原有的 p，只檢測了交易失敗的情形，並沒有檢測成功交易的情形，如圖:

```
// Property Logic
/***** whether the change is right *****/
assign p = initialized && (serviceTypeOut == `SERVICE_OFF) && (itemTypeOut == `ITEM_NONE) && (outExchange != inputValue); // catch the bug
```

```
2    3 2 0 2 0
```

```
==========================================
= cycle 7
==========================================
serviceTypeOut= 0
itemTypeOut= 3
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

這是一個成功交易的 pattern，可以看到在此時 itemTypeOut!=‘ ITEM_NONE，所以 p 自然是 0，其他項目即使有 bug 也抓不出來，所以這邊修正此處 p 的寫法，如圖:

```
// Property Logic
/***** whether the change is right *****/
assign p = initialized && (serviceTypeOut == `SERVICE_OFF)
        && ((itemTypeOut == `ITEM_NONE && outExchange != inputValue) || (itemTypeOut == `ITEM_A && inputValue != outExchange + `COST_A) ||
            ((itemTypeOut == `ITEM_B && inputValue != outExchange + `COST_B) || (itemTypeOut == `ITEM_C && inputValue != outExchange + `COST_C) ); // catch the bug
```

對於最後一項，我把四種情況都考慮進去，如果 itemTypeOut 為其一，那另外三項會是 0，&&後還是 0，因此我們只要檢查該項是否符合條件(||中，要全部為 0 才是 0)，條件寫法跟之前也類似，就只是針對成功交易的情形，作金額的驗證。驗證過程如下:

測試 1:交易成功(ITEM_A、ITEM_B、ITEM_C)且不找零

ITEM_A pattern

```
1    1 0 0 0 0
2    1 3 1 0 0
```

ITEM_A result

```
==========================================
= cycle 6
==========================================
serviceTypeOut= 2
itemTypeOut= 1
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 7
==========================================
serviceTypeOut= 0
itemTypeOut= 1
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 8
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

ITEM_B pattern

```
1   1 0 0 0 0
2   2 0 1 1 0
```

ITEM_B result

```
==========================================
= cycle 6
==========================================
serviceTypeOut= 2
itemTypeOut= 2
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 7
==========================================
serviceTypeOut= 0
itemTypeOut= 2
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 8
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

ITEM_C pattern

```
1   1 0 0 0 0
2   3 2 0 2 0
```

ITEM_C result

```
==========================================
= cycle 6
==========================================
serviceTypeOut= 2
itemTypeOut= 3
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 7
==========================================
serviceTypeOut= 0
itemTypeOut= 3
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 8
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

測試 1 都有通過

測試 2:交易成功 (ITEM_A、ITEM_B、ITEM_C)但有找零

ITEM_A pattern(投 10 元要找 2 個 1 塊)

```
1   1 0 0 0 0
2   1 0 0 1 0
```

ITEM_A result

```
==========================================
= cycle 8
==========================================
serviceTypeOut= 2
itemTypeOut= 1
coinOutNTD_1= 2
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 9
==========================================
serviceTypeOut= 0
itemTypeOut= 1
coinOutNTD_1= 2
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 10
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

ITEM_B pattern(投 20 元要找 1 個 5 塊)

```
1   1 0 0 0 0
2   2 0 0 2 0
```

ITEM_B result

```
==========================================
= cycle 7
==========================================
serviceTypeOut= 2
itemTypeOut= 2
coinOutNTD_1= 0
coinOutNTD_5= 1
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 8
==========================================
serviceTypeOut= 0
itemTypeOut= 2
coinOutNTD_1= 0
coinOutNTD_5= 1
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 9
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

ITEM_C pattern(投 27 元要找 1 個 5 塊)

```
1   1 0 0 0 0
2   3 2 1 2 0
```

ITEM_C result

```
==========================================
= cycle 7
==========================================
serviceTypeOut= 2
itemTypeOut= 3
coinOutNTD_1= 0
coinOutNTD_5= 1
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 8
==========================================
serviceTypeOut= 0
itemTypeOut= 3
coinOutNTD_1= 0
coinOutNTD_5= 1
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
= cycle 9
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

測試 2 都有通過

測試 3:交易失敗

零錢投不夠的 case(要買 ITEM_C，投了 20 塊)

