计算机学院___高级语言程序设计___课程实验报告

实验题目：魔兽世界(一)备战		学号：202200400053
日期：2024-05-16	班级： 2202	姓名： 王宇涵
Email：1941497679@qq.com		

实验目的：

锻炼面向对象综合设计能力.

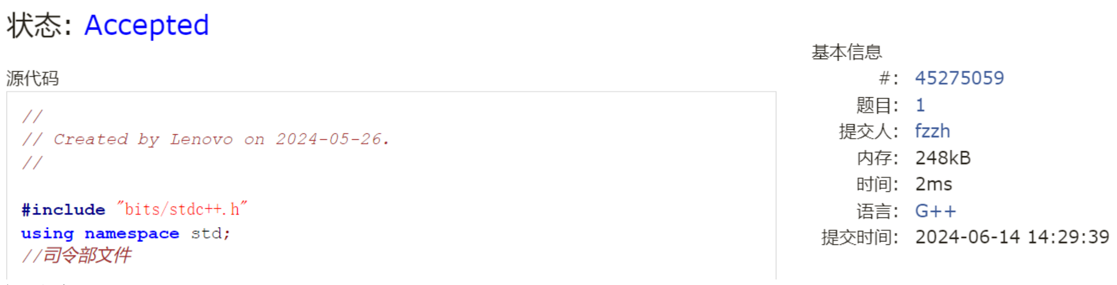
实验软件和硬件环境：

软件环境 ：Clion

硬件环境 ：Legion Y7000P

实验步骤与内容：

AC 截图如下



状态: Accepted

源代码

```
//
// Created by Lenovo on 2024-05-26.
//

#include "bits/stdc++.h"
using namespace std;
//司令部文件
```

基本信息

#:	45275059
题目:	1
提交人:	fzzh
内存:	248kB
时间:	2ms
语言:	G++
提交时间:	2024-06-14 14:29:39

需求分析

本次实验的主要需求为红蓝司令部依此按时间顺序生成武士，并输出对应的操作过程.

设计思路

总体思路设计司令部类， 通过调用生成武士的函数来进行功能完成， 时间和编号由类自己维护成员变量.

设计司令部类 headQuarter, 具体信息如下:

```
class headQuarter{
private:
      int color;//0 代表红色，1 代表蓝色
      int totalLifeValue; //总生命元
      int warriorsNum[6];//生成的武士对应的个数
      int totalWarriorNum;//生成的武士数量
      int makingSeq[6];//武士的制造顺序
      map<int, string> warriorName;//武士的名字
      int warrior2LifeValue[6] ;//武士对应的生命值
      int curMakingIndex;//当前制造的武士的索引
      int curId;//当前制造的武士的 id
      int curTime;//当前时间
      int minLifeValue;//武士最小生命值
      bool Stop;//是否停止制造武士
public:
      //制造武士
      void makeWarrior();
      //构造函数
      headQuarter(int color_, int lifeValue_, int *makingSeq_, int *warrior2LifeValue_);
      bool isStop() const {
          return Stop;
      }
};
```

制造武士函数 makeWarrior 如下:

```
void headQuarter::makeWarrior() {
    if (Stop) {//如果已经停止制造武士
        return;
    }
    if(minLifeValue > totalLifeValue){//如果生命元不够制造最小生命值的武士
        cout << setw(3) << setfill('0') << curTime << " ";
        cout << (color == 0 ? "red" : "blue") << " headquarter stops making warriors" << endl;
        Stop = true;
        return;
    }
    int lifeValue = warrior2LifeValue[makingSeq[curMakingIndex]];//当前制造的武士的生命值
    if(totalLifeValue >= lifeValue){//如果生命元足够制造当前武士
        totalLifeValue -= lifeValue;//生命元减少
        totalWarriorNum++;
        warriorsNum[makingSeq[curMakingIndex]]++;//当前武士的数量增加
        curId = totalWarriorNum;//当前武士的 id
        cout << setw(3) << setfill('0') << curTime << " ";
        cout << (color == 0 ? "red" : "blue") << " " << warriorName[makingSeq[curMakingIndex]] << "
"
        << curId << " born with strength " << lifeValue << "," <<
warriorsNum[makingSeq[curMakingIndex]] << " " <<
        warriorName[makingSeq[curMakingIndex]] << " in " << (color == 0 ? "red" : "blue") << "
headquarter" << endl;
        //更新当前制造的 Index，映射到 1~5
        curMakingIndex = (curMakingIndex + 1) % 6;
        if (curMakingIndex == 0)
            curMakingIndex = 1;
    } else {//如果生命元不够制造当前武士
        curMakingIndex = (curMakingIndex + 1) % 5;
    }
    curTime++;//时间增加
}
```

