基礎程式設計與應用_G3

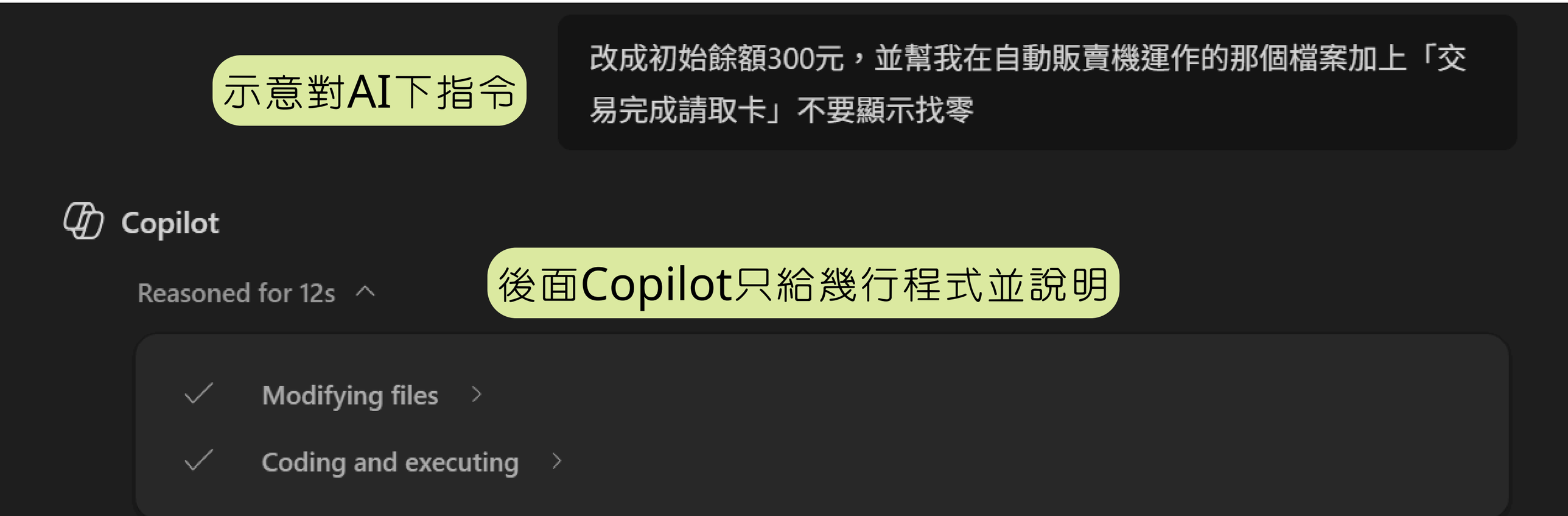
自動販賣機

U1343020 高巧芸 U1343021 何凱琍 U1343030 許坊琳

動機&過程

多元支付的自動販賣機是我們這些八甲新宿住宿生不可或缺的日常，半夜肚子餓就只有宿舍一樓的販賣機可以拯救我們，使用悠遊卡也很方便，不需要與時間賽跑找零錢投幣。因此本組想嘗試用python程式復刻出一台用悠遊卡消費的自動販賣機。

因為本組寫程式並不熟練只會課堂上學到的，所以本組程式編寫過程有使用Copilot輔助。明確給予指令使AI生成出程式，再請AI解說程式是如何導出這樣的結果，在不斷溝通後，最後由我們自己去改成更符合原先構思的樣子(因為它有時候會忽略掉我們原先的要求)，從做中學程式編寫。本組三人先各自編寫檔案，最後挑最完整的那個人的版本當本組自動販賣機程式。



程式構思&編寫

在與老師討論過後，我們決定將悠遊卡的餘額檢驗獨立寫一個檔案，並在自動販賣機運作的程式呼叫執行，於是我們寫了兩個*.py*的檔案：

- **悠遊卡餘額檢驗：easycard_service.py**
- **自動販賣機運作：vending_machine.py**

我們設定悠遊卡的卡號為八個數字號碼，若八碼數字可以被*3*整除，就設定為「先天」餘額不足的卡號，其餘卡號都有初始餘額*100*元。自動販賣機的運作流程則是選代碼、輸入卡號、商品庫存與餘額減少、交易成功或失敗。

程式構思&編寫

- **悠遊卡餘額檢驗：easycard_service.py**

1. 先不做任何假設，設定初始餘額為*100*元
2. 基礎驗證卡號須為八位數
3. 基礎驗證八位數字加總為*3*的倍數視為「 餘額不足 」
4. 將所有八位數字排列組合都視為存在，僅需格式正確即可
5. 扣款回饋分成「 格式錯誤 」、「 餘額不足 」與「 扣款金額須為正整數 」
6. 示範檢查*8*組八位數字，確認無誤即可使用

```python
from typing import Tuple, Dict, Any

class EasyCardService:
    def __init__(self, initial_balance: int = 100) -> None:
        self._accounts: Dict[str, int] = {}
        self._initial_balance = int(initial_balance)

    def is_valid_card(self, card_number: str) -> bool:
        return isinstance(card_number, str) and card_number.isdigit() and len(card_number) == 8

    def digit_sum(self, card_number: str) -> int:
        return sum(int(d) for d in card_number)

    def is_insufficient_by_rule(self, card_number: str) -> bool:
        return self.digit_sum(card_number) % 3 == 0
```

設定初始餘額100元

設定先天餘額不足條件

```python
    # --- 卡片存在與餘額 ---
    def card_exists(self, card_number: str) -> bool:
        """
        所有 8 位數字卡號皆視為存在(考慮所有排列組合)。
        僅需格式正確即可。
        """
        return self.is_valid_card(card_number)

    def ensure_account_initialized(self, card_number: str) -> None:
        if card_number not in self._accounts:
            # 依規則決定初始餘額(3 的倍數 → 0;其餘 → initial_balance)
            self._accounts[card_number] = 0 if self.is_insufficient_by_rule(card_number) else self._initial_balance

    def get_balance(self, card_number: str) -> int:
        """回傳卡片餘額;若首次使用則依規則建立初始餘額。"""
        if not self.is_valid_card(card_number):
            return -1
        self.ensure_account_initialized(card_number)
        return self._accounts[card_number]
```

設定先天餘額不足=0元

```python
    # --- 扣款流程 ---
    def has_sufficient_balance(self, card_number: str, amount: int) -> bool:
        bal = self.get_balance(card_number)
        return bal >= 0 and bal >= amount

    def charge(self, card_number: str, amount: int) -> Tuple[bool, Dict[str, Any]]:
        """
        嘗試扣款指定金額。
        規則:若卡號數字和為 3 的倍數 → 直接回覆餘額不足;其餘照常扣款。
        回傳 (True, {'new_balance': int}) 或 (False, {'error': str})。
        """
        if not self.is_valid_card(card_number):
            return False, {'error': '卡號格式錯誤,需為 8 位數字'}
        if amount <= 0:
            return False, {'error': '扣款金額需為正整數'}

        # 依規則判斷:3 的倍數 → 餘額不足
        if self.is_insufficient_by_rule(card_number):
            # 確保帳戶存在並為 0(保持一致的 get_balance 顯示)
            self.ensure_account_initialized(card_number)
            return False, {'error': '餘額不足'}

        # 其餘卡號 → 有餘額(首次使用建立初始餘額),再檢查金額
        self.ensure_account_initialized(card_number)
        if not self.has_sufficient_balance(card_number, amount):
            return False, {'error': '餘額不足'}

        # 扣款
        self._accounts[card_number] -= amount
        return True, {'new_balance': self._accounts[card_number]}
```

```python
if __name__ == '__main__':
    svc = EasyCardService(initial_balance=100)
    samples = ['12345678', '87654321', '11112222', '22223333', '33334444', '13572468', '00000000', '99999999']
    print('示範檢查:')
    for c in samples:
        bal = svc.get_balance(c)
        rule = '3 的倍數 → 餘額不足' if svc.is_insufficient_by_rule(c) else '非 3 的倍數 → 有餘額'
        print(f'卡號 {c} | 初始餘額: {bal} | 規則:{rule}')
```

```
In [1]: %runfile 'C:/Users/kylie/Downloads/自動販賣機python/easycard_service.py' --wdir
示範檢查:
卡號 12345678 | 初始餘額: 0 | 規則:3 的倍數 → 餘額不足
卡號 87654321 | 初始餘額: 0 | 規則:3 的倍數 → 餘額不足
卡號 11112222 | 初始餘額: 0 | 規則:3 的倍數 → 餘額不足
卡號 22223333 | 初始餘額: 100 | 規則:非 3 的倍數 → 有餘額
卡號 33334444 | 初始餘額: 100 | 規則:非 3 的倍數 → 有餘額
卡號 13572468 | 初始餘額: 0 | 規則:3 的倍數 → 餘額不足
卡號 00000000 | 初始餘額: 0 | 規則:3 的倍數 → 餘額不足
卡號 99999999 | 初始餘額: 0 | 規則:3 的倍數 → 餘額不足
```

程式構思&編寫

- **自動販賣機運作：vending_machine.py**

1. 呼叫*easycard_service.py*
2. 列出商品清單並排版對齊，使用者輸入商品代碼或離開
3. 設定商品代碼錯誤與商品缺貨的回饋
4. 使用者輸入卡號，設定重新輸入卡號的回饋
5. 設定交易成功商品庫存減少、悠遊卡餘額減少
6. 設定交易失敗卡號輸入3次、餘額不足的回饋
7. 返回商品清單

```python
from easycard_service import EasyCardService

ITEMS = {
    'A01': {'name': '生活泡沫綠茶', 'price': 25, 'stock': 10},
    'B02': {'name': '百事可樂鋁罐', 'price': 30, 'stock': 2},
    'C03': {'name': '綜合蔓越莓汁', 'price': 29, 'stock': 5},
}

def print_menu():
    print('=' * 50)
    print('【悠遊卡自動販賣機】')
    print('商品清單:')
    print('-' * 50)
    print(f"{'代碼':<6}{'商品名稱':<14}{'價格':<8}{'庫存':<8}")
    print('-' * 50)
    for code, info in ITEMS.items():
        print(f"{code:<8}{info['name']:<12}{info['price']:<10}{info['stock']:<8}")
    print('-' * 50)
    print('輸入商品代碼購買後按 Enter，或輸入 Q 離開。')
    print('=' * 50)

def ask_item_code() -> str:
    return input('請輸入商品代碼:').strip().upper()
```

呼叫悠遊卡餘額檢驗程式

設定初始數值

列商品清單

```python
def validate_item(code: str) -> bool:
    if code not in ITEMS:
        print('⚠️ 商品代碼錯誤，請重新輸入。')
        return False
    if ITEMS[code]['stock'] <= 0:
        print('⚠️ 此商品目前缺貨。')
        return False
    return True

def main():
    svc = EasyCardService(initial_balance=100)

    while True:
        print_menu()
        code = ask_item_code()

        if code == 'Q':
            print('系統結束，感謝使用！')
            break

        if not validate_item(code):
            input('按 Enter 回到清單...')
            continue
        item = ITEMS[code]
        name = item['name']
        price = item['price']
        print(f"您選擇:{name}，價格:{price} 元")

        combined_attempts = 0
        max_attempts = 3

        while True:
            card_number = input('請輸入悠遊卡卡號(8 碼):').strip()
            error_msg = None

            if not (card_number.isdigit() and len(card_number) == 8):
                error_msg = '卡號格式錯誤，需為 8 位數字'
                combined_attempts += 1
            else:
                ok, info = svc.charge(card_number, price)
                if ok:
                    ITEMS[code]['stock'] -= 1
                    remaining_bal = info.get('new_balance', None)
                    print('✅ 交易成功！')
                    if remaining_bal is not None:
                        print(f'悠遊卡剩餘餘額:{remaining_bal} 元')
                    print('交易完成請取卡，謝謝光臨！')
                    input('按 Enter 回到清單...')
                    break
                else:
                    error_msg = info.get('error', '未知錯誤')
                    combined_attempts += 1
```

```
            remaining = max_attempts - combined_attempts
            if remaining > 0:
                print(f'❌ 交易失敗:{error_msg}。可再試 {remaining} 次。')
                continue
            else:
                print('❌ 交易失敗(重試達 3 次)，返回商品清單。')
                input('按 Enter 回到清單...')
                break

if __name__ == '__main__':
    main()
```

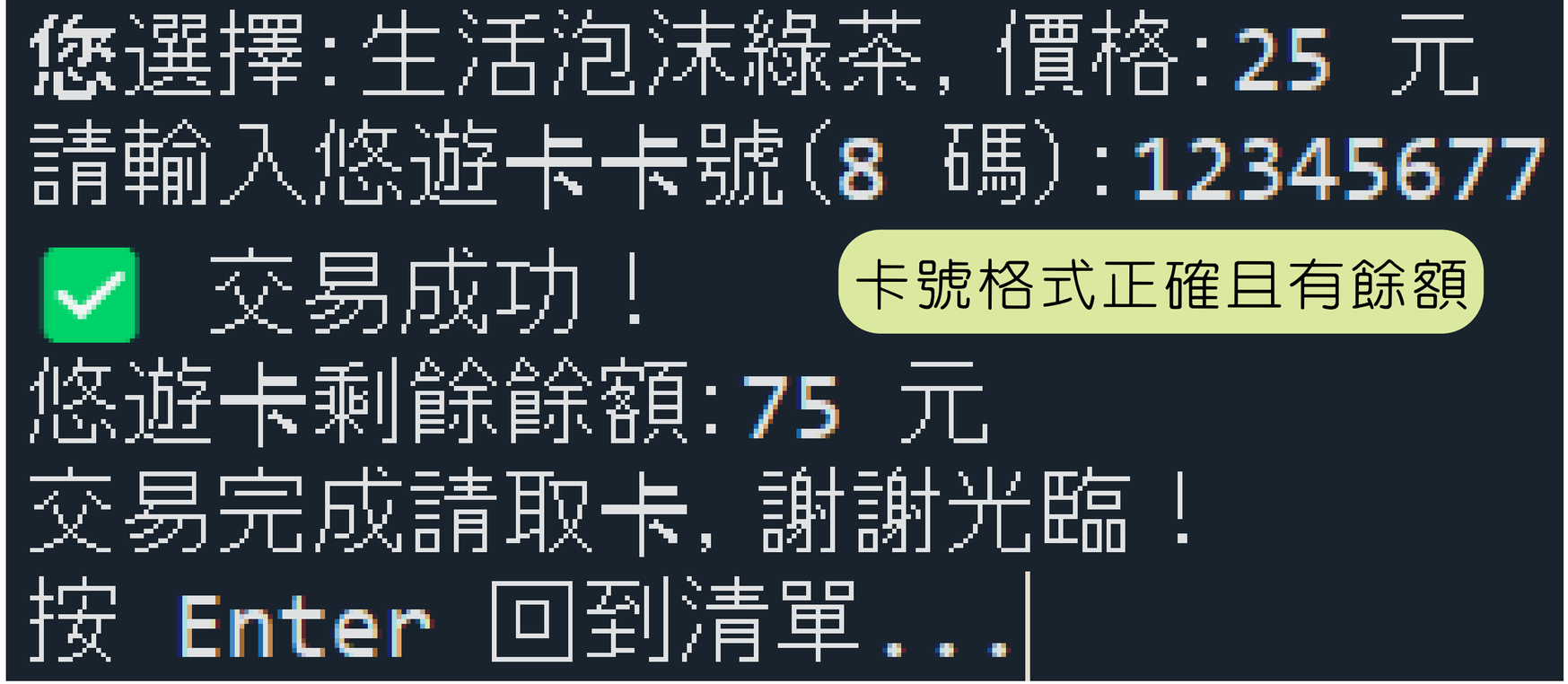
回到商品清單不停迴圈(直到使用者按Q離開)

