Report_110550085

I.Implementation Detailed

1. Actions Menu

使用 chooseAction(vector<Object*>);函式實作，每個房間只有一個特殊事件（怪物、NPC 或是寶箱）。所以讀取該房間的 Object*向量的第 0 格，判定其 Tag 屬於何者，如果是怪物，那第一個選項就輸出 Retreat，並且用一個變數 monava(monster_availability)告知下方函式此房間有怪物，如果不是，就輸出 Move。接著讀取輸入，如果輸入的是 1 且該房間有怪物，則用 changeroom()將玩家傳回上一個房間。如果輸入 2 則顯示狀態（下述），如果輸入 3 則打開背包，如果輸入 4 則執行該房間的事件，讀取 Tag 後 downcast 成 npcptr/monptr/itemptr 並執行 ptr->triggerEvent(&player);。

相關程式碼：

```
void Dungeon::chooseAction(vector<Object*> action){
    int monava=0;
    cout<<"What do I want to do?"<<endl;
    for(int i=0;i<action.size()+3;i++){
        if(i==0){
            if(action.size()!=0){
                if(action[0]->getTag()=="Monster"){
                cout<<"1. Retreat."<<endl;
                monava=1;
                }
                else{
                    cout<<"1. Move."<<endl;
                }
            }
            else{
                cout<<"1. Move."<<endl;
            }
        }
        else if(i==1){
            cout<<"2. Check Status."<<endl;
        }
        else if(i==2){
            cout<<"3. Open my backpack."<<endl;
        }
        else if(i==3){
            if(action[0]->getTag()=="NPC"){
                NPC* iaction=dynamic_cast<NPC*>(action[0]);
                cout<<i+1<<". Talk to "<<iaction->getName()<<"."<<endl;
            }
            else if(action[0]->getTag()=="Monster"){
                Monster* iaction=dynamic_cast<Monster*>(action[0]);
                cout<<i+1<<". Fight with "<<iaction->getName()<<"."<<endl;
            }
            else if(action[0]->getTag()=="Key"){
                cout<<i+1<<". Open the Chest."<<endl;
            }
            else{
                i++;
            }
        }
    }
    int choseAction;
    while(cin>>choseAction){
        if(choseAction>action.size()+3||(choseAction<=0)){
            cout<<"Invalid Input."<<endl;
            continue;
        }
```

```
        else{
            if(choseAction==1){
                if(monava==1){
                    player.changeRoom(player.getPreviousRoom());
                    cout<<endl<<"Successfully retreated!"<<endl<<endl;
                }
                else{
                    handleMovement();
                }
                break;
            }
            else if(choseAction==2){
                player.triggerEvent(&player);
                break;
            }
            else if(choseAction==3){
                player.openBackpack();
                break;
            }
            else if(choseAction==4){
                if(action[0]->getTag()=="NPC"){
                    NPC* npcptr=dynamic_cast<NPC*>(action[0]);
                    npcptr->triggerEvent(&player);

                }
                else if(action[0]->getTag()=="Monster"){
                    Monster* monptr=dynamic_cast<Monster*>(action[0]);
                    monptr->triggerEvent(&player);
                }
                else if(action[0]->getTag()=="Key"){
                    Item* itemptr=dynamic_cast<Item*>(action[0]);
                    itemptr->triggerEvent(&player);
                    vector<Object*> blank;
                    player.getCurrentRoom()->setObjects(blank);
                }
                break;
            }
        }
    }
}
```

2. Movement

使用 handleMovement 函式實作，讀取輸入，1 代表往上，2 代表往下，3 代表往左，4 代表往右，如果其方向是空指標，則輸出(Unable to go.)。較特別的是因為第十四間房間是鎖住的，如果讀取到玩家所在房間的 Index 是 13，對上面房間則輸出(Requires a Key.)。根據輸入將玩家傳送到相應的房間。若是需要鑰匙的情況，檢查玩家身上所有東西的 Tag，若有 getTag()=="Key"，則將 goable 定為 1，若否則定為 0，若 Tag==0，輸出玩家身上沒有鑰匙的訊息，並 break;以離開 handleMovement()。

