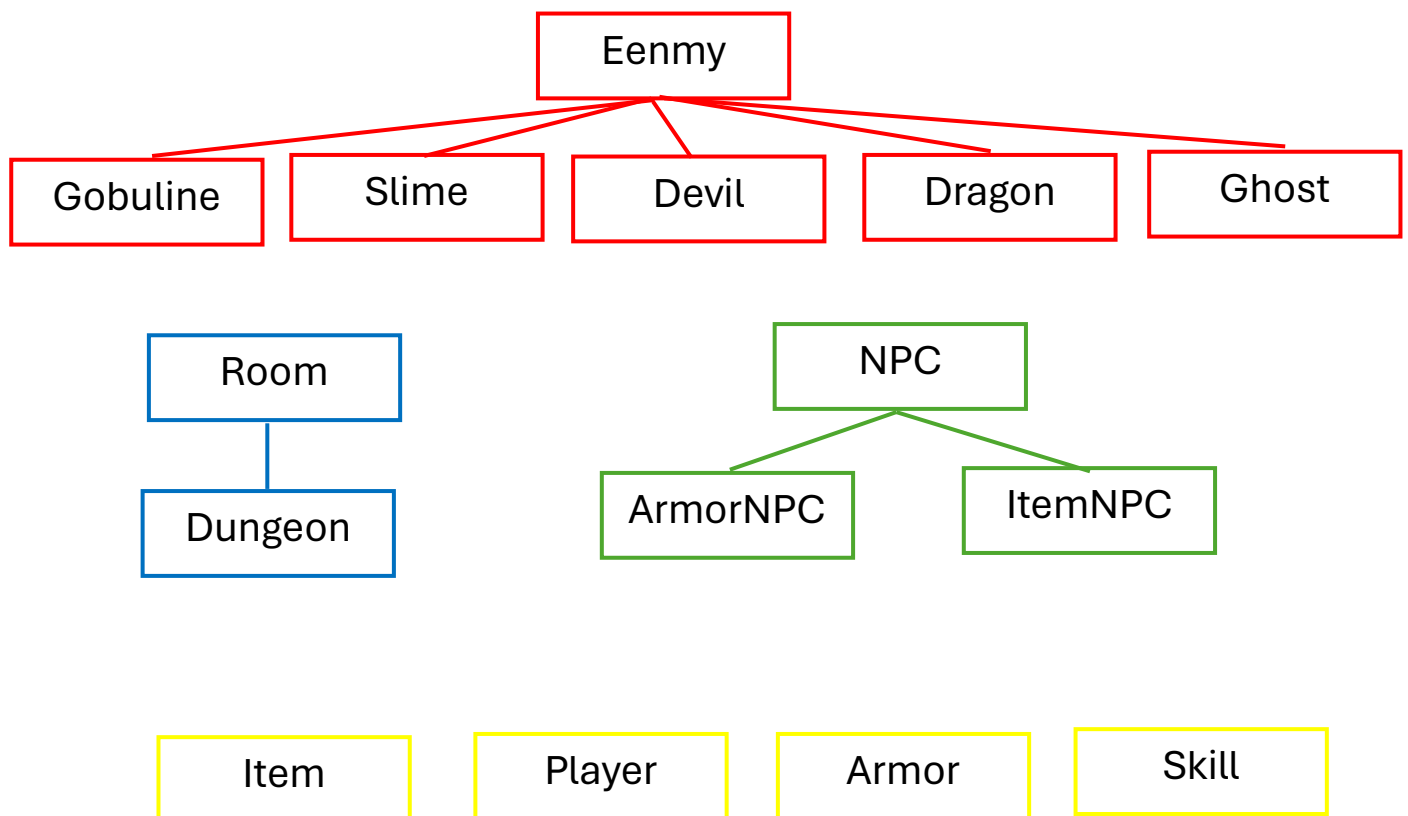


Dungeon report

一、程式基本架構與原理簡介



(繼承關係圖)

以下簡介一些由遊戲運行時的重要 Function(在 class Dungeon 內)

<code>void startgame()</code>	開始遊戲
<code>void mainscence(Player* player)</code>	控制腳色移動與休息介面的操控
<code>void death(Player* player);</code>	判斷腳色是否死亡
<code>void showitem(Player* player);</code>	顯示背包裡的道具
<code>void showarmor(Player* player);</code>	顯示身上的裝備
<code>void showskill(Player* player);</code>	顯示技能

void statesystem(Player* player); 控制腳色的飢餓、水分、中毒狀態

void Fight(Player* player, Enemy* enemy); 戰鬥系統

void shopsystem(NPC* shopkeeper, Player* player); 商店系統

void procedure(Player* player); 遊戲每回合的流程控制

void bossfight(Player* player); 特殊制定的 BOSS 戰

接下來我會介紹遊戲的運行流程流程

首先我們會啟動遊戲

void startgame()	開始遊戲
------------------	------

接下來我們可以把每個遊戲分成好幾回合

void procedure(Player* player);	遊戲每回合的流程控制
---------------------------------	------------

第一階段（觸發事件）：

當腳色進到房間裡後會先觸發房間內的物件

1.Enemy 2.NPC 3.空房間

Case1.

當遇到 Enemy 時會觸發戰鬥系統，玩家可以選擇普通攻擊、技能一、技能二、使用道具和逃跑(BOSS 戰不能逃跑)，選擇一動作後便會被怪物攻擊(逃跑成功除外)，接下來重複選擇攻擊...直到怪物或玩家死亡。

void Fight(Player* player, Enemy* enemy);	戰鬥系統
---	------

註:逃跑成功後會回到上一個房間

Case2.

當遇到 NPC 時會觸發商店系統，你有可能會遇到防具商人或道具商人，你可以向他們購買物品(可重複購買)

void shopsystem(NPC* shopkeeper, Player* player);	商店系統
---	------

Case3.

