Final Project

陳韡承 學號 : 40947016S

大綱

- 檔案
- Structure結構
- 主要函式
- 職業函式
- 卡片函式
- 存檔/讀檔函式
- 編輯器函式

檔案

- main.c
- sanJuan.c
- sanJuan.h
- editor.c
- editor.h
- saving.c
- saving.h
- struct.h
- text.h
- Makefile
- README.md
- HowToPlay

Struct結構

- define

```
#define u8 uint8_t
typedef char string[600];
```

- card

```
typedef struct _card
{
    u8 place; //卡片位置(牌堆、棄牌堆、手牌、場上)
    u8 played;//是否打出
    u8 id;//卡片id
    string cardName;//卡片名稱
    string description;//卡片敘述
    u8 type;//卡片類別
    u8 phase;//卡片效果執行階段
    u8 cost;//卡片費用
    u8 vp;//勝利點數
    u8 hasProduct;//是否有貨物
    u8 extraValue;//額外分數
} card;
```

- role

```
typedef struct _role
{
    u8 used;//職業是否已被選取
    u8 id;//職業id
    string roleName;//職業名稱
    string desription;//職業敘述
} role;
```

- player

```
typedef struct _player
{
    u8 cardCount;//手牌數量
    u8 maxCard;//手牌上限
    u8 boardCount;//場地上牌的數量
    card hand[20];//手牌
    card board[15];//場上的牌
    u8 point;//分數
    u8 playerOrder;//玩家順序
    u8 extraProduce;//溝渠token
    u8 extraTrade;//交易所token
    u8 extraCard;//辦公處token
    u8 library;//圖書館token
    u8 archive;//檔案室token
    u8 quarry;//採石場token
    u8 smithy;//鐵匠鋪token
    u8 productsCount;//場上貨品數量
    u8 crane;//起重機token
    u8 blackMarket;//黑市token
} player;
```

主要函式

```
void init(u8);
```
初始化各變數、陣列，以帶入變數去處理玩家數，遊戲開始前需要呼叫，若是有讀取存檔則不會呼叫
```
void menu();
```
顯示主畫面及處理選項輸入，程式執行時或是
```
void about();
```
顯示關於此專案的內容
```
void setting();
```
帶到設定的畫面，以及處理各設定的變數(語言、遊戲結束條件、cpu難度)
```
void loadGame();
```
讀去遊戲存檔，處理存檔的行動順序
```
void saveGame();
```
呼叫save.c裡面的save()去存檔
```
void mainGame();
```
遊戲開始後的主體，呼叫各個需要的函式(init(),loadGame()等等)，內有遊戲進行中的迴圈處理各行動
```
void GameEnd();
```
遊戲結束後呼叫，處理遊戲結束後的卡片效果和分數計算，並顯示遊戲結果
```
void printPlayerStatus(player *);
```
顯示行動畫面，可以選擇顯示玩家的狀態(手牌、場地上的牌)，處理呼叫接下來的行動函式
```
void shuffle();
```
就洗牌

```
void draw(player *p);
```
將牌組最上面的牌給予帶入的玩家，確保牌組中有牌，若沒有則呼叫recycleCard()
```
void recycleCard();
```
將棄牌堆的牌放回牌組並洗牌
```
void discardCard(player *, u8 choosed);
```
將玩家手牌中編號為choosed的牌丟棄，在各個需要將牌從手上移走(除了建造)會使用
```
void readDes(u8 player, u8 mode);
```
顯示卡片檢視的畫面，mode 1 顯示場地的牌，mode 0 顯示手牌，可以檢視各牌的詳細資訊
```
void printPlayerCard(player *);
```
顯示玩家的手牌，有需要選取手牌的功能就會呼叫
```
void printPlayerBoard(player *);
```
顯示玩家的場上，有需要選取場上的牌的功能就會呼叫
```
void chooseRole(u8 governor);
```
顯示職業選擇的畫面，選擇要行動的職業，並呼叫該行動的函式
```
u8 chooseAction(u8 player);
```
處理卡片檢視時的選項
```
void produce(player *, u8 produceBuilding);
```
生產時呼叫此函式
```
void sell(player *, u8 produceBuilding, u8 value );
```
販售貨品時呼叫此函式
```
void printCPUstatus(player *);
```
顯示電腦玩家的狀態(手牌數，場上的狀態)
```
void computerAction(player *);
```
處理電腦玩家的行動，難度由全域變數cpuLevel決定
```
void roleReset(role *r*[]);
```
回合結束後呼叫，將職業重製
```
void reduceCard(player *);
```
回合開始時呼叫，使各玩家手牌不超過上限
```
u8 searchCard(player *, u8 );
```
需要在手牌找牌時呼叫，回傳值為是否找到該牌，1為是、0為否
```
u8 searchBoard(player *, u8 );
```
需要在場上找牌時呼叫，回傳值為是否找到該牌，1為是、0為否
```
void giveCard(player *, card );
```
給予特定牌給玩家，在市長職業或是從牌組拿牌是會呼叫
```
void openEditor();
```
開啟存檔編輯器