相關程式碼：

```
void Dungeon::handleMovement(){
    cout<<"Where do I want to go?"<<endl;
    cout<<"1. Go up.";
    if(player.getCurrentRoom()->getUpRoom()==nullptr){
        cout<<"(Unable to go.)";
    }
    if(player.getCurrentRoom()->getIndex()==13){
        cout<<"(Requires a key.)";
    }
    cout<<endl;
    cout<<"2. Go down.";
    if(player.getCurrentRoom()->getDownRoom()==nullptr){
        cout<<"(Unable to go.)";
    }
    cout<<endl;
    cout<<"3. Go left.";
    if(player.getCurrentRoom()->getLeftRoom()==nullptr){
        cout<<"(Unable to go.)";
    }
    cout<<endl;
    cout<<"4. Go right.";
    if(player.getCurrentRoom()->getRightRoom()==nullptr){
        cout<<"(Unable to go.)";
    }
    cout<<endl;
```

```
int where;
while(cin>>where){
    if(where==1){
        if(player.getCurrentRoom()->getIndex()==13){
            int goable=0;
            vector<Item> ite=player.getInventory();
            for(int x=0;x<ite.size();x++){
                if(ite[x].getTag()=="Key"){
                    goable=1;
                }
            }
            if(goable==0){
                cout<<"You don't have the Key."<<endl;
                break;
            }
        }
        if(player.getCurrentRoom()->getUpRoom()==nullptr){
            cout<<"Unable to go."<<endl;
            continue;
        }

        else{
            player.setPreviousRoom(player.getCurrentRoom());
            player.setCurrentRoom(player.getCurrentRoom()->getUpRoom());
            break;
        }
    }
    else if(where==2){
        if(player.getCurrentRoom()->getDownRoom()==nullptr){
            cout<<"Unable to go."<<endl;
            continue;
        }

        else{
            player.setPreviousRoom(player.getCurrentRoom());
            player.setCurrentRoom(player.getCurrentRoom()->getDownRoom());
            break;
        }
    }
    else if(where==3){
        if(player.getCurrentRoom()->getLeftRoom()==nullptr){
            cout<<"Unable to go."<<endl;
            continue;
        }
        else{
            player.setPreviousRoom(player.getCurrentRoom());
            player.setCurrentRoom(player.getCurrentRoom()->getLeftRoom());
            break;
        }
    }
    else if(where==4){
        if(player.getCurrentRoom()->getRightRoom()==nullptr){
            cout<<"Unable to go."<<endl;
            continue;
        }
        else{
            player.setPreviousRoom(player.getCurrentRoom());
            player.setCurrentRoom(player.getCurrentRoom()->getRightRoom());
            break;
        }
    }
    else{
        cout<<"Invalid Input."<<endl;
    }
}
```

3. Showing Status

使用 Player::triggerEvent(Object* Objptr);函式實作，輸出包含 Name,CurrentHealth/MaxHealth,Attack,Defense 以及四個部件的 Item，分別是 Helmet, Chestplate, LeftHand 和 RightHand。接著輸出玩家身上的所有 Item。

相關程式碼：

```
bool Player::triggerEvent(Object* Objptr){
    Player* playerPtr=dynamic_cast<Player*>(Objptr);
    vector<Item> inv=getInventory();
        cout<<endl;
        cout<<"----------------------"<<endl;
        cout<<"I am "<<getName()<<"!"<<endl;
        cout<<"HP: "<<getCurrentHealth()<<"/"<<getMaxHealth()<<endl;
        cout<<"ATK: "<<getAttack()<<endl;
        cout<<"DEF: "<<getDefense()<<endl;
        cout<<endl;
        cout<<"Helmet:"<<endl;
        if(Helmet!=nullptr)
            cout<<getHelmet()->getName()<<endl;
        cout<<endl;
        cout<<"Chestplate:"<<endl;
        if(Chestplate!=nullptr)
            cout<<getChestplate()->getName()<<endl;
        cout<<endl;
        cout<<"LeftHand:"<<endl;
        if(LeftHand!=nullptr)
            cout<<getLeftHand()->getName()<<endl;
        cout<<endl;
        cout<<"RightHand:"<<endl;
        if(RightHand!=nullptr)
            cout<<getRightHand()->getName()<<endl;
        cout<<endl<<endl;



        cout<<"Items:"<<endl<<endl;
        for(int i=0;i<inv.size();i++){
            cout<<i+1<<". "<<inv[i].getName()<<endl;
        }
        cout<<"----------------------"<<endl;
        cout<<endl;
    return true;
}
```

