我：我已完成以下代码

int event=0;  
    printf("\n今天是在SWMU的周%d\n",player->day);      
    event=rand()%100;  
    if(event<50){  
        int choice;  
        printf("\nOH NO!今天要上早八\n你的选择是:n1.早起学习 2.给自己'放假'\n");  
        scanf("%d",&choice);  
        switch (choice)  
        {  
        case 1:  
            printf("Study Study Study Day Day Up\n");  
            player->PP=80;player->IQ=5;  
            showP(player);  
        break;  
        case 2:  
            printf("'放假'固然舒适，但感觉有种不详之感涌上心头\n");  
            player->PP=120;player->HP=80;  
            showP(player);  
        }  
        }  
    else if(event<100)  
    {  
        int choice;  
        printf("\n今天无早八\n你的选择是:\n1.早起学习 2.外出散步\n");  
        scanf("%d",&choice);  
        switch (choice)  
        {  
        case 1:  
            printf("Study Study Study Day Day Up\n");  
            player->PP=80;player->IQ=5;  
            showP(player);  
            break;  
        case 2:  
            printf("运动有利于身体健康\n");  
            player->PP=80;player->HP=120;  
            showP(player);  
            break;  
    }  
}  
    printf("\n====== 上午事件：小组讨论 ======\n");  
    int choice1;  
    printf("今天有小组讨论\n你的选择是:\n1.积极参与 2.摸鱼划水\n");  
    scanf("%d", &choice1);  
    switch(choice1){  
    case 1:  
        printf("和组员激烈讨论，收获颇丰！\n");  
        player->IQ += 8; player->PP -= 15; player->mood += 5;  
        break;  
    case 2:  
        printf("默默玩手机，但被老师发现了...\n");  
        player->PP -= 20; player->IQ -= 3; player->mood -= 10;  
        break;  
    }  
    showP(player);  
    printf("\n====== 中午事件：食堂选择 ======\n");  
    int choice2;  
    printf("食堂推出新菜品\n你的选择是:\n1.勇敢尝试 2.保守选择\n");  
    scanf("%d", &choice2);  
    switch(choice2){  
    case 1:  
        if(rand() % 100 < 70){  
            printf("新菜品意外地好吃！\n");  
            player->HP += 15; player->PP += 10; player->mood += 15;  
        } else {  
            printf("新菜品味道奇怪，肚子不舒服...\n");  
            player->HP -= 10; player->PP -= 5; player->mood -= 10;  
        }  
        break;  
    case 2:  
        printf("安全第一，选择了经典套餐\n");  
        player->HP += 10; player->PP += 5; player->mood += 5;  
        break;  
    }  
    showP(player);

请根据以上形式在帮我续写2-3个校园场景下事情

AI：// 固定事件3：学习难题

printf("\n====== 下午事件：学习难题 ======\n");

int choice3;

printf("遇到难题不会做\n你的选择是:\n1.请教老师 2.自己钻研\n");

scanf("%d", &choice3);

switch(choice3){

case 1:

printf("老师耐心讲解，茅塞顿开！\n");

player->IQ += 10; player->PP -= 10; player->mood += 8;

break;

case 2:

if(rand() % 100 < 60){

printf("苦思冥想，终于解决了问题！\n");

player->IQ += 15; player->PP -= 20; player->mood += 12;

} else {

printf("想了很久还是不会，有点沮丧...\n");

player->IQ += 5; player->PP -= 15; player->mood -= 8;

}

break;

}

show(player);

// 晚间事件（根据心情值有不同的选择）

printf("\n====== 晚上事件：自由时间 ======\n");

int nightChoice;

if(player->mood > 60){

printf("心情不错！晚上的计划是:\n1.图书馆自习 2.运动锻炼 3.和朋友出去玩\n");

} else if(player->mood > 30){

printf("心情一般，晚上的计划是:\n1.图书馆自习 2.运动锻炼 3.宿舍休息\n");

} else {

printf("心情低落，需要调整:\n1.听音乐放松 2.早点睡觉 3.吃零食治愈\n");

}

scanf("%d", &nightChoice);

switch(nightChoice){

case 1:

printf("在图书馆高效学习！\n");

player->IQ += 10; player->PP -= 15; player->mood += 3;

break;

case 2:

if(player->mood > 60){

printf("和朋友一起运动，超开心！\n");

player->HP += 15; player->PP += 10; player->mood += 10;

} else {

printf("独自运动，出出汗舒服多了\n");

player->HP += 15; player->PP += 10; player->mood += 5;

}

break;

case 3:

if(player->mood > 60){

printf("和朋友出去玩，超开心！\n");

player->HP += 10; player->PP += 15; player->mood += 12;

} else if(player->mood > 30){

printf("在宿舍好好休息\n");

player->HP += 20; player->PP += 25; player->mood += 8;

} else {

printf("吃零食看剧，心情好多了\n");

player->HP += 10; player->PP += 15; player->mood += 15;

}

break;

}

我：函数声明中const的作用

AI：非常好的问题！const在函数声明中是一个很重要的概念，尤其在C语言中它能帮助我们写出更安全、更可靠的代码。

const的基本含义

const关键字用于声明一个常量，表示这个变量的值不能被修改。

在函数参数中使用const的几种情况

1. 保护指针指向的数据不被修改

这是最常见的使用场景，就像我们代码中的情况：

```c

void showStatus(const Player \*player);

```

含义：player是一个指向Player结构体的指针，但不能通过这个指针修改Player的内容。

为什么这样用？

· 安全保证：函数只是"查看"数据，不会意外修改

· 代码意图清晰：告诉其他程序员这个函数不会修改数据

· 编译器优化：编译器知道数据不会被修改，可以进行更好的优化

示例对比：

```c

// 错误：这行代码在const指针下会编译错误

void showStatus(const Player \*player) {

player->health = 100; // 编译错误！不能修改const数据

}

// 正确：只能读取，不能修改

void showStatus(const Player \*player) {

printf("生命值: %d\n", player->health); // 正确：只是读取

}

```

2. 保护普通变量参数

```c

void printNumber(const int num) {

// num = 10; // 错误！不能修改const参数

printf("%d\n", num); // 正确：只是读取

}

```

3. 保护数组内容

```c

// 不能通过指针修改字符串内容

void printString(const char \*str) {

// str[0] = 'A'; // 错误！不能修改

printf("%s\n", str); // 正确

}

```