當遇到空房間時代表沒有遇到任何商人或怪物

第二階段(探索)：

觸發事件完後，你會在這個房間內探索，每個房間有不同地形，因此會遇到每個地形特有的事件。

<code>void procedure(Player* player);</code>	遊戲每回合的流程控制
--	------------

第三階段(選擇方向)：

選擇你要前往哪個房間，並可以在這個階段查看腳色各種資料

<code>void mainscence(Player* player)</code>	控制腳色移動與休息介面的操控
--	----------------

最終到 BOSS 房進行最終的戰鬥

<code>void bossfight(Player* player);</code>	特殊制定的 BOSS 戰
--	--------------

二、各項要求的實作介紹

註:有些太長的程式碼因空間關係因此不會放進說明

Actions Menu



介面經過美化能讓玩家更了解遊戲運行流程，不用因太多文字而感到疲憊

```

void map(Player* player)
{
    gotoxy(0, 0);    cout << " ";
    gotoxy(0, 1);    cout << " | Lev." << player->getLevel() << " " << "name:" << player->getName() << " ";
    gotoxy(0, 2);    cout << " | ";
    gotoxy(0, 3);    cout << " | ";
    gotoxy(0, 4);    cout << " | ";
    gotoxy(0, 5);    cout << " | ";
    gotoxy(0, 6);    cout << " | ";
    gotoxy(0, 7);    cout << " | ";
    gotoxy(0, 8);    cout << " | ";
    gotoxy(0, 9);    cout << " | ";
    gotoxy(0, 10);   cout << " | ";
    gotoxy(0, 11);   cout << " | ";
    gotoxy(0, 12);   cout << " | ";
    gotoxy(0, 13);   cout << " | ";
    gotoxy(0, 14);   cout << " | ";
    gotoxy(0, 15);   cout << " | ";
    gotoxy(0, 16);   cout << " | ";
    gotoxy(0, 17);   cout << " | ";
    gotoxy(0, 18);   cout << " | ";
    gotoxy(0, 19);   cout << " | ";
    gotoxy(0, 20);   cout << " | ";
    gotoxy(0, 21);   cout << " | EXP:";
    SetColor(170);
    for (int h = 0; h < 20; h++)
    {
        if (h >= (float)player->getnowEXP() / ((float)player->getEXP() / 20))
        {
            SetColor();
        }
        cout << " ";
    }
    SetColor();
    cout << " " << player->getnowEXP() << "/" << player->getEXP();
    gotoxy(82, 21);
    cout << " | HP:";
    //gotoxy(0, 21);    cout << " | ";
    SetColor(204);
    for (int h = 0; h < 20; h++)
    {
        if (h >= (float)player->getnowHP() / ((float)player->getHP() / 20))
        {
            SetColor();
        }
        cout << " ";
    }
    SetColor();
    cout << " " << setw(3) << setfill(' ') << player->getnowHP() << "/" << setw(3) << setfill(' ') << player->getHP();
    gotoxy(119, 21);
    cout << " | ";
    gotoxy(0, 22);    cout << " | ";
    gotoxy(40, 22);   cout << " | 技能:" << player->getownskill().getname();
}

```

```

gotoxy(40, 22);    cout << " | 技能:" << player->getownskill().getname();
gotoxy(82, 22);   cout << " | MP:";
SetColor(17);
for (int h = 0; h < 20; h++)
{
    if (h >= (float)player->getnowMP() / ((float)player->getMP() / 20))
    {
        SetColor();
    }
    cout << " ";
}
SetColor();
cout << " " << setw(3) << setfill(' ') << player->getnowMP() << "/" << setw(3) << setfill(' ') << player->getMP();
gotoxy(119, 22);
cout << " | ";
gotoxy(0, 23);    cout << " | ";
gotoxy(0, 24);    cout << " | 裝備(請按e鍵)";
gotoxy(40, 24);   cout << " | 技能一:" << player->getskill1().getname();
gotoxy(82, 24);   cout << " | ";
cout << " | "; cout << " | ATK: " << player->getATK(); gotoxy(101, 24); cout << "DEF : " << player->getDEF(); gotoxy(119, 24); cout << " | ";
gotoxy(0, 25);    cout << " | ";
gotoxy(0, 26);    cout << " | 道具(請按h鍵)";
gotoxy(40, 26);   cout << " | 技能二:" << player->getskill2().getname();
gotoxy(82, 26);   cout << " | ";
cout << " | "; cout << " | LUK: " << player->getLUK(); gotoxy(101, 26); cout << "AGI: " << player->getAGI(); gotoxy(119, 26); cout << " | ";
gotoxy(0, 27);    cout << " | ";
gotoxy(0, 28);    cout << " | (檢視技能請按s鍵) | ";
gotoxy(62, 22);   cout << " | ";
cout << " | 飽食度:" << setw(3) << setfill('0') << player->getthunger() << "/100";
gotoxy(62, 24);   cout << " | ";
cout << " | 水分:" << setw(3) << setfill('0') << player->getthirsty() << "/100";
gotoxy(62, 26);   cout << " | ";
cout << " | 狀態:";
if (player->getpoison() == 1)
{
    cout << "中毒";
}
else if (player->getpoison() == 2)
{
    cout << "失智";
}
else if (player->getpoison() == 3)
{
    cout << "中毒+失智";
}
else if (player->getpoison() == 0)
{
    cout << "無";
}
}

```

1. Movement (10%)

如 Action Menu 的圖，可以前進的方向會直接顯示在正中心，按下 l、u、d 或 r 後便可前往指定房間，遇到牆壁時不能走的方向便不會顯示箭頭。

```
while (st != 'u' && st != 'd' && st != 'r' && st != 'l')
{
    map(player);
    movementmap(player);
    st = _getche();
    //map(player1);
    if (st == 'h') { ... }
    else if (st == 'g') { ... }
    else if (st == 's')
    {
        showskill(player);
        gotoxy(0, 14); cout << "輸入e離開此頁面";
        st = _getche();
        while (st != 'e')
        {
            system("cls");
            showskill(player);
            gotoxy(0, 14); cout << "輸入e離開此頁面";
            st = _getche();
        }
    }
    else if (st == 'u')
    {
        if (player->getx() == 0)
        {
            system("cls");
            gotoxy(0, 15); cout << "不能往上走";
            st = 'A';
            Sleep(1000);
        }
        else
        {
            statesystem(player);
            death(player);
            player->setlocation(-1, 0);
        }
    }
}
```

這裡我們會利用這個 void mainscence(Player* player)，藉由傳進去 player 的 x、y 判斷有沒有牆壁阻擋，並印出不同的箭頭圖示。

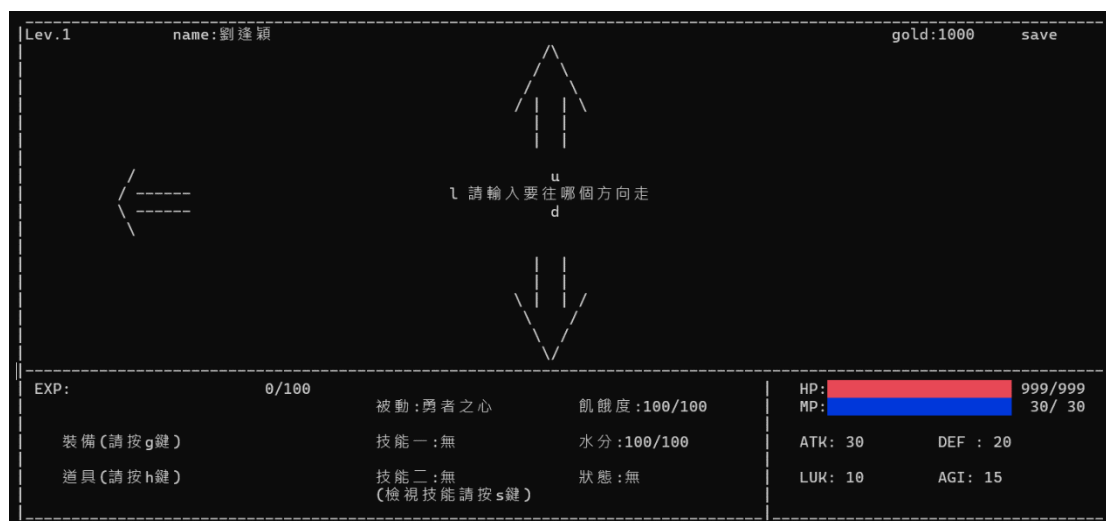
當選擇一方向移動時玩家的 x、y 質便會改變，並判斷下一個房間你會遇到什麼怪物、NPC 或地形事件



Showing Status(5%)