4. Pick up Items

使用 Item::triggerEvent(Object* Objptr);函式實作，將 Objptr downcast 成 playerPtr 並用 playerPtr->addItem(*this)函式，將該 Item 加入到 player 的 inventory 中。

相關程式碼：

```
bool Item::triggerEvent(Object* ObjPtr){
    Player* playerPtr = static_cast<Player*>(ObjPtr);
    cout<<"\n*Hint: You got "<<this->getName()<<".*\n"<<endl;
    playerPtr->addItem(*this);
    return 0;
}

void Player::addItem(Item it){
    inventory.push_back(it);
}
```

5. Fighting System

使用 Monster::triggerEvent(Object* Objptr);函式實作，先將 Objptr downcast 成 playerPtr，用迴圈偵測怪物的 HP 是否大於零，若是則繼續執行 while 迴圈，迴圈中使用 player->checkIsDead()偵測玩家血量是否>0，若是則輸出死亡訊息並結束程式，若否則繼續戰鬥。

戰鬥中有三個選項：攻擊、撤退或打開背包。

攻擊時，玩家輸出傷害以(玩家攻擊力-怪物防禦力)來計算，並有四分之一的機率輸出

爆擊，能打出兩倍傷害（下述）。怪物輸出傷害則用(怪物攻擊力-玩家防禦力)來計算。另外如果傷害計算出來小於 0，則強制將其變為 0。以 takeDamage(damage);造成傷害。每次攻擊完會輸出怪物 HP、玩家 HP 以及玩家造成的傷害。怪物死亡後，將房間的 vector 清空。

撤退時，將玩家傳送到上一個房間，另外有宣告一個 int doublebreak;，因為沒有要將房間裡的 Monster 刪除，但若是原定流程在離開戰鬥的 while 迴圈時，Monster 即會遭到移除，因此定義 doublebreak，將其值設定為 1，接收到時則多 break;一次出迴圈。

打開背包時，戰鬥中只有治療藥水和傷害藥水可以使用，其他則輸出 Invalid choice，列出玩家所有的物品，若玩家選擇治療藥水，則加玩家的 HP，並檢查是否超過最高 HP，若是則將 HP 設定為最高 HP。若選擇傷害藥水，則對怪物造成傷害。