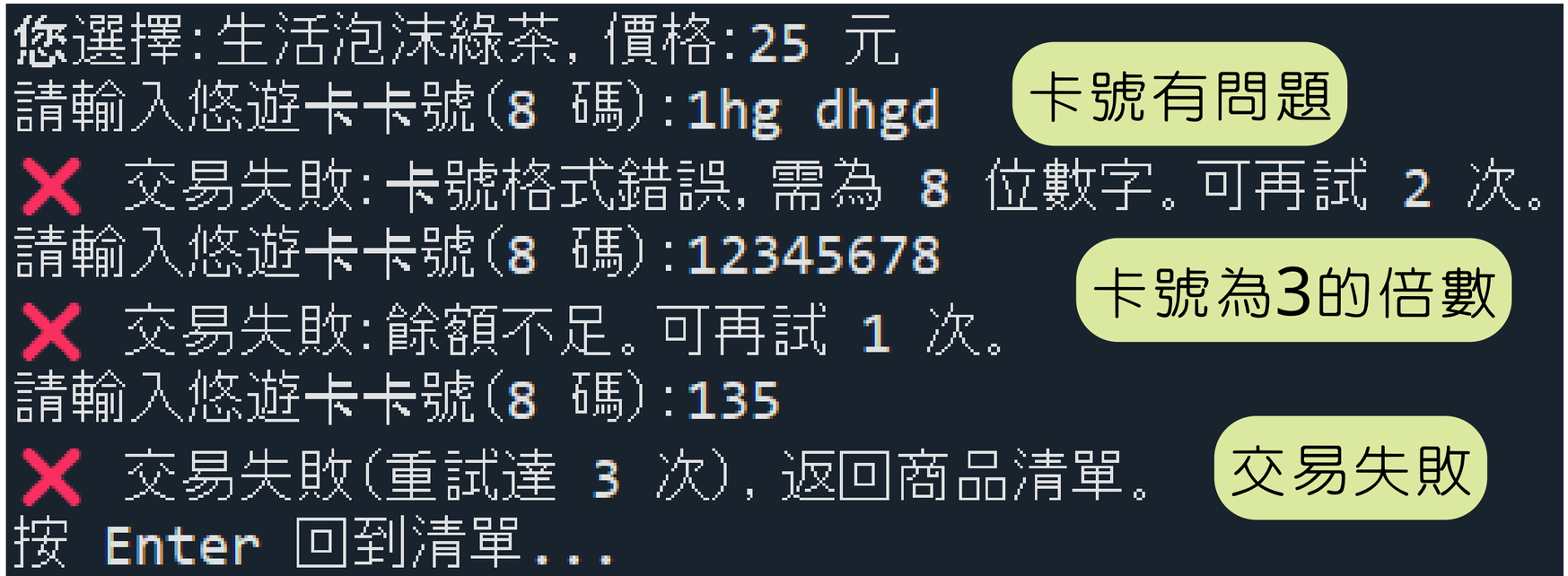
```
In [1]: %runfile 'C:/Users/kylie/Downloads/自動販賣機python/vending_machine.py' --wdir
==================================================
【悠遊卡自動販賣機】
商品清單:
--------------------------------------------------
代碼     商品名稱              價格       庫存
--------------------------------------------------
A01      生活泡沫綠茶          25         10
B02      百事可樂鋁罐          30         2
C03      綜合蔓越莓汁          29         5
--------------------------------------------------
輸入商品代碼購買後按 Enter，或輸入 Q 離開。
==================================================
請輸入商品代碼:A
⚠️ 商品代碼錯誤，請重新輸入。
按 Enter 回到清單...
```