血量、魔力、攻擊、防禦、暴擊率、閃避率、飽食度、水分、狀態、名字、持有金錢和等級會藉由 `void map(Player* player)` 顯示出來。

其他的玩家資訊也是透過 `dungeon` 裡的 `function` 來顯示玩家現在的狀態



按下 `g` 可以顯示裝備，按下 `e` 便會退出裝備介面

```
void showarmor(Player* player);
```

初心者頭盔		頭部
普通的安全帽		
初心者盔甲		軀幹
用紙箱做的盔甲		
初心者鞋子		鞋子
穿了一年的鞋子		
輸入e離開此頁面		

```
void showarmor(Player* player)
{
    gotoxy(50, 4);    cout << " ";
    gotoxy(50, 5);    cout << " ";
    gotoxy(50, 6);    cout << " |" << player->armor[0].getname(); gotoxy(67, 6); cout << "| " << player->armor[0].getdetail(); gotoxy(95, 6); cout << "|";
    gotoxy(50, 7);    cout << " |-----| ";
    gotoxy(50, 8);    cout << " |" << player->armor[0].getdetail(); gotoxy(95, 8); cout << "| ";
    gotoxy(50, 9);    cout << " |-----| ";
    gotoxy(50, 10);   cout << " |" << player->armor[1].getname(); gotoxy(67, 10); cout << "| " << player->armor[1].getdetail(); gotoxy(95, 10); cout << "|";
    gotoxy(50, 11);   cout << " |-----| ";
    gotoxy(50, 12);   cout << " |" << player->armor[1].getdetail(); gotoxy(95, 12); cout << "| ";
    gotoxy(50, 13);   cout << " |-----| ";
    gotoxy(50, 14);   cout << " |" << player->armor[2].getname(); gotoxy(67, 14); cout << "| " << player->armor[2].getdetail(); gotoxy(95, 14); cout << "|";
    gotoxy(50, 15);   cout << " |-----| ";
    gotoxy(50, 16);   cout << " |" << player->armor[2].getdetail(); gotoxy(95, 16); cout << "| ";
    gotoxy(50, 17);   cout << " |-----| ";
    gotoxy(50, 18);   cout << " |-----| ";
}
```

按下 h 可以查看現有的道具，還可以進一步查看道具的詳細資料

```
void showitem(Player* player);
```

[illegible]

```
void Item::getdetail()
```

```
交大學餐
```

```
雖然不難吃但也沒也好吃到哪裡去，吃了可以回復飢餓值跟水分
```

```
void showitem(Player* player)
{
    gotoxy(30, 3);    cout << "
    gotoxy(30, 4);    cout << " |-----| ";
    gotoxy(30, 5);    cout << " |";
    gotoxy(30, 6);    cout << " |";
    gotoxy(30, 7);    cout << " |";
    gotoxy(30, 8);    cout << " |";
    gotoxy(30, 9);    cout << " |";
    gotoxy(30, 10);   cout << " |";
    gotoxy(30, 11);   cout << " |";
    gotoxy(30, 12);   cout << " |";
    gotoxy(30, 13);   cout << " |";
    gotoxy(30, 14);   cout << " |";
    gotoxy(30, 15);   cout << " |";
    gotoxy(30, 16);   cout << " |";
    gotoxy(30, 17);   cout << " |";
    gotoxy(30, 18);   cout << " |";
    gotoxy(30, 19);   cout << " |";
    gotoxy(30, 20);   cout << " |";
    gotoxy(30, 21);   cout << " |";
    gotoxy(30, 22);   cout << " |";
    gotoxy(30, 23);   cout << " |";
    gotoxy(30, 24);   cout << " |-----| ";
    gotoxy(30, 25);   cout << " ";
    for (int i = 0; i < player->item.size(); i++)
    {
        gotoxy(33, 2 * i + 5); cout << "[" << i << "]" << player->item[i].getname() << "輸入" << i << "取得更詳細的資料"; gotoxy(80, 2 * i + 5);
        gotoxy(30, 2 * i + 6); cout << " |-----| ";
    }
}
```

按下 s 可以查看已學習的技能

```
void showskill(Player* player);
```

```
被動：勇者之心
勇敢的你得到了作者的電話0970228136
```

```
技能一：無
無
```

```
技能二：無
無
```

```
void showskill(Player* player)
{
    system("cls");
    cout << "被動：" << player->getownskill().getname() << endl;
    cout << player->getownskill().getdepict() << endl;
    cout << "-----" << endl;
    cout << "技能一：" << player->getskill1().getname() << endl;
    cout << player->getskill1().getdepict() << endl;
    cout << "-----" << endl;
    cout << "技能二：" << player->getskill2().getname() << endl;
    cout << player->getskill2().getdepict() << endl;
}
```


Pick up Items(10%)

擊殺怪物時會掉落物品玩家會自動撿起，商店購買物品便會裝備上去或放進道具欄(這應該也算 Pick up Items 吧)。

掉落物會藉由 push_back 進入玩家的 item 裡

你成功擊敗Slime了
獲得50經驗、50金幣 戰利品:史萊姆黏液

```
void enemygood(Player* player, Enemy* enemy)
{
    srand(time(NULL));
    int randomnb;
    if (enemy->getname() == "Gobulin")
    {
        cout << "獲得60經驗、100金幣  ";
        player->setnowEXP(60);
        player->setgold(100);
        randomnb = rand();
        if (randomnb % 100 <= 50)
        {
            cout << "戰利品:哥布林水袋      ";
            bool haveornot = 0;
            for (int i = 0; i < player->item.size(); i++)
            {
                if (player->item[i].getname() == "哥布林水袋")
                {
                    player->item[i].setstack(1);
                    haveornot = 1;
                }
            }
            if (!haveornot)
            {
                player->item.push_back(allItem[0]);
                player->item[player->item.size() - 1].setstack(1);
            }
        }
        else
        {
            cout << "戰利品:無";
        }
    }
}
```

商店購買物品也是，購買成功後也會藉由 push_back 進入玩家的 item 裡，這裡要

惡魔獠牙	可愛小蘿莉的內褲	交大學餐
輸入1購買 花費:100	輸入2購買 花費:500	輸入3購買 花費:080
購買成功，歡迎再來		

注意到道具是可以疊加的可是裝備不行，而且一旦買了新裝備後舊裝備就會直接被丟棄

```
else if (shopkeeper->getindex() == 2)
{
    if (purchase == '1')
    {
        if (player->getGold() < shopkeeper->getshopitem()[0].getcost())
        {
            gotoxy(0, 27); cout << " |你的錢錢好像不夠          ";
            Sleep(2000);
        }
        else
        {
            bool haveornot = 0;
            for (int i = 0; i < player->item.size(); i++)
            {
                if (player->item[i].getname() == shopkeeper->getshopitem()[0].getname())
                {
                    player->item[i].setstack(1);
                    haveornot = 1;
                }
            }
            if (!haveornot)
            {
                player->item.push_back(shopkeeper->getshopitem()[0]);
                player->item[player->item.size() - 1].setstack(1);
            }
            player->setgold(-(shopkeeper->getshopitem()[0].getcost()));
            shopsceance(shopkeeper, player);
            gotoxy(0, 27); cout << " |購買成功，歡迎再來          ";
            mciSendString(TEXT("play 商店購買音效.wav"), NULL, 0, NULL);
            Sleep(2000);
        }
    }
}
```