相關程式碼：

```
bool Monster::triggerEvent(Object* ObjPtr){
    int doublebreak = 0;
    Player* playerPtr = dynamic_cast<Player*>(ObjPtr);
    cout<<endl<<getName()<<" is ready to attack!"<<endl<<endl;
    while(getCurrentHealth()>0){
        if(playerPtr->checkIsDead()){
            cout<<"----------------------"<<endl;
            cout<<"My eyes begin to be blurred."<<endl;
            cout<<"I begin to feel exhausted."<<endl;
            cout<<"I am falling down."<<endl;
            cout<<"The world seems to be great, huh?"<<endl;
            cout<<"But now it matters no more."<<endl<<endl;
            cout<<"You're dead, try harder!"<<endl;
            cout<<"----------------------"<<endl;
            exit(0);
            return 1;
        }
        cout<<"What do I want to do?"<<endl;
        cout<<"1. Attack "<<getName()<<endl;
        cout<<"2. Retreat"<<endl;
        cout<<"3. Open my backpack"<<endl;
        int n;
        while(cin>>n){
            if(n==1){
                srand(time(NULL));
                int x=rand();
                int cri=0;
                x%=4;
                if(x==0){
                    cout<<endl<<"Critical Hit!"<<endl;
                    cri=1;
                }
                int damage=playerPtr->getAttack()-getDefense();
                if(damage<0){
                    damage=0;
                }
                if(cri==1){
                    damage*=2;
                }
                takeDamage(damage);
                cout<<endl<<getName()<<" has now "<<getCurrentHealth()<<" HP."<<endl;
                cout<<"My attack does "<<damage<<" damage."<<endl;
                int mondamage=getAttack()-playerPtr->getDefense();
                if(mondamage<0){
                    mondamage=0;
                }
                playerPtr->takeDamage(mondamage);
```

```
                cout<<"I have now "<<playerPtr->getCurrentHealth()<<" HP."<<endl<<endl;
                break;
            }
            else if(n==2){
                cout<<endl<<"Successively retreated!"<<endl<<endl;
                playerPtr->changeRoom(playerPtr->getPreviousRoom());
                doublebreak=1;
                break;
            }
            else if(n==3){
                cout<<"What do I want to use?"<<endl;
                playerPtr->showInventory();
                int number;
                cin>>number;
                number--;
                vector<Item> inv=playerPtr->getInventory();
                if(inv[number].getTag()=="HealingPotion"){
                    cout<<endl<<"I drink the healing potion."<<endl<<endl;
                    int ht=playerPtr->getCurrentHealth()+inv[number].getHealth();
                    if(ht>playerPtr->getMaxHealth()){
                        ht=playerPtr->getMaxHealth();
                    }
                    playerPtr->setCurrentHealth(ht);
                    for(int i=number;i<inv.size()-1;i++){
                        inv[i]=inv[i+1];
                    }
                    inv.pop_back();
                    playerPtr->setInventory(inv);
                }
                else if(inv[number].getTag()=="HurtingPotion"){
                    cout<<endl<<"I throw the hurting potion to the monster."<<endl<<endl;
                    takeDamage(inv[number].getHealth());
                    for(int i=number;i<inv.size()-1;i++){
                        inv[i]=inv[i+1];
                    }
                    inv.pop_back();
                    playerPtr->setInventory(inv);
                }
                else{
                    cout<<"I cannot use it now."<<endl;
                }
                break;
            }
        }
        if(doublebreak==1){
            break;
        }
    }
    if(doublebreak==0){
        cout<<getName()<<" is defeated!"<<endl<<endl;
        vector<Object*> vec;
        playerPtr->getCurrentRoom()->setObjects(vec);
    }

    return true;
}
void GameCharacter::takeDamage(int dmg){
    currentHealth-=dmg;
}
```

6. NPC

使用 NPC::triggerEvent(Object* objptr);函式實作，包含輸出台詞 cout<<getScript();以及交易系統，須從 NPC 手上拿走所有 Item，對每個 Item 使用 item.triggerEvent();，交易結束後將 NPC 從房間裡清除。

相關程式碼：

```
bool NPC::triggerEvent(Object* Objptr){
    Player* playerPtr=dynamic_cast<Player*>(Objptr);
    cout<<getScript();
    cout<<"\n\"You may take these.\""<<endl;
    listCommodity();
    vector<Item> com;
    com=getCommodity();

    int n;
    while(com.size()!=0){
        for(int i=0;i<com.size();i++){
            cout<<i+1<<". "<<com[i].getName()<<endl;
        }
        cin>>n;
        n--;
        if(n>=com.size()||n<0){
            continue;
        }
        com[n].triggerEvent(playerPtr);
        for(int k=n;k<com.size()-1;k++){
            com[k]=com[k+1];
        }
        com.pop_back();
    }
    vector<Object*> blank;
    blank.clear();
    playerPtr->getCurrentRoom()->setObjects(blank);
    return true;
}
```

