value)
    print('========\nReturned Total\t: RM',total,'.00')

def main():
    notes = insertNotes()
    checkInsertedNotes(notes)
    returnNotes(processNotes(notes))
```

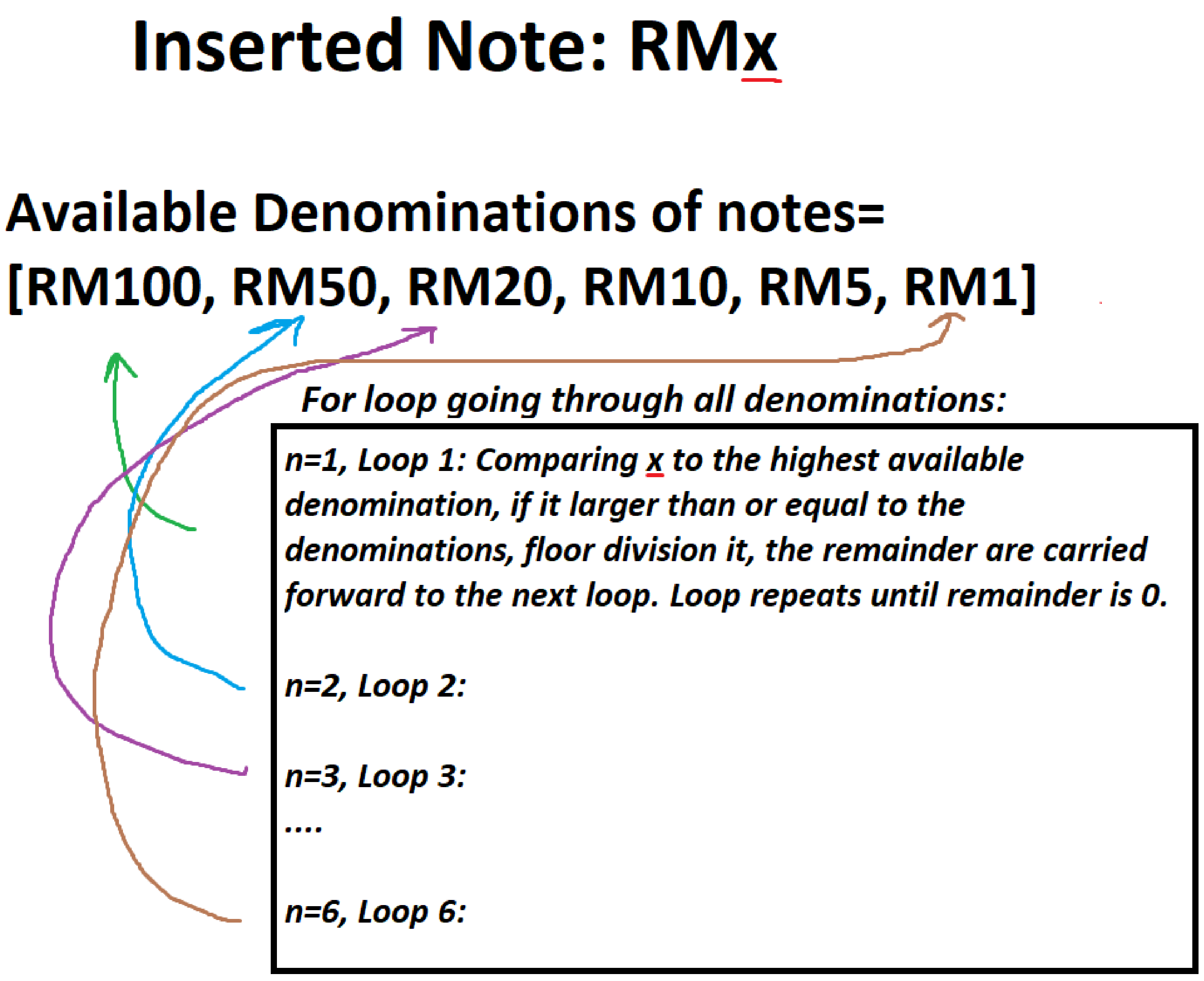
```python
if __name__ == "__main__":
    main()
```

```
Note Inserted? Format: xx.xx
20.00

Inserted Note: RM 20.00

*Total Returning Notes:*
RM 100  :  0
RM 50   :  0
RM 20   :  0
RM 10   :  1
RM 5    :  1
RM 1    :  4
========
Returned Total  : RM 19 .00
```

Working Logic in Diagram



Inserted Note: RMx

Available Denominations of notes= [RM100, RM50, RM20, RM10, RM5, RM1]

For loop going through all denominations:

n=1, Loop 1: Comparing x to the highest available denomination, if it larger than or equal to the denominations, floor division it, the remainder are carried forward to the next loop. Loop repeats until remainder is 0.

n=2, Loop 2:

n=3, Loop 3:

....

n=6, Loop 6: