**实验报告四**

**程序设计思路：**实现一个技能的伤害分摊效果，我这里选择的是诸葛亮的技能——东风破。因此我需要写一个函数来完成东风破这个技能，当命中两个敌人时伤害减半，当命中单个敌人时造成全额伤害。在程序执行完后调用析构函数清除。

**程序代码：**

#include<iostream>

using namespace std;

class Hero

{

public:

Hero();

Hero(int num,int hurt,string s,int hero\_blood,

float hero\_orign\_x,float hero\_orign\_y,int x1,int x2,int x3);

Hero(const Hero & name);

~Hero();

void changename(string myname);

int use\_skill1(int num);

Hero &operator=(const Hero &rhs);

void showskill();

// int use\_skill2(float hero\_orign\_x, float hero\_orign\_y);

private:

int skill1;

int skill2;

int skill3;

int orign\_x;

int orign\_y;

int hurt;

int num;

string hero\_name;

};

头文件

#include"Hero.h"

Hero::Hero()

{

cout <<"this is a defult consturctor"<<endl;

cout << "英雄：诸葛亮" << endl;

cout << "技能一：Dongfengpo" << endl;

cout << "技能二：Tujin" << endl;

cout << "技能三：Yuanqidan" << endl;

skill1=1000;

skill2=500;

skill3=2000;

}

Hero::Hero(int num,int hurt,string s,int hero\_blood,

float hero\_orign\_x,float hero\_orign\_y,int x1,int x2,int x3)

{

cout <<"this is a overload consturctor"<<endl;

skill1=x1;

skill2=x2;

skill3=x3;

}

Hero::Hero(const Hero & name)

{

cout <<"this is a copy consturctor"<<endl;

skill1=name.skill1;

skill2=name.skill2;

skill3=name.skill3;

}

Hero ::~Hero()

{

cout <<"this is a desturctor"<<endl;

cout << "对象：诸葛亮已被清除" << endl;

}

Hero & Hero::operator=(const Hero & rhs)

{

if (this==&rhs)

{

return \*this;

}

this->skill1=rhs.skill1;

this->skill2=rhs.skill2;

this->skill3=rhs.skill3;

return \*this;

}

void Hero::showskill()

{

cout <<"技能一"<<skill1<<endl;

cout <<"技能二"<<skill2<<endl;

cout <<"技能三"<<skill3<<endl;

}

int Hero::use\_skill1(int num)

{

if (num == 2)

{

return 500;

}

if (num == 1)

{

return 1000;

}

}

/\*int Hero::use\_skill2(float hero\_orign\_x, float hero\_orign\_y)

{

if (num == 2)

{

return 500;

}

if (num == 1)

{

return 1000;

}

}

\*/

主函数

#include<iostream>

#include"Hero.h"

using namespace std;

string sunwukong;

int skill;

int num=2;

int aim1=5000;

int aim2=3000;

int main()

{

Hero Zhugeliang;

Zhugeliang.showskill();

cout << "回合开始" << endl;

cout << "1：技能一 2：技能二 3：技能三" << endl;

cin >>skill;

if (skill == 1)

{

aim1=aim1-Zhugeliang.use\_skill1(num);

aim2 = aim2- Zhugeliang.use\_skill1(num);

cout << "目标一生命值" << aim1;

cout << "目标二生命值" << aim2 << endl;

}

while (1)

{

cout << "按1继续释放技能：东风破" << endl;

cin >> skill;

aim1 = aim1 - Zhugeliang.use\_skill1(num);

aim2 = aim2 - Zhugeliang.use\_skill1(num);

cout << "目标一生命值" << aim1;

cout << " 目标二生命值" << aim2 << endl;

if (aim2 <= 0)

{

cout << "你击杀了一名敌人" << endl;

break;

}

}

while(1)

{

cout << "按1继续释放技能" << endl;

cin >> skill;

aim1 = aim1 - Zhugeliang.use\_skill1(1);

cout << " 目标二生命值" << aim1 << endl;

if (aim1 <= 0)

{

cout << "来犯之敌已尽数诛灭！" << endl;

break;

}

}

return 0;

}

**产生的错误：**在编写这个程序时，我遇到的问题和实验三差不多，也是.h文件和.cpp文件中的对应问题。这个问题解决后剩下的就是debug的事了。

**解决办法：**在程序中设置断点，逐个排查，看看程序到底哪一步出了问题。在debug的时候程序报错时会有详细的提示，根据提示找错误。

**思想感悟：**我觉得实验三和四在一定程度上有很多相似之处，当我搞定实验三后，实验四也就迎刃而解了，给英雄设置的技能就是在类中添加普通成员函数。当程序出现错误时，可以在程序中增添断点，单步执行查找错误。

**兴趣创新：**我对诸葛亮这个英雄的技能进行了复现，尽管程度不高，而且非常的粗糙。我希望能够继续优化下去，制作一个真正的游戏。