輸入代碼錯誤

```
==========================================================
【悠遊卡自動販賣機】
商品清單:
----------------------------------------------------------
代碼      商品名稱                  價格        庫存
----------------------------------------------------------
A01       生活泡沫綠茶              25          10
B02       百事可樂鋁罐              30          2
C03       綜合蔓越莓汁              29          5
----------------------------------------------------------
輸入商品代碼購買後按 Enter，或輸入 Q 離開。
==========================================================
請輸入商品代碼:A01
您選擇:生活泡沫綠茶，價格:25 元
請輸入悠遊卡卡號(8 碼):
```

輸入代碼正確


您選擇：生活泡沫綠茶，價格：25 元
請輸入悠遊卡卡號（8 碼）：12345677
交易成功！
卡號格式正確且有餘額
悠遊卡剩餘餘額：75 元
交易完成請取卡，謝謝光臨！
按 Enter 回到清單...


您選擇：生活泡沫綠茶，價格：25 元
請輸入悠遊卡卡號（8 碼）：1hg dhgd
卡號有問題
交易失敗：卡號格式錯誤，需為 8 位數字。可再試 2 次。
請輸入悠遊卡卡號（8 碼）：12345678
卡號為3的倍數
交易失敗：餘額不足。可再試 1 次。
請輸入悠遊卡卡號（8 碼）：135
交易失敗
交易失敗（重試達 3 次），返回商品清單。
按 Enter 回到清單...

您選擇：百事可樂鋁罐，價格：30 元
請輸入悠遊卡卡號（8 碼）：12345677
✅ 交易成功！
悠遊卡剩餘餘額：15 元
餘額不夠再買一瓶(非先天)
交易完成請取卡，謝謝光臨！
按 Enter 回到清單...
==========
【悠遊卡自動販賣機】
商品清單：

代碼　　商品名稱　　價格　　庫存

A01　　生活泡沫綠茶　　25　　9
B02　　百事可樂鋁罐　　30　　0
C03　　綜合蔓越莓汁　　29　　5
庫存不足

輸入商品代碼購買後按 Enter，或輸入 Q 離開。
==========
請輸入商品代碼：B02
⚠️ 此商品目前缺貨。
缺貨回饋
按 Enter 回到清單...

```
==========================================================
【悠遊卡自動販賣機】
商品清單：
----------------------------------------------------------
代碼      商品名稱                  價格        庫存
----------------------------------------------------------
A01       生活泡沫綠茶              25          9
B02       百事可樂鋁罐              30          0
C03       綜合蔓越莓汁              29          5
----------------------------------------------------------
輸入商品代碼購買後按 Enter，或輸入 Q 離開。
==========================================================
請輸入商品代碼：C03
您選擇：綜合蔓越莓汁，價格：29 元
請輸入悠遊卡卡號（8 碼）：12345677
❌ 交易失敗：餘額不足。可再試 2 次。
```