Fighting System(10%)

簡潔的戰鬥畫面，可使用攻擊、技能、道具或逃跑，與怪物的戰鬥資訊會顯示在左下的方框中(會連續撥放並消除先前戰鬥紀錄)。

Void Dungeon::Fight(Player* player, Enemy* enemy); 傳入玩家與怪物的狀態

這裡我利用 while 判斷玩家和腳色的血量，只要其中一方血量小於 0 便會跳出迴圈，不然就是一直持續戰鬥

玩家的血量、魔力和怪物的血量會在雙方進行動作時做即時的變化。玩家在擊敗怪物後可獲得金幣、經驗值，有機率會有掉落物。

```
-----
獲得 500 經驗、1000 金幣   戰利品: 龍肉
|
你成功擊敗 Dragon 了
|
ATK(請輸入 a)   S1(請輸入 1)   S2(請輸入 2)   Item(使用請按 h)   逃跑請輸入 r
|
-----
```

獲得經驗值、金幣與
掉落物

你使用綠寶石噴射對敵人造成了136.2點暴擊傷害
你成功擊敗Dragon了
獲得500經驗、1000金幣 戰利品:龍肉

使用技能

你使用普通攻擊對敵人造成了99.75點暴擊傷害
你成功擊敗Dragon了
獲得500經驗、1000金幣 戰利品:龍肉

ATK(請輸入a) S1(請輸入1) S2(請輸入2) Item(使用請按h) 逃跑請輸入r

使用普通
攻擊

成功使用可愛小蘿莉的內褲
你滑溜的身法躲掉了敵人的攻擊

ATK(請輸入a) S1(請輸入1) S2(請輸入2) Item(使用請按h) 逃跑請輸入r

使用道具

逃跑成功

ATK(請輸入a) S1(請輸入1) S2(請輸入2)

逃跑

```
void Fight(Player* player, Enemy* enemy)
{
    srand(time(NULL));
    int randomnb;
    Fightscene(player, enemy);
    char action;
    while (player->getnowHP() > 0 && enemy->getnowHP() > 0)
    {
        gotoxy(4, 25);
        action = _getche();
        gotoxy(4, 25);
        cout << "          ";

        if (action == 'a')
        {
            if (*linepoint == 25)
            {
                gotoxy(0, 21); cout << " |          ";
                gotoxy(0, 22); cout << " |          ";
                gotoxy(0, 23); cout << " |          ";
                //gotoxy(0, 24); cout << " |          ";
                *linepoint = 21;
            }
            gotoxy(4, *linepoint); cout << "你使用普通攻擊對敵人造成了";
            if (player->getATK() - enemy->getDEF() <= 0)
            {
                cout << "1點傷害          ";
                enemy->getdamage(1);
            }
        }
    }
}
```

```

else
{
    randomb = rand();
    if (randomb % 100 <= player->getLUK())
    {
        cout << player->getATK() * 1.35 - enemy->getDEF() << "點暴擊傷害";
        enemy->getdamage(player->getATK() * 1.35 - enemy->getDEF());
    }
    else
    {
        cout << player->getATK() - enemy->getDEF() << "點傷害";
        enemy->getdamage(player->getATK() - enemy->getDEF());
    }
}
linedetect(linepoint);
gotoxy(0, 2); cout << " |"; cout << enemy->getname() << " HP:" << setw(3) << setfill(' ') << enemy->getnowHP() <
if (enemy->getnowHP() <= 0)
{
    if (*linepoint == 25)
    {
        gotoxy(0, 21); cout << " |";
        gotoxy(0, 22); cout << " |";
        gotoxy(0, 23); cout << " |";
        //gotoxy(0, 24); cout << " |";
        *linepoint = 21;
    }
    gotoxy(4, *linepoint); cout << "你成功擊敗" << enemy->getname() << "了";
    linedetect(linepoint);
    if (*linepoint == 25)
    {
        gotoxy(0, 21); cout << " |";
        gotoxy(0, 22); cout << " |";
        gotoxy(0, 23); cout << " |";
    }
}

```

```

        *linepoint = 21;
    }
    gotoxy(4, *linepoint);
    enemygood(player, enemy);
    *linepoint = 21;
}
}
else if (action == '1') { ... }
else if (action == '2') { ... }
else if (action == 'h') { ... }
else if (action == 'r') { ... }
else { ... }
Sleep(1000);
LevelUp(player);
if (enemy->getnowHP() > 0) { ... }
death(player);
}

```

```

enemy->setnowHP();
Sleep(3000);
}

```

NPC (10%)

Dungeon 裡一共有兩種 NPC，一個賣道具一個賣裝備，這兩種 NPC 都繼承自 NPC，藉由

```
virtual vector<Armor>& getshopArmor() = 0;
```

```
virtual vector<Item>& getshopitem() = 0;
```

來判斷賣的物品，而且可以重複購買直到輸入 0 離開，如果金幣不夠也會顯示

“金幣不夠無法購買”

```

      惡魔獠牙          可愛小蘿莉的內褲          交大學餐
    |-----|          |-----|          |-----|
    |輸入1購買 花費:100|  |輸入2購買 花費:500|  |輸入3購買 花費:080|
    |-----|          |-----|          |-----|
購買成功，歡迎再來    |
  
```

```

      惡魔獠牙          可愛小蘿莉的內褲          交大學餐
    |-----|          |-----|          |-----|
    |輸入1購買 花費:100|  |輸入2購買 花費:500|  |輸入3購買 花費:080|
    |-----|          |-----|          |-----|
你的錢錢好像不夠    |          如果不購買請輸入 0
  
```

```

class armorNPC :public NPC {
public:
    armorNPC(string name, Armor a1, Armor a2, Armor a3) : NPC(name, 1) {
        shopArmor.push_back(a1);
        shopArmor.push_back(a2);
        shopArmor.push_back(a3);
    }
    vector<Armor>& getshopArmor() override {
        return shopArmor;
    }
    vector<Item>& getshopitem() override {
        return shopitem;
    }
};

vector<armorNPC> AllarmorNPC = { armorNPC("翠須",headArmor[1],bodyArmor[1],feetArmor[1]),armorN

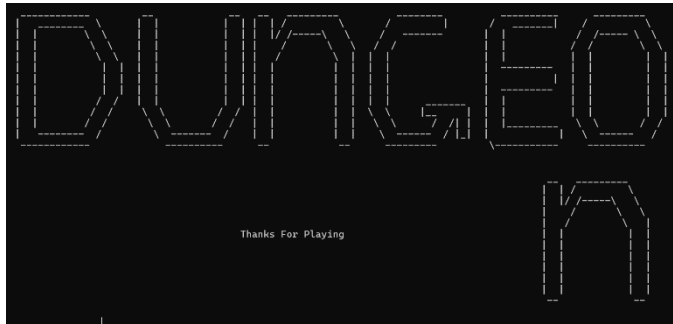
class ItemNPC :public NPC {
public:
    ItemNPC(string name, Item I1, Item I2, Item I3) : NPC(name, 2) {
        shopitem.push_back(I1);
        shopitem.push_back(I2);
        shopitem.push_back(I3);
    }
    vector<Item>& getshopitem() override {
        return shopitem;
    }
    vector<Armor>& getshopArmor() override {
        return shopArmor;
    }
};
  