7. GameLogic

包含勝利及失敗兩部分，每次動作以前都用 checkgamelogic()檢查。

若玩家所在房間的 isExit==true，則輸出勝利訊息並結束程式。

若玩家已死亡，則輸出死亡訊息並結束程式。

（這個情況發生在玩家和怪物同時死亡的時候）

另外在作戰系統中，也偵測玩家是否已死亡，若是則輸出死亡訊息並結束程式。

相關程式碼：

```
while(getCurrentHealth()>0){
    if(playerPtr->checkIsDead()){
        cout<<"----------------------"<<endl;
        cout<<"My eyes begin to be blurred."<<endl;
        cout<<"I begin to feel exhausted."<<endl;
        cout<<"I am falling down."<<endl;
        cout<<"The world seems to be great, huh?"<<endl;
        cout<<"But now it matters no more."<<endl<<endl;
        cout<<"You're dead, try harder!"<<endl;
        cout<<"----------------------"<<endl;
        exit(0);
        return 1;
    }
```

```
void Dungeon::runDungeon(){
    startGame();
    while(checkGameLogic()==0){
        vector<Object*> action=player.getCurrentRoom()->getObjects();
        chooseAction(action);
    }
}
```

```
int Dungeon::checkGameLogic(){
    if(player.checkIsDead()){
        cout<<"My eyes begin to be blurred."<<endl;
        cout<<"I begin to feel exhausted."<<endl;
        cout<<"I am falling down."<<endl;
        cout<<"The world seems to be great, huh?"<<endl;
        cout<<"But now it matters no more."<<endl<<endl;
        cout<<"You're dead, try harder!"<<endl;
        exit(0);
        return 1;
    }
    if(player.getCurrentRoom()->getIsExit()==true){
        cout<<endl;
        cout<<"------------------------"<<endl;
        cout<<"I walk out to the whole new world."<<endl;
        cout<<"Birds are singing, flowers are blooming."<<endl;
        cout<<"My journey seems to be stopped for a while."<<endl;
        cout<<"But my story would never end."<<endl<<endl;
        cout<<"You won! Congratulations!"<<endl;
        cout<<"------------------------"<<endl;
        exit(0);
        return 2;
    }
    else{
        return 0;
    }
}
```

8. Backpack System

背包系統，包含了平常使用的系統和在戰鬥中使用的系統。

平常使用的系統提供穿戴裝備的功能，以 openBackpack();實作，列出所有物品，讀取輸入，若選到裝備則穿戴裝備，四個部件各有不同的函數但功能相似，若選擇其他物品（藥水或鑰匙）則輸出 Invalid choice。

戰鬥中使用的系統則提供使用藥水的功能，若是傷害藥水對怪物造成傷害，若是治療藥水則補充玩家 HP，若選擇其他 Item 則輸出 I cannot use it now.。

相關程式碼：

```
void Player::openBackpack(){
    if(getInventory().size()==0){
        cout<<"\nThere's nothing in my backpack.\n"<<endl;
    }
    else{
        cout<<endl<<"What do I want to use?"<<endl;
        vector<Item> inv=getInventory();
        for(int i=0;i<inv.size();i++){
            cout<<i+1<<". "<<inv[i].getName()<<endl;
        }
        int chose;
        while(cin>>chose){
            if(chose>inv.size()){
                cout<<"Invalid choice."<<endl;
                break;
            }
            chose--;
            if(inv[chose].getTag()=="Helmet"){
                setHelmet(&inv[chose]);
                break;
            }
            else if(inv[chose].getTag()=="Chestplate"){
                setChestplate(&inv[chose]);
                break;
            }
            else if(inv[chose].getTag()=="LeftHand"){
                setLeftHand(&inv[chose]);
                break;
            }
            else if(inv[chose].getTag()=="RightHand"){
                setRightHand(&inv[chose]);
                break;
            }
            else{
                cout<<"Invalid choice."<<endl;
            }
        }
    }

}
```

```
else if(n==3){
    vector<Item> inv=playerPtr->getInventory();
    if(inv.size()==0){
        cout<<"\nThere's nothing in my backpack.\n"<<endl;
        break;
    }
    cout<<"What do I want to use?"<<endl;
    playerPtr->showInventory();
    int number;
    cin>>number;
    number--;

    if(inv[number].getTag()=="HealingPotion"){
        cout<<endl<<"I drink the healing potion."<<endl<<endl;
        int ht=playerPtr->getCurrentHealth()+inv[number].getHealth();
        if(ht>playerPtr->getMaxHealth()){
            ht=playerPtr->getMaxHealth();
        }
        playerPtr->setCurrentHealth(ht);
        for(int i=number;i<inv.size()-1;i++){
            inv[i]=inv[i+1];
        }
        inv.pop_back();
        playerPtr->setInventory(inv);
    }
    else if(inv[number].getTag()=="HurtingPotion"){
        cout<<endl<<"I throw the hurting potion to the monster."<<endl<<endl;
        takeDamage(inv[number].getHealth());
        for(int i=number;i<inv.size()-1;i++){
            inv[i]=inv[i+1];
        }
        inv.pop_back();
        playerPtr->setInventory(inv);
    }
    else{
        cout<<"I cannot use it now."<<endl;
    }
    break;
}
```