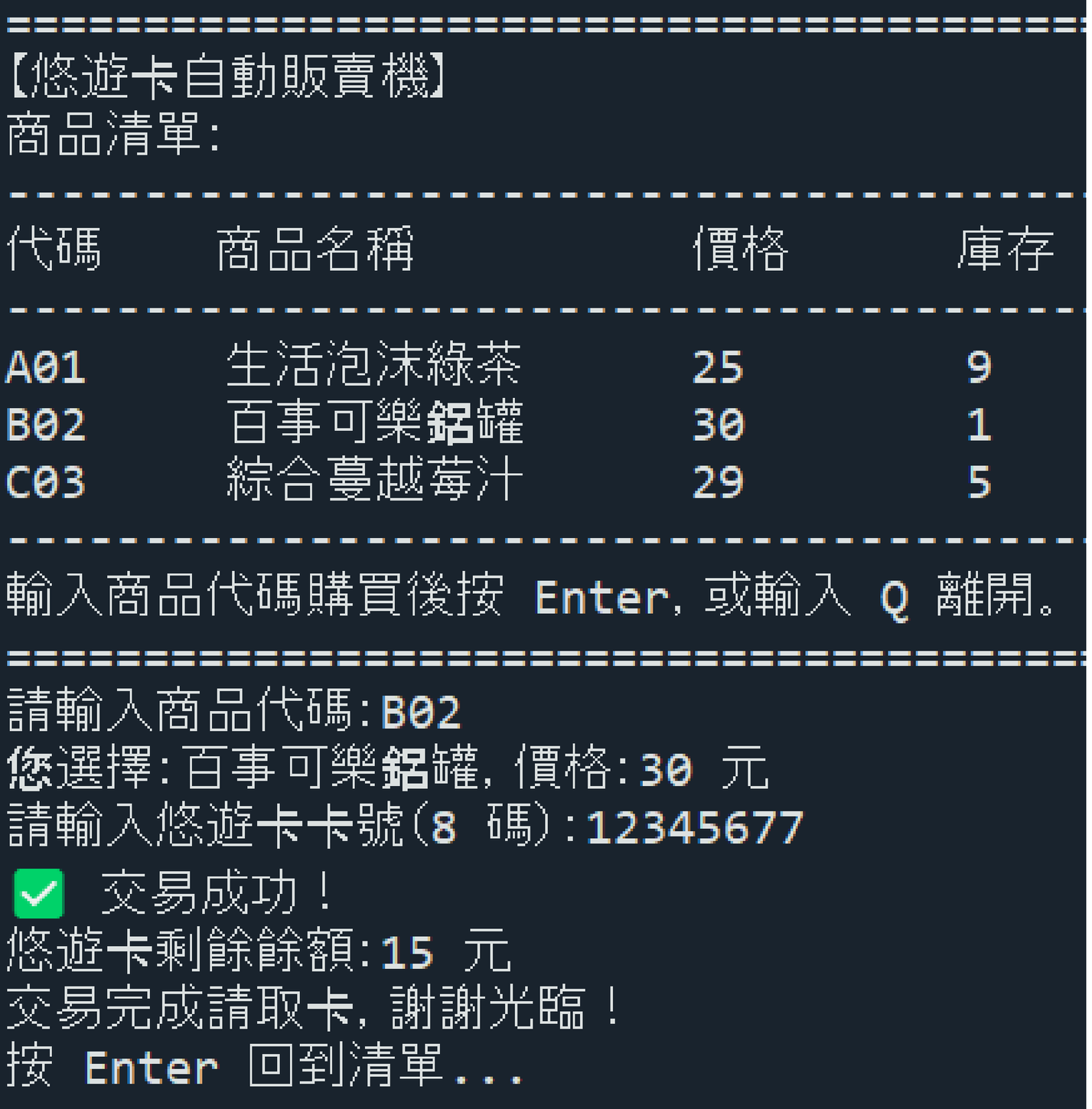
非先天餘額不足回饋

```
==========================================================
【悠遊卡自動販賣機】
商品清單:
----------------------------------------------------------
代碼      商品名稱                  價格        庫存
----------------------------------------------------------
A01       生活泡沫綠茶              25          9
B02       百事可樂鋁罐              30          0
C03       綜合蔓越莓汁              29          5
----------------------------------------------------------
輸入商品代碼購買後按 Enter，或輸入 Q 離開。
==========================================================
請輸入商品代碼: q
系統結束，感謝使用！

In [3]:
```

使用者離開則結束程式迴圈

再次打開全部回到初始值



心得

第一次用*python*做出有實際功能且可以不停使用的程式，非常有成就感覺得很好玩。雖然透過*AI*輔助編寫，而且花很多時間理解，甚至現在其實有些程式細節還看不太懂，但這次經驗也增長了程式編寫與*AI*應用的數位素養，收穫良多。

基礎程式設計與應用_G3

自動販賣機

U1343020 高巧芸 U1343021 何凱琍 U1343030 許坊琳