```

Logic(10%)

我接下來會把遊戲邏輯分成 WIN 和 LOSE 來討論

1. WIN

當 BOSS 被擊敗時遊戲會進行到勝利結尾的部分，這時才算成功通關。



2. LOSE

當玩家在戰鬥中被擊敗亦或是被飽食度、水分和中毒的原因扣血導致玩家生命值低於 0 時即會被判定 LOSE，此部分是由

Void Dungeon::death(Player* player)

來判定。

```
void death(Player* player)
{
    if (player->getnowHP() <= 0)
    {
        PlaySound(TEXT("死亡音效.wav"), NULL, SND_FILENAME | SND_ASYNC);
        system("cls");
        gotoxy(0, 15);    cout << "                視線慢慢的模糊，你的生命到了盡頭永遠的沉眠在地下城中" << endl;
        Sleep(10000);
        exit(0);
    }
}
```

視線慢慢的模糊，你的生命到了盡頭永遠的沉眠在地下城中

Hunger System Design (7.5%)

因為 Hunger System Design 包含三種狀態因此我會個別敘述我如何實行這個系統。

Hunger:

玩家初始的飽食度為 100，每經過一回合飽食度便會被扣 20，在森林地形會扣 30，當遇到沙塵暴時飽食度還會被額外扣 10。

玩家可由 3 種方式補充飽食度

1. .使用龍肉 (怪物掉落物)
2. .使用交大學餐(NPC 購買)
3. 在森林地形遇到兔子(peko)，捕捉後即可補充 10 飽食度

當飽食度到 0 時便不會繼續往下扣，取而代之的是每經一回合玩家將會減 5 點生命。

```
void hungersystem(Player* player)
{
    if (dungen[player->getx()][player->gety()].getEN() == 1)
    {
        if (player->gethunger() > 30)
        {
            player->sethunger(-30);
        }
        else
        {
            player->sethunger(-30);
            player->setnowHP(-5);
        }
    }
    else
    {
        if (player->gethunger() > 20)
        {
            player->sethunger(-20);
        }
        else
        {
            player->sethunger(-20);
            player->setnowHP(-5);
        }
    }
}
```

Thirst:

玩家初始的水分為 100，每經過一回合水分便會被扣 20，在沙漠地形會扣 30，當遇到沙塵暴時飽食度還會被額外扣 10。

我們有 4 種方式補充水分

1. 使用交大學餐(NPC 購買)
2. 使用哥布林水袋(怪物掉落物)
3. 在沙漠遇到綠洲
4. 在森林遇到水源

當水分到 0 時便不會繼續往下扣，取而代之的是每經一回合玩家將會減 5 點生命和 1 點魔力。

```
void thirstsystem(Player* player)
{
    if (dungen[player->getx()][player->gety()].getEN() == 2)
    {
        if (player->getthirsty() > 30)
        {
            player->setthirsty(-30);
        }
        else
        {
            player->setthirsty(-30);
            player->setnowHP(-5);
            player->setnowMP(-1);
        }
    }
    else
    {
        if (player->getthirsty() > 20)
        {
            player->setthirsty(-20);
        }
        else
        {
            player->setthirsty(-20);
            player->setnowHP(-5);
            player->setnowMP(-1);
        }
    }
}
```


Poison:

遊戲中的中毒狀態有兩種


1. 中毒

當在沼澤地形遇到毒氣沼澤時便會中毒，在中毒的狀態每經過一回合扣 5 生命

2. 失智

當被幽靈攻擊時有機率會得失智，在失智的狀態下每經過一回合扣 2 點魔力

兩種狀態可以疊加，而且只有當使用交大學餐(附牛奶，不要問我為什麼交大學餐會有牛奶)時才能解毒。

EXP: 	20/160	被動:勇者之心	飽食度:010/100
裝備(請按g鍵)		技能一:無	水分:050/100
道具(請按h鍵)		技能二:無 (檢視技能請按s鍵)	狀態:中毒

```

void poisonsystem(Player* player)
{
    if (player->getpoison() == 1)
    {
        player->setnowHP(-5);
    }
    else if (player->getpoison() == 2)
    {
        player->setnowMP(-2);
    }
    else if (player->getpoison() == 3)
    {
        player->setnowHP(-5);
        player->setnowMP(-2);
    }
}

```

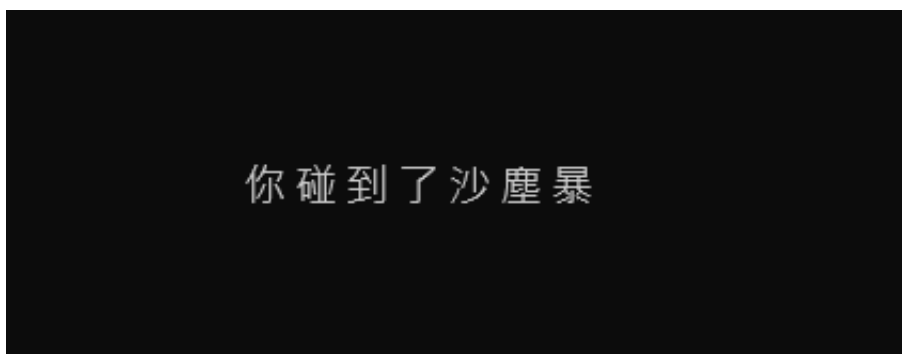
Room System Design (7.5%)

房間地形也有三種，以下個別敘述

```
void procedure(Player* player)
{
    srand(time(NULL));
    int randomnb;
    while (player->getnowHP() > 0 && dungen[4][4].getenemy()->getnowHP() > 0)
    {
        system("cls");
        player->setlastposition();
        retreatornot = 0;
        mainscence(player);
        system("cls");
        if (dungen[player->getx()][player->gety()].getEN() == 1)
        {
            gotoxy(0, 15);    cout << "                你身處在一片沙漠
            Sleep(2000);
        }
        else if (dungen[player->getx()][player->gety()].getEN() == 2)
        {
            gotoxy(0, 15);    cout << "                你身處在一片森林
            Sleep(2000);
        }
        else if (dungen[player->getx()][player->gety()].getEN() == 3)
        {
            gotoxy(0, 15);    cout << "                你身處在一片沼澤地
            Sleep(2000);
        }
        else
        {
            gotoxy(0, 15);    cout << "                你身處在普通的地牢環境
            Sleep(2000);
        }
    }
}
```

Desert:

在沙漠地形時水分會比其他地形多扣 10，並有 10%的機率碰上沙塵暴，遇到沙塵暴時飽食度和水分都會各扣 10。在沙漠中也有 30%的機率遇到綠洲並增加 50 點水分。



```

if (dungen[player->getx()][player->gety()].getEN() == 1 && retreatornot == 0)
{
    PlaySound(TEXT("沙漠bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    randomnb = rand();
    if (randomnb % 100 <= 10)//碰到沙塵暴
    {
        system("cls");
        gotoxy(0, 15); cout << "                                你碰到了沙塵暴";
        Sleep(2000);
        player->sethunger(-10);
        player->setthirsty(-10);
    }
    randomnb = rand();
    if (randomnb % 100 <= 30)//找到綠洲
    {
        system("cls");
        gotoxy(0, 15); cout << "                                你碰到了綠洲";
        Sleep(2000);
        player->setthirsty(50);
    }
}