主测试函数如下

```
int t, m;
    int d, n, i, l, w;
    cin >> t;
    for (int j = 1 ; j <= t ; j ++){
        cout << "Case:" << j << endl;
        cin >> m;
        cin >> d >> n >> i >> l >> w;
        headQuarter redHeadQuarter(0, m, new int[6]{0, 3, 4, 5, 2, 1},
                                                       new int[6]{0, d, n, i, l, w});
        headQuarter blueHeadQuarter(1, m, new int[6]{0, 4, 1, 2, 3, 5},
                                                         new int[6]{0, d, n, i, l, w});
        while (true) {
            redHeadQuarter.makeWarrior();
            blueHeadQuarter.makeWarrior();
            if (redHeadQuarter.isStop() && blueHeadQuarter.isStop()) {
                break;
            }
        }
    }
```
功能划分

1. headQuarter.h 司令部类的函数定义和实现
2. main.cpp 输入输出和对应测试函数

测试截屏

测试样例 1

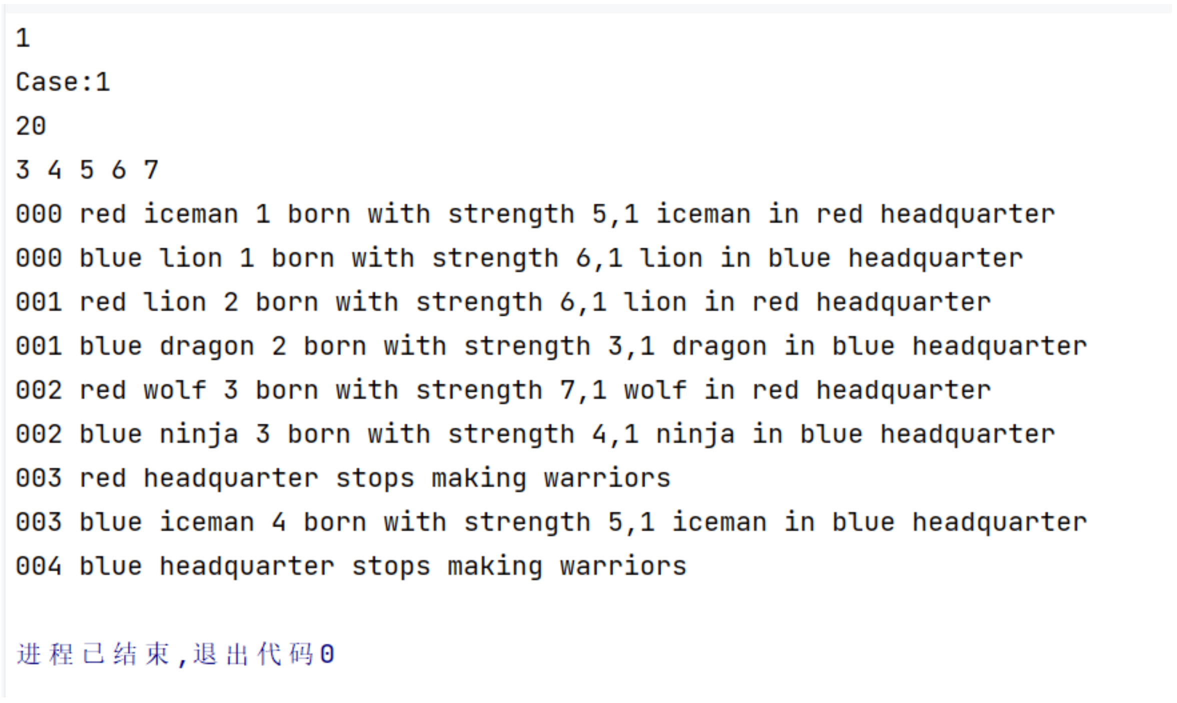
输入:

1

20

3 4 5 6 7

输出:



```
1
Case:1
20
3 4 5 6 7
000 red iceman 1 born with strength 5,1 iceman in red headquarter
000 blue lion 1 born with strength 6,1 lion in blue headquarter
001 red lion 2 born with strength 6,1 lion in red headquarter
001 blue dragon 2 born with strength 3,1 dragon in blue headquarter
002 red wolf 3 born with strength 7,1 wolf in red headquarter
002 blue ninja 3 born with strength 4,1 ninja in blue headquarter
003 red headquarter stops making warriors
003 blue iceman 4 born with strength 5,1 iceman in blue headquarter
004 blue headquarter stops making warriors

进程已结束,退出代码0
```