9. Optional Enhancement

爆擊系統

在戰鬥系統中出現，先在 monster.h 中#include <ctime>，接著在戰鬥系統的函式中設定亂數種子為時間，宣告 int x=rand();，接著 x%=4;，若 x==0 則輸出 Critical Hit!，並將玩家造成的傷害乘上 2 倍。

相關程式碼：

```
#ifndef ENEMY_H_INCLUDED
#define ENEMY_H_INCLUDED

#include <iostream>
#include <string>
#include <vector>
#include <ctime>
#include <cstdlib>
#include "GameCharacter.h"
#include "Player.h"
```

```
srand(time(NULL));
int x=rand();
int cri=0;
x%=4;
if(x==0){
    cout<<endl<<"Critical Hit!"<<endl;
    cri=1;
}
int damage=playerPtr->getAttack()-getDefense();
if(damage<0){
    damage=0;
}
if(cri==1){
    damage*=2;
}
takeDamage(damage);
```

II.Results

1. Actions Menu

```
What do I want to do?
1. Move.
2. Check Status.
3. Open my backpack.
4. Talk to Kirito.
```

```
What do I want to do?
1. Retreat.
2. Check Status.
3. Open my backpack.
4. Fight with Guardian.
```

2. Movement

```
Where do I want to go?
1. Go up.
2. Go down.
3. Go left.(Unable to go.)
4. Go right.(Unable to go.)
1
```

```
Where do I want to go?
1. Go up.(Requires a key.)
2. Go down.
3. Go left.(Unable to go.)
4. Go right.(Unable to go.)
1
You don't have the Key.
```

3. Showing Status

```
----------------------
I am Jeremy!
HP: 40/40
ATK: 6
DEF: 4

Helmet:
LeatherHelmet

Chestplate:
LeatherChestplate

LeftHand:
LeatherLeftGlove

RightHand:
Knife


Items:

1. LeatherRightGlove
----------------------
```

4. Pick up items

```
What do I want to do?
1. Move.
2. Check Status.
3. Open my backpack.
4. Open the Chest.
4

*Hint: You got TheKeytoAWholeNewWorld.*
```

5. Fighting System and Optional Enhancement(Critical Hit System)

```
What do I want to do?
1. Attack Slimy
2. Retreat
3. Open my backpack
1

Critical Hit!

Slimy has now -8 HP.
My attack does 12 damage.
I have now 40 HP.

Slimy is defeated!
```

6. NPC

```
I wake up in a dark room with only a lamp glowing dimly.
An old man sitting on a chair is looking at me with his eyes brimming in radiation.
"Uwo! Kimiwaatarashiiyuusyadesuka?
Mukashimukashi, oremokatsuteyuusyadeshitane."
I shake my head to tell him that I don't even know a single word.
"Naruhotone, daga isekaidewanihongowomanabanakya. Korewo tabette."
He gave you something, it seems like that I should eat that to get to know what he's talking about.
I eat the jelly, suddenly...
"Now you know what I am saying, huh?"
The world is now in danger, you should go save the world.
"Keep going, and then you will see a tree."
"In this journey, you might encounter some monsters."
"Here, take some equipments."
"Don't be afraid, good luck!"


"You may take these."
1. LeatherHelmet
2. LeatherChestplate
3. LeatherLeftGlove
4. LeatherRightGlove
5. Knife
1
```