```

Forest:

在森林地形時飽食度會比其他地形多扣 10，並有 50%的機率遇到水源，遇到水源時水分會加 50 點。在森林中也有 50%的機率遇到兔子，遇到後可選擇捕捉與否，當捕捉成功(40%成功率)時便會增加飽食度 10 點。另外在森林中有 10%機率遇到熊，遇到熊後會進入掙扎遊戲，在 5 秒鐘內輸入 50 次以上的 a 才能逃脫否則就要扣 5 生命，這部分是由<ctime>涵式庫來實現計算 5 秒鐘的時間，並在 while 回圈內輸入 a 來計次。

你在森林裡遇到一隻熊
 那隻熊一直講著兄弟，有急事，可以轉一下身嗎？
 你感覺生命受到了威脅，趕快按 a 逃脫
 11/50|a

```

else if (dungen[player->getx()][player->gety()].getEN() == 2 && retreatornot == 0)
{
    PlaySound(TEXT("森林bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    randomnb = rand();
    if (randomnb % 100 <= 50)//找到水源
    {
        system("cls");
        gotoxy(0, 15);    cout << "                                你在森林裡找到一處水源                                ";
        Sleep(2000);
        player->setthirsty(50);
    }
    randomnb = rand();
    if (randomnb % 100 >= 50)//遇到動物
    {
        char action;
        system("cls");
        gotoxy(0, 15);    cout << "                                你在森林裡遇到一隻兔子                                ";
        gotoxy(0, 16);    cout << "                                還一邊發出peko~peko~的叫聲                                ";
        gotoxy(0, 17);    cout << "                                輸入c來捕捉，若不捕捉則輸入0                                ";

        while (1) {
            action = _getche();
            if (action == 'c')
            {
                randomnb = rand();
                if (randomnb % 100 <= 40)
                {
                    gotoxy(0, 18);    cout << "                                捕捉成功!                                ";
                    gotoxy(0, 18);    cout << "                                飽食度提升10                                ";
                    player->sethunger(10);
                }
                else
                {
                    gotoxy(0, 18);    cout << "                                捕捉失敗!                                ";
                }
            }
            break;
        }
    }
}

```

```

randomnb = rand();
if (randomnb % 100 <= 10)
{
    system("cls");
    int times = 0;
    char num;
    system("cls");
    gotoxy(0, 15);    cout << "                                你在森林裡遇到一隻熊                                ";
    Sleep(500);
    gotoxy(0, 16);    cout << "                                那隻熊一直講著""兄弟，有急事，可以轉一下身嗎?""                                ";
    Sleep(1000);
    gotoxy(0, 17);    cout << "                                你感覺生命受到了威脅，趕快按a逃脫                                ";

    gotoxy(50, 18);
    SetColor(17);

    for (int h = 0; h < 20; h++)
    {
        if (h >= (float)times / ((float)50 / 20))
        {
            SetColor();
        }
        cout << " ";
    }
    SetColor();
    cout << " " << setw(3) << setfill(' ') << times << "/50";
}

```

Swamp:

在沼澤地形有機會遇到毒氣沼澤，會讓玩家中毒。

你碰到了沼澤毒氣

```
else if (dungen[player->getx()][player->gety()].getEN() == 3 && retreatornot == 0)
{
    PlaySound(TEXT("沼澤bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    randomnb = rand();
    if (randomnb % 100 >= 50)///毒氣沼澤特性
    {
        system("cls");
        gotoxy(0, 15);  cout << "
        if (player->getpoison() == 2 || player->getpoison() == 3)
        {
            player->setpoison(3);
        }
        else
        {
            player->setpoison(1);
        }

        Sleep(2000);
        player->setnowHP(-5);
    }
}
```

你碰到了沼澤毒氣

Optional Enhancement(10%)

金錢系統:

我在地牢中增加了金錢系統，讓玩家擊敗怪物後能獲得一些獎勵，這樣玩家玩起來才有成就感，另外可以用金幣在商店中購買各種東西，也使玩家增添了更多可玩性。

魔力、防禦、暴擊、閃避:

除了基本要求的生命外，我還添加了其他腳色屬性，這些腳色屬性讓玩家在跟怪物對戰時有更多彈性可以操作，例如暴擊的機率可以讓玩家打出逆轉的關鍵，閃避率也可讓玩家在瀕死邊緣起始回生，魔力更可讓玩家施展強力的技能擊敗怪物。

音效:

遊戲中添加了音效，我將它分為背景 BGM 和特殊音效，背景 BGM 是常駐的但特殊音效只有在使用技能、移動和購買東西時才會出現。但因為音樂不能在一個函式中同時撥放，因此為了能讓這兩個音效能正常撥放，我利用兩個撥放音樂的函式

```
PlaySound(TEXT(" 音 樂 位 址 "), NULL, SND_FILENAME |
SND_ASYNC|SND_LOOP);
```

```
mciSendString(TEXT("play 音樂位址"), NULL, 0, NULL);
```

音效的添加能使玩家在遊玩時有更有臨場感，更好玩。

```
if (dungen[player->getx()][player->gety()].getnpc() != NULL)
{
    system("cls");
    gotoxy(0, 15); cout << "                                你遇到了";
    if (dungen[player->getx()][player->gety()].getEN() == 1)
    {
        PlaySound(TEXT("沙漠bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    }
    else if (dungen[player->getx()][player->gety()].getEN() == 2)
    {
        PlaySound(TEXT("森林bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    }
    else if (dungen[player->getx()][player->gety()].getEN() == 3)
    {
        PlaySound(TEXT("沼澤bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    }
    else
    {
        PlaySound(TEXT("地牢bgm.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
    }
    Sleep(1500);
    shopsystem(dungen[player->getx()][player->gety()].getnpc(), player);
}
```

介面：

雖然這次的 project 是文字地牢，但我還是做了一些調整，讓每個文字顯示在不同位置，不再是以每一航區隔，這樣可以有效的利用這個執行的空間。而且當介面要刷新時我也是直接使用 system("cls")，清空介面後再重新印出更新資料後的介面。而如何實現介面的固定性呢？我是利用

```
void gotoxy(double x, double y)
```

```
void gotoxy(double x, double y) // allows to move inside the terminal using coordinates
{
    // the type is double, so objects can move less than 1 unit
    HANDLE hCon = GetStdHandle(STD_OUTPUT_HANDLE);
    COORD dwPos;
    dwPos.X = x; // start from 0
    dwPos.Y = y; // start from 0
    SetConsoleCursorPosition(hCon, dwPos);
}
```

這個涵式可以讓我移動到指定的位置做下一個動作，重複利用這個涵式就可以做出非常工整漂亮的系統介面。

等級:

