程式的優化與完整性

學習目標：

- 將程式流程完整做好，符合商業邏輯需求！
- 將重複的程式，進行模組化或是利用類別物件去除！

處理重複的程式！

1. 將地圖程式，利用函式，去除重複的部份: playMap.py

```
import Stores
class playMap1:
    def __init__(self):
        self.__mapEmpty = "　"
        self.__mapWall = "｜"
        self.__mapLine = "－"
        self.myStores = Stores.Stores()

    def printMap(self,userPo):
        # 新程式區塊
        for k in range(1,28):
            # 印出第一、二、三行
            if (( k == 1 ) or (k == 2) or ( k == 3)):
                self.printmap1(k,0,7,userPo)
            # 印出第四行，以及第二十四行
            if (( k == 4 ) or ( k == 24 )):
                print(48*self.__mapLine)
            # 印出第五、六、七行，九、十、十一行，十三、十四、十五行，十七、十八、十九行，二十一、二十二、二十三行
            if (( k == 5 ) or (k == 6) or ( k == 7)):
                self.printmap2(k,23,7,userPo)
            if (( k == 9 ) or (k == 10) or ( k == 11)):
                self.printmap2(k,22,8,userPo)
            if (( k == 13 ) or (k == 14) or ( k == 15)):
                self.printmap2(k,21,9,userPo)
            if (( k == 17 ) or (k == 18) or ( k == 19)):
                self.printmap2(k,20,10,userPo)
            if (( k == 21 ) or (k == 22) or ( k == 23)):
                self.printmap2(k,19,11,userPo)
            # 印出第八、十二、十六、二十行
            if (( k == 8 ) or (k == 12) or ( k == 16) or ( k == 20)):
                print(7*self.__mapLine + 34*self.__mapEmpty + 7*self.__mapLine)
            # 印出第二十五、二十六、二十七行
            if (( k == 25 ) or (k == 26) or ( k == 27)):
                self.printmap1(k,18,11,userPo)


    # 列印地圖程式
    def printmap1(self,k,min,max,userPo):
        if (max-min) > 0 :
            j = 1
        else:
            j = -1
        if ((k == 1) or (k == 25)):
            for i in range(min,max,j):
                if (self.myStores.getStoreData(str(i))[2] == "-1"):
                    owner = "　"
                else:
                    owner = self.transferNo(self.myStores.getStoreData(str(i))[2])
                print(self.__mapEmpty + self.getStoreName(self.myStores.getStoreData(str(i))[1]) + owner,end = '')
                if ((i < 6) or (i > 12)):
                    print(self.__mapWall,end = '')
                else:
                    print()

        elif (( k == 2) or (k == 26)):
            for i in range(min,max,j):
                print(self.__mapEmpty + self.getStoreName(self.transferNo(self.myStores.getStoreData(str(i))[3])) + self.__mapEmpty,end = '')
                if ((i < 6) or (i > 12)):
                    print(self.__mapWall,end='')
                else:
                    print()

        elif (( k == 3) or (k == 27)):
            po_tmp = ""
            for i in range(min,max,j):
                po_tmp = self.__mapEmpty
                for l in range(len(userPo)):
                    if (userPo[l] == str(i)):
                        po_tmp = po_tmp + self.transferNo(str(l+1))
                    else:
                        po_tmp = po_tmp + self.__mapEmpty

                # 若人數不足四人則補足其它空間
                if (len(userPo) < 4):
                    po_tmp = po_tmp + (4-len(userPo))*self.__mapEmpty
                po_tmp = po_tmp + self.__mapEmpty
                if ((i < 6) or (i > 12)):
                    print(po_tmp + self.__mapWall,end = '')
                else:
                    print(po_tmp,end = '')
            print()

    def printmap2(self,k,min,max,userPo):
        for i in (min,max):
                if (self.myStores.getStoreData(str(i))[2] == "-1"):
                    owner = "　"
                else:
                    owner = self.transferNo(self.myStores.getStoreData(str(i))[2])
        if (( k == 5) or ( k == 9) or( k == 13) or( k == 17) or( k == 21)):
            lines = ""
            lines = lines + self.__mapEmpty + self.getStoreName(self.myStores.getStoreData(str(min))[1]) + owner + self.__mapWall
            lines = lines + 34*self.__mapEmpty
            lines = lines + self.__mapWall + self.__mapEmpty + self.getStoreName(self.myStores.getStoreData(str(max))[1]) + owner
            print(lines)

        elif (( k == 6) or ( k == 10) or( k == 14) or( k == 18) or( k == 22)):
            lines = ""
            lines = lines + self.__mapEmpty + self.getStoreName(self.transferNo(self.myStores.getStoreData(str(min))[3])) + owner + self.__mapWall
            lines = lines + 34*self.__mapEmpty
            lines = lines + self.__mapWall + self.__mapEmpty + self.getStoreName(self.transferNo(self.myStores.getStoreData(str(max))[3])) + owner
            print(lines)

        elif (( k == 7) or ( k == 11) or( k == 15) or( k == 19) or( k == 23)):
            po_tmp = ""
            lines = self.__mapEmpty
            for j in range(len(userPo)):
                if (userPo[j] == str(str(min))):
                    po_tmp = po_tmp + self.transferNo(str(j+1))
                else:
                    po_tmp = po_tmp + self.__mapEmpty
            # 若人數不足四人則補足其它空間
            if (len(userPo) < 4):
                po_tmp = po_tmp + (4-len(userPo))*self.__mapEmpty
            po_tmp = po_tmp + self.__mapEmpty
            lines = lines + po_tmp + self.__mapWall
            po_tmp = ""
            lines = lines + 34*self.__mapEmpty
            for j in range(len(userPo)):
                if (userPo[j] == str(str(max))):
                    po_tmp = po_tmp + self.transferNo(str(j+1))
                else:
                    po_tmp = po_tmp + self.__mapEmpty
            po_tmp = po_tmp + self.__mapEmpty
            lines =  lines + self.__mapWall + self.__mapEmpty + po_tmp
            print(lines)

    # 控制每一行的格式大小
    def getStoreName(self,data):
        storeName = ""
        if (len(data) <= 4):
            storeName = data + (4-len(data))*"　"
        return storeName

    # 半形全形轉換功能
    def transferNo(self,data):
        nums = (0,"０",1,"１",2,"２",3,"３",4,"４",5,"５",6,"６",7,"７",8,"８",9,"９")
        tmp = []
        dataleng = len(data)
        for j in range(0,dataleng):
            tmp.append(0)

        newdata = ""
        for i in range(1,dataleng+1):
            tmp[(dataleng-i)] = int(data)%10
            data = int(int(data) / 10)

        for i in range(0,len(tmp)):
            newdata += nums[nums.index(tmp[i])+1]

        return newdata

if __name__ == "__main__":
    myMap = playMap1()
    userPo = ['11']
    myMap.printMap(userPo)
```