7. GameLogic

```
Mizuria has now -3 HP.
My attack does 6 damage.
I have now -1 HP.

Mizuria is defeated!


-------------------------
My eyes begin to be blurred.
I begin to feel exhausted.
I am falling down.
The world seems to be great, huh?
But now it matters no more.

You're dead, try harder!
-------------------------
```

```
Where do I want to go?
1. Go up.(Requires a key.)
2. Go down.
3. Go left.(Unable to go.)
4. Go right.(Unable to go.)
1

-------------------------
I walk out to the whole new world.
Birds are singing, flowers are blooming.
My journey seems to be stopped for a while.
But my story would never end.

You won! Congratulations!
-------------------------
```

8. Backpack System

```
What do I want to use?
1. LeatherRightGlove
2. HealingPotion I
3. HurtingPotion I
4. LeatherHelmet
5. HealingPotion II
6. HurtingPotion II
7. LeatherChestplate
8. LeatherLeftGlove
1

*LeatherRightGlove is equipped.*
```

```
What do I want to do?
1. Attack Guardian
2. Retreat
3. Open my backpack
3
What do I want to use?
1. HealingPotion I
2. LeatherHelmet
3. HealingPotion II
4. HurtingPotion II
5. LeatherChestplate
6. LeatherLeftGlove
7. Knife
8. RyuRyu
9. HealingPotion II
10. HurtingPotion II
1

I drink the healing potion.
```

```
What do I want to do?
1. Attack Guardian
2. Retreat
3. Open my backpack
3
What do I want to use?
1. HealingPotion I
2. HurtingPotion I
3. LeatherHelmet
4. HealingPotion II
5. HurtingPotion II
6. LeatherChestplate
7. LeatherLeftGlove
8. Knife
9. RyuRyu
10. HealingPotion II
11. HurtingPotion II
2

I throw the hurting potion to the monster.
```

III.Discussion

1.Problems that I Met

遇到的問題不少。

第一是在函式宣告完所有房間和所有的角色後，有出現房間連結關係正確但　裡面角色全錯的狀況，以及 downcast 後仍然會呼叫到 pure virtual function 的狀況，　討論後