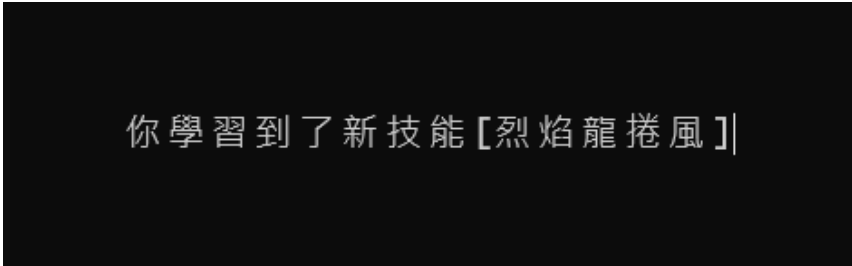
在我玩過的遊戲中通常都有等級機制，因此我也在這個 project 中加入這個元素。玩家通過擊敗怪物獲得經驗值，每升一等基礎數值就會上升，而且升到 5 級時能學習到一個小招，升到 10 級時能學習到大招。

快速輸入鍵:

玩家在輸入指令時不用按下 enter 鍵便可執行指令，這是利用了 `_getche()` 來輸入字元，有了這個涵式玩家便可以加快遊戲進度，不用因每次輸入都要按 enter 而感到煩躁。

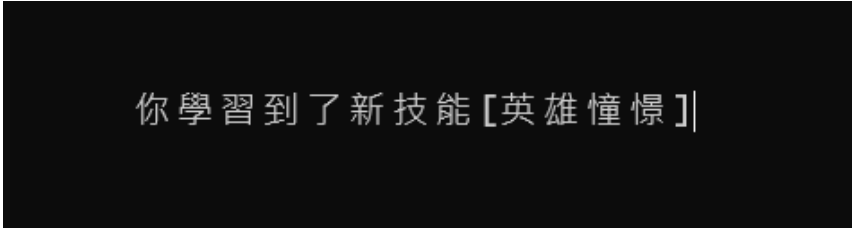
技能系統:

玩家能通過升級來學習技能，技能的學習分為小招和大招，遊戲內的小招有 2 種，玩家會隨機學習其中一種，而大招有 3 種，也是會隨機學習。這裡要注意的是大招有經過特別設計，當施放大招時會進入特殊動畫，搭配震撼的音效，讓玩家更有使出必殺技的感覺。



你學習到了新技能[烈焰龍捲風]

小招



你學習到了新技能[英雄憧憬]

大招

```

void levelup()
{
    srand(time(NULL));
    int randomnb;
    Level += 1;
    if (Level == 5)
    {
        randomnb = rand();
        if (randomnb % 2 == 0)
        {
            skill11 = allskill1[1];
            system("cls");
            gotoxy(0, 15); cout << "
            Sleep(2500);
        }
        else { ... }
    }
    if (Level == 10)
    {
        randomnb = rand();
        if (randomnb % 10 < 4) { ... }
        else if (randomnb % 10 >= 4 && randomnb % 10 < 8)
        {
            skill12 = allskill1[4];
            system("cls");
            gotoxy(0, 15); cout << "
            Sleep(2500);
        }
        else
        {
            skill12 = allskill1[5];
            system("cls");
            gotoxy(0, 15); cout << "
            Sleep(2500);
        }
    }
}

```

你學習到了新技能[烈焰龍捲風];

你學習到了新技能[無量空處];

你學習到了新技能[I am atomic];

補充說明：

Virtual function：

這個 project 中有使用到 Virtual function 的地方只有怪物的攻擊，因為有六種不同怪物的攻擊所以在這個地方採用 Virtual function，來做為不同怪物的攻擊方式。

```

class Enemy
{
public:
    void getdetail() { ... }
    string getname() { ... }
    int getATK() { ... }
    int getHP() { ... }
    int getnowHP() { ... }
    int getDEF() { ... }
    void getdamage(int value) { ... }
    void setnowHP() { ... }
    Enemy() { ... }
    virtual void attack(Player* player) = 0;
protected:
    string name;
    int HP;
    int nowHP;
    int ATK;
    int DEF;
};

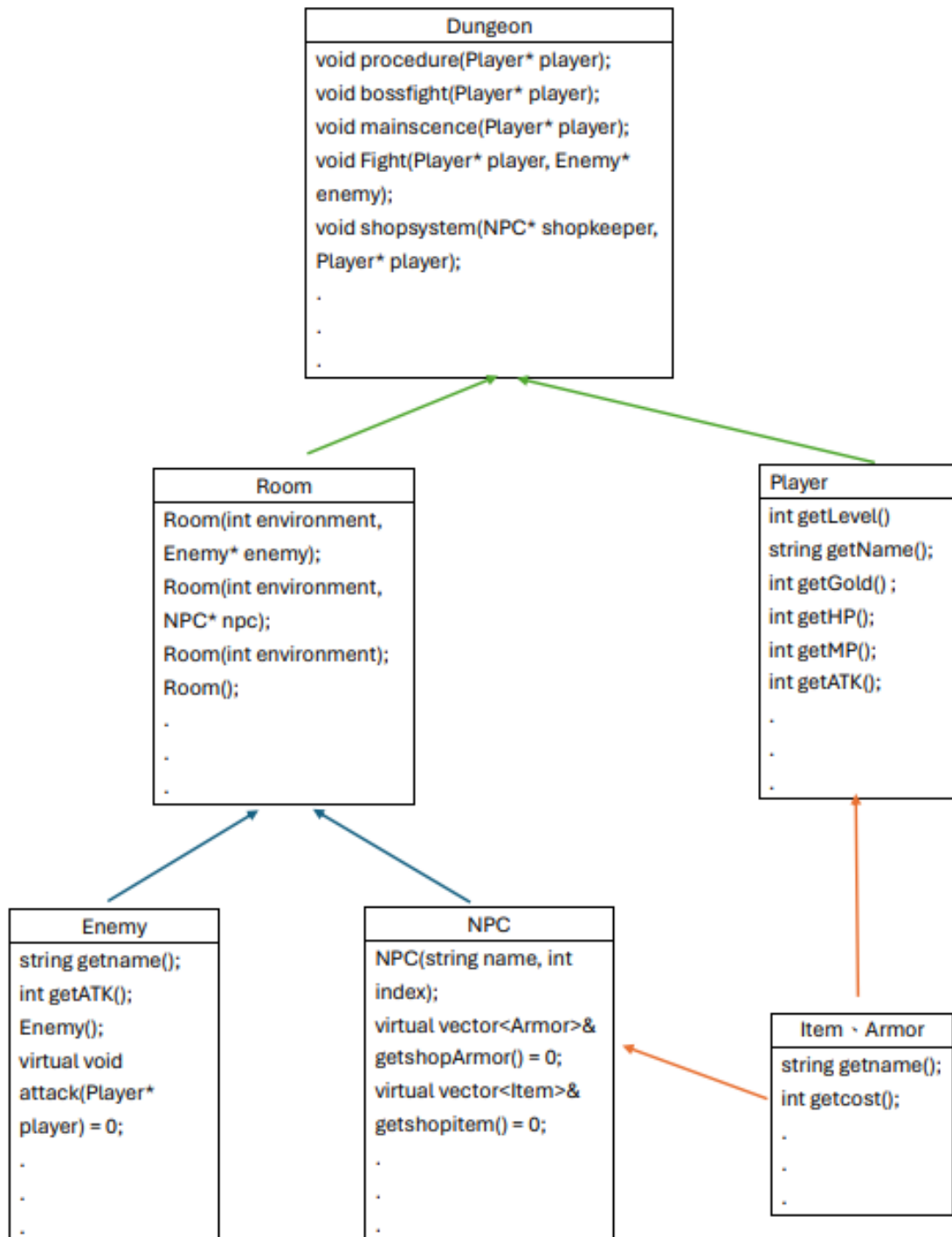
```