職業函式

```
void builder(player p[], u8 goveror);
```
建築師函式，內部處理各個玩家選擇建築及是否能負擔費用的確認，還有各建築師階段有效果的卡片函式呼叫
```
void producer(player p[], u8 goveror);
```
生產者函式，處理玩家要生產的建築物及是否能夠生產的判斷，還有各生產者階段有效果的卡片函式呼叫
```
void trader(player p[], u8 goveror);
```
商人函式，處理玩家要販售的建築物及是否能夠販售的判斷，還有各商人階段有效果的卡片函式呼叫
```
void councilor(player p[], u8 goveror);
```
市長函式，處理玩家選擇保留的牌，還有各市長階段有效果的卡片函式呼叫
```
void prospector(player p[], u8 goveror);
```
掏金者函式，處理玩家抽牌及呼叫金礦功能

卡片函式

下列所有函式皆會先呼叫searchBoard()來判斷是否有該卡
未特別解釋的函式皆為修改player裡面的token及提醒玩家有此效果

```
void tower(player *p);
```

調整手牌上限

```
void chapel(player *p);
```

處理將牌放到禮拜堂底下的行動，及記錄禮拜堂的分數

```
void smithy(player *p);
void poorHouse(player *p);
```

符合條件了話呼叫一次draw()

```
void blackMarket(player *p);
void crane(player *p);
void carpenter(player *p);
```

符合條件了話呼叫一次draw()

```
void quarry(player *p);
void well(player *p);
```

符合條件了話呼叫一次draw()

```
void aqueduct(player *p);
void marketStand(player *p, u8 sellCount);
```

符合條件了話呼叫一次draw()

```
void marketHall(player *p);
```

符合條件了話呼叫一次draw()

```
void tradingPost(player *p);
void archive(player *p);
void perfecture(player *p);
void goldMine(player *p);
```

判斷金礦能力是否發動及處理玩家選牌

```
void library(player *p);
```

下列三個為dummyFunction，方便init而放的函式，無功能

```
void statue(u8 cardowner);
void victoryColumn(u8 cardowner);
void hero(u8 cardowner);
```

最後四個皆為遊戲最後處理額外加分的函式

```
void guildHall(player *p, u8 *vp);
void cityHall(player *p, u8 *vp);
void triumhalArch(player *p, u8 *vp);
void palace(player *p, u8 *vp);
```

存檔/讀檔函式

```
void save(u8 langauge, player p[], u8 playerCount, u8 end, u8 level, card
deck[110], card discard[110],role r[5]);
```

帶入所有savefile structure需要的變數，在內部包成一個struct並寫入輸入的檔案名稱

```
i32 load(u8 langauge,savefile *loadfile);
```

讀取輸入的檔案，確認檔案的存在及檔案為存檔所需格式

編輯器函式

```
void printDeck(card d[110]);
```

印出牌組

```
void printDiscardDeck(card d[110]);
```

印出棄牌堆

```
void printHand(player *p);
```

印出帶入玩家的手牌

```
void printBoard(player *p);
```
印出帶入玩家場上的牌
```
void editHand(player *p, card d[110], card dis[]);
```
顯示編輯介面，呼叫其他編輯用的函式來編輯手牌
```
void editBoard(player *p, card d[110], card dis[]);
```
顯示編輯介面，呼叫其他編輯用的函式來編輯場上的牌
```
void langSet();
```
設定編輯器語言
```
void discardCard(player *p, u8 chosedCard, card discard[110]);
```
將移除的手牌或是場上的牌移入棄牌堆
```
void giveCard(player *p, card c);
```
將指定的牌給予指定玩家，方便其他函式呼叫
```
void addCard(player *p, card c[110]);
```
從牌堆中選擇牌加入指定玩家的手牌，呼叫giveCard()來處理
```
void delCard(player *p, card dis[], card d[110]);
```
從指定玩家的手牌中選擇牌移入棄牌堆，呼叫discardCard()來處理
```
void addBoard(player *p, card c[110]);
```
從牌堆中選擇牌加入指定玩家的場上
```
void delBoard(player *p, card dis[], card d[110]);
```
從指定玩家場上的牌中選擇牌移入棄牌堆