2. 修改 main.py 程式，將玩家資訊，導入 Player.py:

```
(前方略過....)
## 清除舊資料
def clearOldData():
    files = open('players.csv','w',encoding='utf-8')
    files.truncate()
    files.close()
(中間略過...)
   # 逐次產生玩家名稱、玩家代號、玩家初始遊戲幣、玩家初始位置等物件內容
    clearOldData()
    for i in range(players_num):
        players.append(Player.Player())
        # 要求玩家輸入玩家名稱
        players[i].setName(input("請輸入玩家名稱:"),i)
```

3. 修改玩家程式 Player.py ，將資訊寫入 players.csv ，並且設置可讀出方式：

```
(前方略過...)
# 初始化玩家，每人發 20000 遊戲幣以及出發位置為 0
    def __init__(self,money = 20000, po = 0):
        self.__money = money
        self.__po = po
        self.__status = 0

    # 設定玩家名稱
    def setName(self,name,id):
        self.__name = name
        self.__id = id
        with open('players.csv','a',newline='') as csvfile:
            writer = csv.writer(csvfile, delimiter=',')
            writer.writerow([self.__id,self.__name,self.__money,self.__po,self.__status])
(以下略過....)
```

4. 編寫訊息程式，將訊息導入文字檔，再由地圖檔輸出：Messages.py

```
class Messages:

    def inputData(self,messages):
        self.__messages = messages
        # 開啟檔案，設定成可寫入模式
        new_files = open("messages.txt","w", encoding='utf-8')

        # 將串列寫入檔案中
        new_files.writelines(messages)

        # 關閉檔案
        new_files.close()

    def outputData(self):
        files = open("messages.txt","r", encoding='utf-8')
        messages = []
        messages = files.readlines()
        self.__messages = messages[0]
        return self.__messages

if __name__ == "__main__":
    news = Messages()
    news.inputData("測試用")
    print(news.outputData())
```