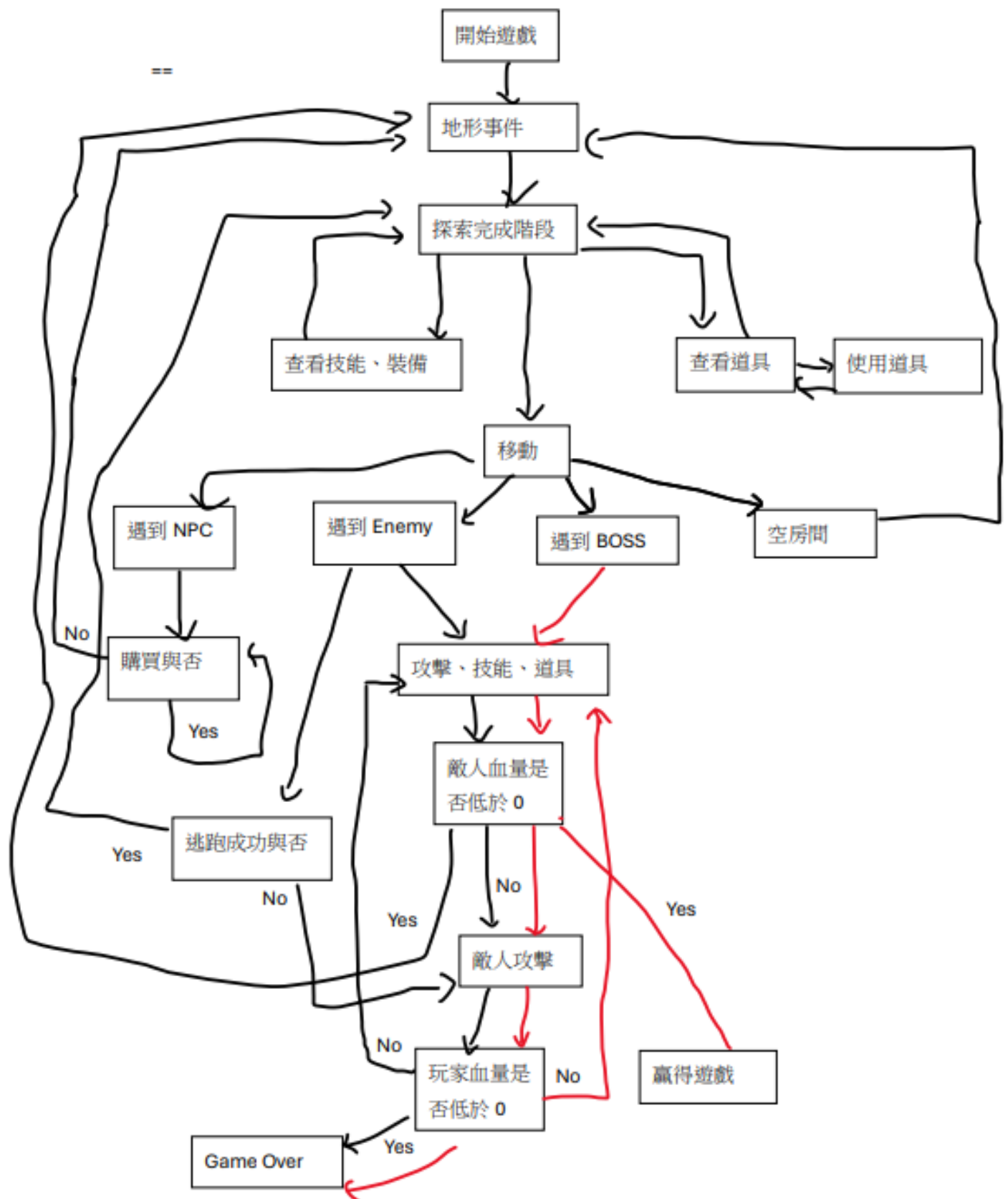
```

class Dragon :public Enemy
{
public:
    Dragon() { ... }
    void attack(Player* player)
    {
        if (getATK() - player->getDEF() <= 0)
        {
            cout << "地牢屍龍攻擊你造成了1點傷害";
            player->setnowHP(-1);
        }
        else
        {
            cout << "地牢屍龍攻擊你造成了" << getATK() - player->getDEF() << "點傷害";
            player->setnowHP(-(getATK() - player->getDEF()));
        }
    }
}

```


UML 介紹





三、問題討論

在本次物件導向課程中，我們設計並實現了一個 C++ 地牢遊戲。在開發過程中，我面臨了一些挑戰和問題，以下是對課程的核心目標、遊戲細節和整體結構方面存在的問題的討論。

1. 物件導向的核心目標未充分實現

在專案中，我們並未充分利用物件導向的核心概念，特別是在繼承方面的應用不足。雖然我在 project 中使用了一些繼承，但是並沒有深入地考慮對象之間的關係，導致部分功能無法有效地複用和擴展。另外，雖然存在一些虛擬函數，但是數量有限，未能充分發揮多態性的優勢。

2. 遊戲某些細節的不完善

在遊戲的具體實現中，存在一些細節問題。例如，裝備的數值玩家無法得到，只能區分強弱；道具的種類和數量有限，缺乏足夠的多樣性。這些問題導致遊戲體驗不夠豐富，玩家的參與度和樂趣有所降低。

3. 遊戲整體結構可以優化

在專案的整體結構方面，存在一些冗余和複雜性過高的情況。某些功能的實現方式過於依賴直覺而不是簡潔的設計，導致程式碼冗長且可讀性較差。我們可以通過重構和簡化程式碼結構，採用更加精簡和優雅的設計來提高程式碼的可維護性和可擴展性。

改進建議

加強繼承和多態性的應用：深入理解繼承和多態性的概念，合理設計對象之間的繼承關係，使得程式碼具有更好的可擴展性和複用性。

完善遊戲細節：增加裝備和道具的種類和屬性，提升遊戲的可玩性和趣味性。考慮添加一些特殊功能和效果，增加遊戲的深度和挑戰性。

優化程式碼結構：通過重構和簡化程式碼，採用更加清晰和模組化的設計，減少冗余程式碼和複雜度，提高程式碼的可讀性和可維護性。盡量避免直覺性地添加功能，而是通過精心設計和思考來實現。

四、心得

當我上大一的時候，我就聽說某位教授在這門課上要求學生製作地牢遊戲。對於已經在高中有製作遊戲經驗的我來說，製作遊戲已經不再陌生。然而，這次的地牢遊戲專案規模更大、更專業。一開始，我沒有想到會花這麼多時間去完成它。在整個專案期間，我撰寫了近 4000 行程式碼。然而，這個過程中，我漸漸體會到了物件導向的目的。我不斷創建各種物件，並將它們串連起來形成了複雜的遊戲關係圖。這讓整個遊戲增添了許多樂趣，同時，如果需要添加新功能，只需在類別中宣告新的變數或函式即可，這讓我在製作遊戲的過程中輕鬆了許多。我希望這次專案的經驗能夠讓我在未來撰寫其他專案時能夠活用所學，進一步提升自己的能力。