才知道我把房間裡面東西的指標留在了 createMap();裡面，因而出了函式後　指標就壞掉了，後來用 new 解決了這個問題。如圖所示：

```
Item leatherhelmet("LeatherHelmet","Helmet",0,1,1);
Item* LeatherHelmet=new Item(leatherhelmet);
Item leatherchestplate("LeatherChestplate","Chestplate", 0, 1, 1);
Item* LeatherChestplate=new Item(leatherchestplate);
Item leatherleftglove("LeatherLeftGlove","LeftHand", 0, 1, 1);
Item* LeatherLeftGlove=new Item(leatherleftglove);
Item leatherrightglove("LeatherRightGlove","RightHand", 0, 1, 1);
Item* LeatherRightGlove=new Item(leatherrightglove);
Item naturehelmet("NatureHelmet","Helmet", 1, 2, 6);
Item* NatureHelmet=new Item(naturehelmet);
Item naturechestplate("NatureChestplate","Chestplate", 1, 2, 6);
Item* NatureChestplate=new Item(naturechestplate);
Item knife("Knife","RightHand", 0, 2, 0);
Item* Knife=new Item(knife);
Item ryuryu("RyuRyu","RightHand",5, 15, 1);
Item* RyuRyu=new Item(ryuryu);
Item tenkafubu("Tenkafubu","LeftHand",5, 15, 1);
Item* Tenkafubu=new Item(tenkafubu);
Item healingpotion_i("HealingPotion I","HealingPotion", 15, 0, 0);
Item* HealingPotion_I=new Item(healingpotion_i);
Item healingpotion_ii("HealingPotion II","HealingPotion", 30, 0, 0);
Item* HealingPotion_II=new Item(healingpotion_ii);
Item hurtingpotion_i("HurtingPotion I","HurtingPotion", 15, 0, 0);
Item* HurtingPotion_I= new Item(hurtingpotion_i);
Item hurtingpotion_ii("HurtingPotion II","HurtingPotion", 30, 0, 0);
Item* HurtingPotion_II=new Item(hurtingpotion_ii);
Item thekeytofinal("TheKeytoAWholeNewWorld","Key", 0, 0, 0);
Item* TheKeytoFinal=new Item(thekeytofinal);

vector<Object*> firstobj;
NPC* KiritoPtr=new NPC(Kirito);
firstobj.push_back(KiritoPtr);
Room firstroom(false, 0, firstobj);
Room* FirstRoom=new Room(firstroom);

vector<Object*> secondobj;
Monster* SlimyPtr=new Monster(Slimy);
secondobj.push_back(SlimyPtr);
Room secondroom(false, 1, secondobj);
Room* SecondRoom=new Room(secondroom);

vector<Object*> thirdobj;
Monster* GoblyPtr=new Monster(Gobly);
thirdobj.push_back(GoblyPtr);
Room thirdroom(false, 2, thirdobj);
Room* ThirdRoom=new Room(thirdroom);

vector<Object*> fourthobj;
Monster* ZombyPtr=new Monster(Zomby);
fourthobj.push_back(ZombyPtr);
Room fourthroom(false, 3, fourthobj);
Room* FourthRoom=new Room(fourthroom);

vector<Object*> treeobj;
NPC* TreeSpiritPtr=new NPC(TreeSpirit);
treeobj.push_back(TreeSpiritPtr);
Room theroomwithatree(false, 4, treeobj);
Room* TheRoomWithATree=new Room(theroomwithatree);

vector<Object*> leftfirstobj;
Monster* FlamiaPtr=new Monster(Flamia);
leftfirstobj.push_back(FlamiaPtr);
Room leftfirstroom(false, 5, leftfirstobj);
Room* LeftFirstRoom=new Room(leftfirstroom);

vector<Object*> leftsecondobj;
Monster* BlaziaPtr=new Monster(Blazia);
leftsecondobj.push_back(BlaziaPtr);
Room leftsecondroom(false, 6, leftsecondobj);
Room* LeftSecondRoom=new Room(leftsecondroom);
```

第二是對 static_cast 和 dynamic_cast 的運作方式不熟悉，在上面情況裡用 downcast 的時候用了 static_cast 有時結果會對，殊不知其實指標早就壞掉，因此會　亂跳，而使用 dynamic_cast 會直接當掉。後來詢問同學才弄懂原來 dynamic_cast

如　果 downcast 失敗會回傳 nullptr，因此在對 nullptr 求裡面的函式結果時會直接當掉。　　這樣以後就知道他們的特性了。

第三是關於 Multiple Files 的特性不夠了解，想著在其他檔案裡面 include 一些標　　頭檔，結果一 include 馬上報出一堆錯誤，後來才用不需要額外 include 的方式解　　決。

另外我在剛開始這個作業的時候，都憑著感覺在寫，以至於後來要運作的時　候直接爆出五十個錯誤。很多都是心不在焉而犯下的失誤，以後就知道要時刻追　　蹤運行的狀況，才不會耗時又做白工。

2. Things that I can Improve in the Future

這次因為平常時間分配不當，以至於到 deadline 前五天才開始趕這個作業，結　　果雖然盡了全力趕卻還是遲交，也沒有時間把 savefile 做出來，是很可惜的事情，　希望以後在時間規畫上能夠更加妥當，讓自己不至於陷入這樣的窘境。

IV.Conclusion

這次做這個作業算是我在程式上完成過最大的一個 Project，是一個難得的經驗，也象徵著自己即將要開始走向程式設計中更深的領域了。然而這次作業因為對於物件導向以及 Multiple Files 的特性仍然不構嫻熟，因而遭遇了許多阻礙，也導致自己沒能完成許多自己想做的東西，相當可惜，希望有了這次的經驗，我能在未來程式設計的路上走得更加順遂。