```
1   1 0 0 0 0
2   3 0 0 2 0
```

result

```
==========================================
= cycle 8
==========================================
serviceTypeOut= 2
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 2
coinOutNTD_50= 0
p= 0
==========================================
= cycle 9
==========================================
serviceTypeOut= 0
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 2
coinOutNTD_50= 0
p= 0
==========================================
= cycle 10
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

販賣機找不出錢的 case(要買 ITEM_C，投了 30 塊，1 塊錢卻不夠找)

```
1   1 0 0 0 0
2   3 0 0 3 0
```

result

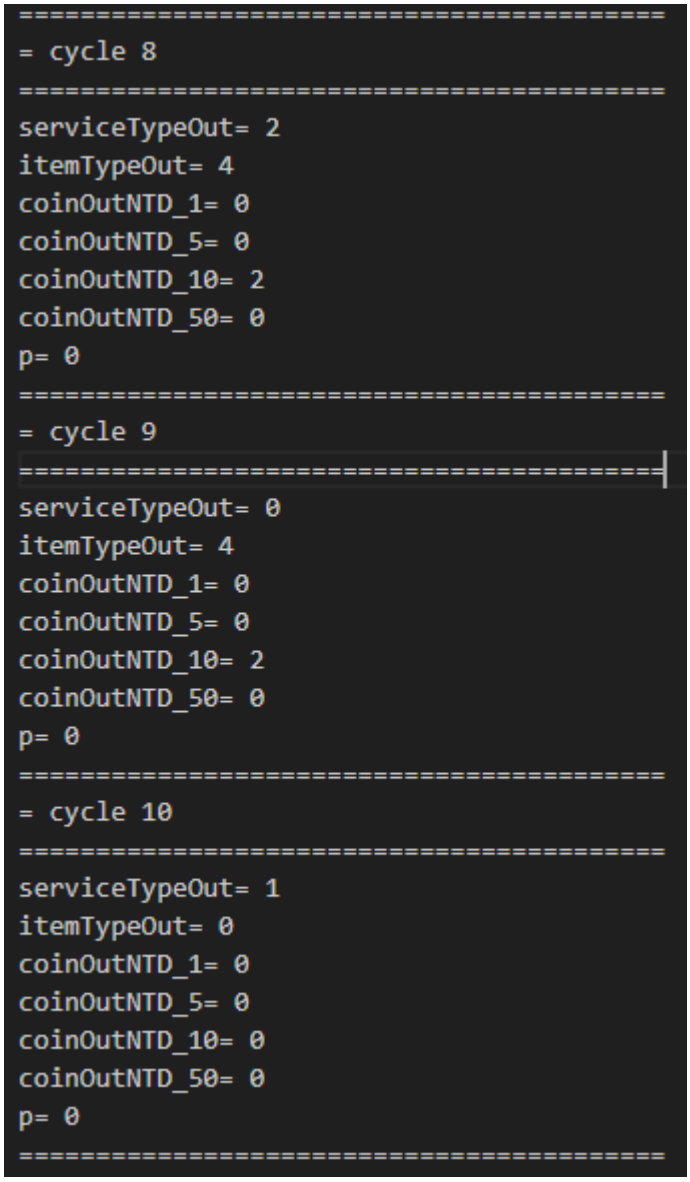
```
==========================================
= cycle 16
==========================================
serviceTypeOut= 2
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 3
coinOutNTD_50= 0
p= 0
==========================================
= cycle 17
==========================================
serviceTypeOut= 0
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 3
coinOutNTD_50= 0
p= 0
==========================================
= cycle 18
==========================================
serviceTypeOut= 1
itemTypeOut= 0
coinOutNTD_1= 0
coinOutNTD_5= 0
coinOutNTD_10= 0
coinOutNTD_50= 0
p= 0
==========================================
```

測試 3 都有通過

測試 4:輸入為 ITEM_NONE

```
1   1 0 0 0 0
2   4 0 0 2 0
```

result



測試結果也正常

目前的測試結果，都符合運作情形，只剩下連續 pattern 的 case 沒有測試，不過由於這樣的種類太多了，也不方便檢查所有可能性，再加上這次的設計是一個 cycle 吃一行 pattern，我還要花心力去檢查每一筆測資共需多少 cycle，以多個 pattern 來說不是很好做，所以這邊也不做測試了，至少我驗證了每一個運作指令的正確性。