5. 將主要流程的程式，功能完備！

```
# 引用 random 類別中的 randrange() 函數
from random import randrange

# 引用 Player 物件
import Player

# 引用 Chance 物件
import Chance
import Destiny
# 引用 Stores 物件
import Stores

# 引用 playMap 物件
import playMap

import Messages

# 常用函式、參數設定區域
## 遊戲方格總數
areas = 24

## 處理玩家是否有經過「開始」
def playerPo(steps):
    if (steps >= areas):
        nums = (steps % areas)
        return nums
    else:
        return steps

## 清除舊資料
def clearOldData():
    files = open('players.csv','w',encoding='utf-8')
    files.truncate()
    titles = "id,name,money,po,status\n"
    files.writelines(titles)
    files.close()
    files = open('messages.txt','w',encoding='utf-8')
    files.truncate()
    titles = "遊戲即將開始...."
    files.writelines(titles)
    files.close()
```

```
# 程式流程開始
# 使用 if __name__
if __name__ == "__main__":

    # 要求玩家要輸入遊戲人數
    players_num = eval(input("請輸入玩家人數："))

    # 建立玩家物件
    players = []

    # 按照遊戲人數，使用 Player 類別
    # 逐次產生玩家名稱、玩家代號、玩家初始遊戲幣、玩家初始位置等物件內容
    clearOldData()
    for i in range(players_num):
        players.append(Player.Player())
        # 要求玩家輸入玩家名稱
        players[i].setName(input("請輸入玩家名稱:"),i)

    # 設定玩家位置值
    players_po = []
    for i in range(players_num):
        players_po.append('0')

    # 設定玩家順序值
    i = 0
    myMap = playMap.playMap()

    # 設定訊息存放物件
    news = Messages.Messages()
    news.inputData("請按下《ＥＮＴＥＲ》進行遊戲")
    myMap.printMap(players_po)
    input()
    # 開始進行遊戲
    while True:
    ##### a.) 印出地圖
        news.inputData(myMap.transferNo(str(i+1)) + "號玩家按下《ＥＮＴＥＲ》進行遊戲")
        myMap.printMap(players_po)
        input()
    ##### b.) 擲骰子
        oldpo = players[i].getPo()
        newstep = randrange(1,6)
        news.inputData(myMap.transferNo(str(i+1)) + "號玩家擲骰子：" + myMap.transferNo(str(newstep)) + "點")
        myMap.printMap(players_po) # 印地圖
        # 設定玩家新的位置
        players[i].setPo(newstep)
    ##### c.) 移動到骰子點數的框格
        newpo = players[i].getPo()

        #  I. 可能經過起點
        if ((int(newpo/areas) > int(oldpo/areas)) and ((newpo % areas) != 0)):
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家越過「開始」位置《ＥＮＴＥＲ》")
            players[i].setMoney(2000,i)
            myMap.printMap(players_po)
            input()
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家得２０００《ＥＮＴＥＲ》")

        newpo = playerPo(newpo)
        players_po[i] = str(newpo)
        myMap.printMap(players_po)
        input()
        news.inputData(myMap.transferNo(str(i+1)) + "號玩家在新位置：" + myMap.transferNo(str(newpo)) + "《ＥＮＴＥＲ》")
        myMap.printMap(players_po)
        input()
        #  II. 可能落在邊角框格
        if (newpo == 0):
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家回到「開始」位置《ＥＮＴＥＲ》")
            myMap.printMap(players_po)
            input()
        elif (newpo  == 6):
            #print(" 休息一天")
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家，無事休息一天《ＥＮＴＥＲ》")
            myMap.printMap(players_po)
            input()
        elif (newpo  == 18):
            #print(" 再玩一次")
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家，再玩一次《ＥＮＴＥＲ》")
            myMap.printMap(players_po)
            input()
            continue
        elif (newpo == 12):
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家，休息三次《ＥＮＴＥＲ》")
            myMap.printMap(players_po)
            input()
        #  III. 可能是在機會與命運框格
        ## 機會的地圖編號是 3,15 兩個號碼
        elif ((newpo == 3) or (newpo == 15)):
            myChance = Chance.Chance()
            chances = myChance.choice()
            players[i].setMoney(int(chances[1]),i)
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家中機會：" + chances[0])
            myMap.printMap(players_po)
            input()
        ## 命運的地圖編號是 9,21 兩個號碼
        elif ((newpo == 9) or (newpo == 21)):
            myDestiny = Destiny.Destiny()
            destines = myDestiny.choice()
            players[i].setMoney(int(destines[1]),i)
            news.inputData(myMap.transferNo(str(i+1)) + "號玩家中命運：" + destines[0])
            myMap.printMap(players_po)
            input()

        #  IV. 可能是在地產框格
        else:
            playerStore = Stores.Stores()
            store = playerStore.getStoreData(str(newpo))
            ## 判斷是否有人己取得該地產所有權了
            if store[2] == '-1':
                #print("該地產無人所有！")
                news.inputData("該地產無人所有！是否買進？（Y｜N）")
                myMap.printMap(players_po)
                results = input()
                if ((results == 'Y') or (results == 'y')):
                    store[2] = str(i+1)
                    playerStore.setStoreData(store)
                    players[i].setMoney(0-int(store[3]),i)
                    news.inputData(myMap.transferNo(str(i+1)) + "號玩家買進地產：" + store[1])
                    myMap.printMap(players_po)
                    input()
                else:
                    news.inputData(myMap.transferNo(str(i+1)) + "號玩家放棄買進")
                    myMap.printMap(players_po)
                    input()
            else:
                print("該地產為：" + str(players[int(store[2])-1].getName()) + "所有")
            playerStore = None
    ##### e.)
        # 輪至下一位玩家
        i = i + 1
        if (i >= players_num):
            i = i - players_num

    ##### f.) 結束遊戲條件
        ends = input("是否結束遊戲？Y：是　N：繼續")
        if ((ends == "Y") or (ends == "y")):
            break
```

6. 持續修正至完成為止！