科研和工程中 C++编程

——车天一分报告

一、项目简介

本项目是一款有故事情节背景的益智闯关类游戏。以经典游戏魔塔为模型，本团队使用 C++语言对其进行重新的编程实现，最终得到了实现部分功能楼层的成品的课程项目。下图展现了魔塔游戏的初始化界面。

魔 塔 游 戏

二、个人分工

(1) 设计数据类，包括人物类和怪兽类

(2) 初始化人物信息、怪兽属性

(3) 设计基本业务逻辑算法思路

二、分工实施

(1) 数据类设计

2.1.1 人物类

勇士即为该游戏的主角，其属性由人物类来定义。再魔塔闯关游戏中，勇士具有的属性总共包括生命值、攻击力、防御力、经验、金币、等级以及钥匙数，这些属性会在之后的逻辑对抗中发挥重要的作用。此外，本类分装各变量，并用 get 和 set 函数将其暴露出去，右图即为人物属性的模板图。

头文件 person.h 如下所示。

```
#ifndef PERSON_H
#define PERSON_H

#include "monster.h"
#include <QString>
#include "fighting.h"

class person {
private:
    int life;
    int attack;
    int defense;
    int exp;
    int level;
    int money;
    int yellowkey;
    int redkey;
```

```
    int bluekey;
    bool mainkey;

public:
    person();
    ~person();
    int key_number(int color)const;
    int attack_monster(Monster mons);
    void raise_level();
    void raise_life(int number);
    void raise_attack(int number);
    void raise_defense(int number);
    void gain_key(int color);
    void lose_key(int color);
    void get_mainkey();
    bool mainkey_num()const;

    QString get_life()const;
    QString get_attack()const;
    QString get_defense()const;
    QString get_money()const;
    QString get_level()const;
    QString get_yellowkey()const;
    QString get_bluekey()const;
    QString get_redkey()const;
    QString get_exp()const;
};

#endif // PERSON_H
```

其中输出属性使用 QString 类的字符串形式，输入属性采用变化量输入指标。person()初始化人物信息，attack_monster()为对抗函数逻辑。下面以某些函数举例说明属性构造。

构造函数 person()，初始化信息由人物界面图所示。

```
person::person()
{
    this->life = 1000;
    this->attack = 10;
```

```
    this->defense = 10;
    this->exp = 0;
    this->level = 1;
    this->money = 0;
    this->yellowkey = 1;
    this->redkey = 1;
    this->bluekey = 1;
    this->mainkey = false;
}
```

攻击函数 attack_monster()

```
int person::attack_monster(Monster mons)
{
    int plife = this->life;
    int mlife = mons.get_life();
    int i = this->attack - mons.get_defense();
    if(i<0) i=0;
    int j = mons.get_attack() - this->defense;
    if(j<0) j=0;

    while (plife > 0) {
        plife -= j;
        if (plife <= 0)
            return 0;
        mlife -= i;
        if (mlife <= 0)
            break;
    }

    this->life = plife;
    this->exp += mons.get_exp();
    this->money += mons.get_money();
    if (this->exp >= this->level * 10) raise_level();
    if (this->money >= 25){
        money -= 25;
        this->attack += 2;
        this->defense += 2;
        this->life += 100;
    }
    return 1;
}
```

对抗机制为勇士和怪兽轮流攻击，彼此对对方造成的伤害为自身攻击力减去对方防御力，当自身攻击力低于对方防御力时无法造成伤害。勇士先受到伤害知道一方血量不高于零时结束。最终人物得到的该野怪的经验和金币。当经验不低于等级的十倍时升级，同时当金币不低于 25 时提升综合属性。

生命属性改变函数 raise_life()，变化量为参数。

```
void person::raise_life(int number)
{
    this->life += number;
}
```

生命属性获取函数 get_life()

```
QString person::get_life()const
{
    return QString::number(life,10);
}
```

2.1.2 怪兽类

怪兽是勇士闯关的阻碍，也是勇士得以获得经验、金币，提升属性的重要途径。本游戏包括多种怪兽，属性数值不同，通过类中初始化函数来设置。此外本类还包括属性 get 函数，封装怪兽信息如下所示。

怪兽类头文件

```
#ifndef MONSTER_H
#define MONSTER_H
class Monster
{
public:
    Monster();
    void initial(int life, int attack, int defense,
int exp, int money);
    ~Monster();

    bool get_alive()const;
    void update_alive();
    int get_life()const;
    int get_attack()const;
    int get_defense()const;
    int get_exp()const;
    int get_money()const;

private:
    bool alive;
    int life;
    int attack;
    int defense;
    int exp;
    int money;
};

#endif // MONSTER_H
```

怪兽初始化函数 initial()

```
void Monster::initial(int life, int attack, int
defense, int exp, int money)
{
    this->life = life;
    this->attack = attack;
    this->defense = defense;
    this->exp = exp;
    this->money = money;
}
```

生命属性获取函数 get_life()

```
int Monster::get_life()const
{
    return life;
}
```

(2) 业务逻辑算法思路

在本项目闯关过程中以人物为主线串联整个故事情节。在这一系列过程中涉及到属性的改变、对象之间的对抗等等操作，这一系列基于数据类 person 和 monster 的业务逻辑，在此给出相关联的基本的逻辑算法。

遇到钥匙时调用 gain_key(int color)，输入颜色使钥匙数加一。

遇到门时调用 lose_key(int color)，相应颜色钥匙数减一，否则无法开门。

遇到野怪时调用 attack_monster(Monster mons)，若返回为 0 则表明无法成功战胜，也不会产生对抗；否则人物属性改变，野怪消失。

遇到血瓶时调用 raise_life(int number)，增加生命值。

遇到红宝石时调用 raise_attack(int number)，增加攻击力。

遇到蓝宝石时调用 raise_defense(int number)，增加防御力。

四、实施效果

本实验我主要负责数据类的构建，业务逻辑搭建，以此为其他组员使用 QT 搭建图形界面提供数据和方法。因为此部分内容比较底层，因此在编译成功后没有成果图用于展示，实施效果可见上文中人物信息初始化图。

五、心得和体会

(1) 首先很遗憾没有在项目中成功运用到 MVVM 模式，在之后的实践过程中深切感受到耦合度紧密的项目设计效率的低下，从反面意识到了 C++在科研和工程中架构模式的重要性。不过吸取教训，我们又重新加深了对于 MVVM 的理解，尤其是 command 命令和绑定机制（使用智能指针），会尽力重新实践以此正确框架下的项目设计。

(2) 综合来看袁老师的课，我学习到的一部分时理论，一部分是实践。这两方面对于我都很有帮助，尤其是学习到理论之后在实践时会产生新的问题，比如最后老师也提到的 shared 智能指针绑定的问题，这些都是我所学习到的知识，相信在接下来的学习工作生涯中我一定会学以致用的。

(3) 感谢遇到如此活泼开朗的老师，感谢我所有组员们共同的努力，虽然我们没有做到很好，但我感受到了每个组员的付出还有老师对于我们细心的指点，希望可以在之后可以又更多交集和学习的机会。

六、对课程的建议

因为我们组时没有很好运用 MVVM 框架的组，在第一次迭代的时候也没有将结果给老师看，所以我提的小建议就是或许可以把第一次迭代框架设置为 6 天左右的期中检查，这样子就能保证大家都是按照 MVVM 设计的，也就会减少偏差，增加教学质量。

个人信息

姓名：车天一

学号：3160105470

专业：计算机科学与技术

邮箱：776946432@qq.com