

Decision coverage of withdraw

Made by Chinese-broccoli

Author: Mason Chang, Yueyang Liu, Ruikang Luo.

```
def processWithdraw(self, toAccount, amount, fromAccount):
    match, idx = self.binarySearch(fromAccount)
    ① if match:
        decreaseAccount = self.masterList[idx].split(' ')
        name = decreaseAccount[2].strip()
        oldAmount = decreaseAccount[1]
        newAmount = int(oldAmount) - int(amount)
    ② if newAmount < 0 :
        print(f"Transaction: WDR {toAccount} {amount} {fromAccount} *** I
            s invalid!")
        print("Due to: insufficient balance.")
        return False
    ③ else:
        updateInfo = f"{fromAccount} {newAmount} {name}\n"
    ④ if self.checkUpdateInfo(updateInfo):
        self.masterList[idx] = updateInfo
    ⑤ else:
        print(f"Transaction: WDR {toAccount} {amount} {fromAccount} *
            ** is invalid!")
        print("Due to: Invalid length.")
        return False
    ⑥ else:
        print(f"Transaction: WDR {toAccount} {amount} {fromAccount} *** is in
            valid!")
        print("Due to: no matching account found!")
        return False
    return True
```

```
def binarySearch(self, targetAccount):
    targetAccount = int(targetAccount)
    start = 0
    end = self.length - 1
    # prepare for the return when not find the account number
    middle = (start + end) // 2
    ⑦ while start <= end:
        middle = (start + end) // 2
```

```

        account = self.masterList[middle][:7]
    ⑧    if targetAccount == int(account):
        return True, middle
    ⑨    elif targetAccount > int(account):
        end = middle - 1
    ⑩    else:
        start = middle + 1
    return False, middle

```

Test No.	Input transaction	Decision	Terminal output	Description
T1.0	WDR 0000000 100000 1000327 ***	① True ⑥ False ② False ③ True ④ True ⑤ False ⑦ True ⑧ True ⑨ False ⑩ True	None	Successfully withdraw, account is in the bottom of the master file
T1.1	WDR 0000000 100000 1000329 ***	① True ⑥ False ② False ③ True ④ True ⑤ False ⑦ True ⑧ True ⑨ False ⑩ True	None	Successfully withdraw, account is in the top of the master file
T1.2	WDR 0000000 10000 1000330 ***	① False ⑥ True ⑦ False ⑧ False	Transaction: WDR 0000000 10000 1000330 *** is invalid! Due to: no matching account found!	Unsuccessfully withdraw, due to no matching account found
Notes: up to now, both side of direction (⑨ , ⑩), whether within given range (⑦), and matching or not (⑧) have tested. For simplicity, they will not be stated in the below table.				
T1.3	WDR 0000000 100000 1000328 ***	① True ⑥ False ② True ③ False	Transaction: WDR 0000000 100000 1000328 *** is invalid! Due to: insufficient balance!	Unsuccessfully withdraw, due to insufficient balances

T1.4	WDR 0000000 10000 1000328 ***	<p>① True ⑥ False</p> <p>② False ③ True</p> <p>④ False ⑤ True</p>	Transaction: WDR 0000000 10000 1000328 *** is invalid! Due to: Invalid length!	Unsuccessfully withdraw, due to the account information is longer than 46 characters.
<p>Sum up: both sides of ① is tested at T1.0 and T1.2; ② is tested at T1.0 and T1.3;</p> <p>③ is tested at T1.0 and T1.3; ④ is tested at T1.0 and T1.4;</p> <p>⑤ is tested at T1.0 and T1.4; ⑥ is tested at T1.0 and T1.2;</p>				

Failure spotted at T1.0 and T1.2

T1.0, found that the account list should be stored in descending order

```

75 # @ return (found or not, closest index)
76 def binarySearch(self,targetAccount):
77     targetAccount = int(targetAccount)
78     start = 0
79     end = self.length - 1
80     # prepare for the return when not find the account number
81     middle = (start + end) // 2
82     while start <= end:
83         middle = (start + end) // 2
84         account = self.masterList[middle][:7]
85         if targetAccount == int(account):
86             return True, middle
87         elif targetAccount < int(account):
88             end = middle - 1
89         else:
90             start = middle + 1
91     return False, middle
92
93 # @newClient string of new client information
94 # @startIdx the closest index start to
95 def insertToList(self, newClient, startIdx):
96     newAccount = newClient[:7]
97     account = self.masterList[startIdx][:7]
98     if newAccount < account:
99         self.masterList.insert(startIdx, newClient)
100     else:
101         self.masterList.insert(startIdx+1, newClient)
102
75 # find target account and then return it's index (if not)
76 def binarySearch(self,targetAccount):
77     targetAccount = int(targetAccount)
78     start = 0
79     end = self.length - 1
80     # prepare for the return when not find the account number
81     middle = (start + end) // 2
82     while start <= end:
83         middle = (start + end) // 2
84         account = self.masterList[middle][:7]
85         if targetAccount == int(account):
86             return True, middle
87         elif targetAccount > int(account):
88             end = middle - 1
89         else:
90             start = middle + 1
91     return False, middle
92
93 # insert the new line to the master list with specific index
94 def insertToList(self, newClient, startIdx):
95     newAccount = newClient[:7]
96     account = self.masterList[startIdx][:7]
97     if newAccount > account:
98         self.masterList.insert(startIdx+1, newClient)
99     else:
100         self.masterList.insert(startIdx, newClient)
101

```

T1.2, found that sys.exit() can't be used in pytest. Changing sys.exit() to return False (For simplicity, will not post more similar example.)

<pre> 108 # The process for deposit transcription 109 def processDeposit(self, toAccount, amount, fromAccount): 110 match, idx = self.binarySearch(toAccount) 111 if match: 112 increaseAccount = self.masterList[idx].split(' ') 113 name = increaseAccount[2].strip() 114 oldAmount = increaseAccount[1] 115 newAmount = int(oldAmount) + int(amount) 116 updateInfo = f"{toAccount} {newAmount} {name}" 117 if self.checkUpdateInfo(updateInfo): 118 self.masterList[idx] = updateInfo 119 else: 120 print(f"Transaction: DEP {toAccount} {amount} {fromAccount}") 121 print("Due to: invalid length!\n") 122 sys.exit() 123 else: 124 print(f"Transaction: DEP {toAccount} {amount} {fromAccount}") 125 print("Due to: no matching account found!\n") 126 sys.exit() 127 </pre>	<p><i>change to →</i></p>	<pre> 108 # The process for deposit transcription 109 def processDeposit(self, toAccount, amount, fromAccount): 110 match, idx = self.binarySearch(toAccount) 111 if match: 112 increaseAccount = self.masterList[idx].split(' ') 113 name = increaseAccount[2].strip() 114 oldAmount = increaseAccount[1] 115 newAmount = int(oldAmount) + int(amount) 116 updateInfo = f"{toAccount} {newAmount} {name}" 117 if self.checkUpdateInfo(updateInfo): 118 self.masterList[idx] = updateInfo 119 else: 120 print(f"Transaction: DEP {toAccount} {amount} {fromAccount}") 121 print("Due to: invalid length!") 122 return False 123 else: 124 print(f"Transaction: DEP {toAccount} {amount} {fromAccount}") 125 print("Due to: no matching account found!") 126 return False 127 return True </pre>
---	---------------------------	---