测试样例 2

输入

2

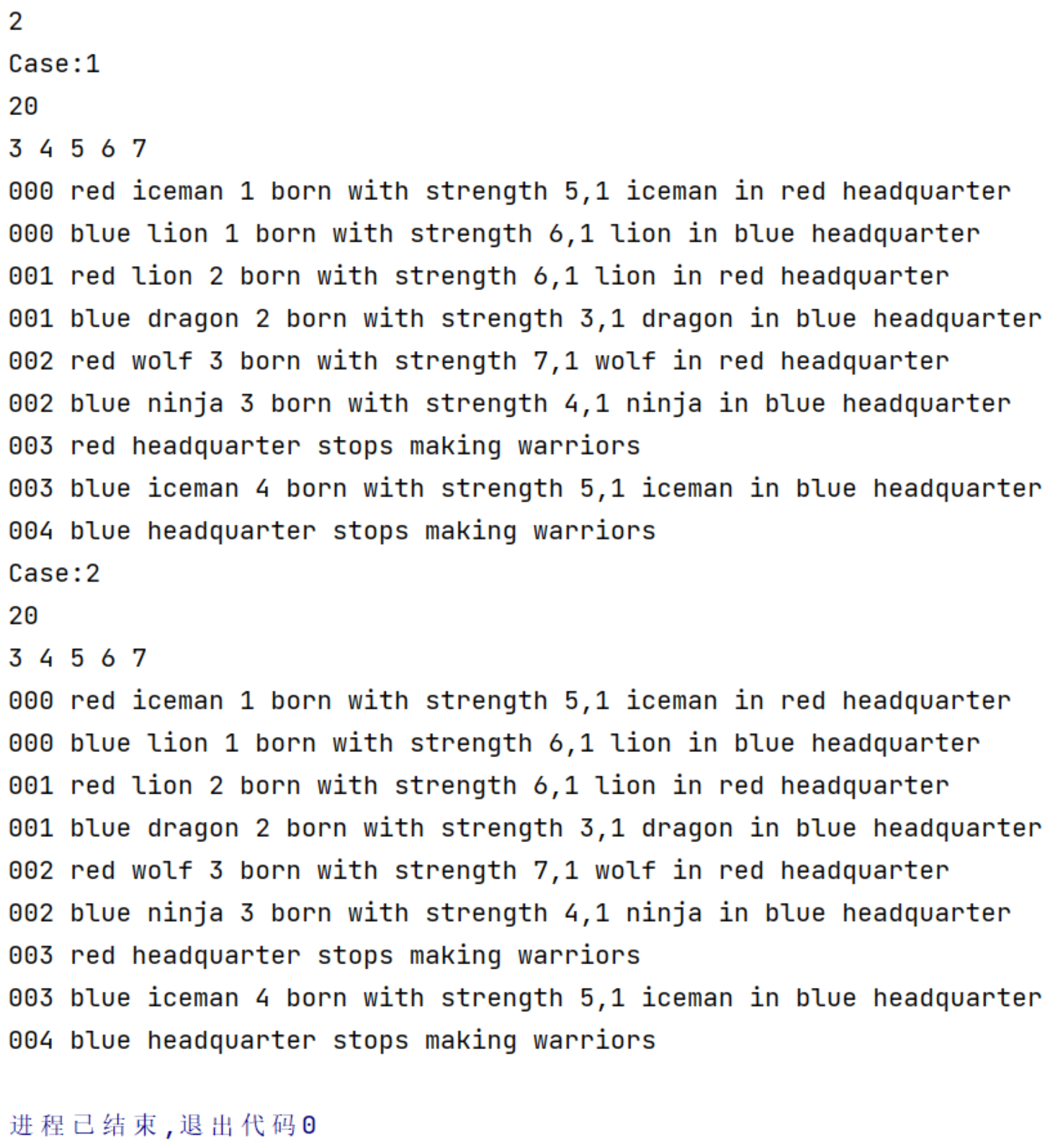
20

3 4 5 6 7

20

3 4 5 6 7

输出:



```
2
Case:1
20
3 4 5 6 7
000 red iceman 1 born with strength 5,1 iceman in red headquarter
000 blue lion 1 born with strength 6,1 lion in blue headquarter
001 red lion 2 born with strength 6,1 lion in red headquarter
001 blue dragon 2 born with strength 3,1 dragon in blue headquarter
002 red wolf 3 born with strength 7,1 wolf in red headquarter
002 blue ninja 3 born with strength 4,1 ninja in blue headquarter
003 red headquarter stops making warriors
003 blue iceman 4 born with strength 5,1 iceman in blue headquarter
004 blue headquarter stops making warriors
Case:2
20
3 4 5 6 7
000 red iceman 1 born with strength 5,1 iceman in red headquarter
000 blue lion 1 born with strength 6,1 lion in blue headquarter
001 red lion 2 born with strength 6,1 lion in red headquarter
001 blue dragon 2 born with strength 3,1 dragon in blue headquarter
002 red wolf 3 born with strength 7,1 wolf in red headquarter
002 blue ninja 3 born with strength 4,1 ninja in blue headquarter
003 red headquarter stops making warriors
003 blue iceman 4 born with strength 5,1 iceman in blue headquarter
004 blue headquarter stops making warriors

进程已结束,退出代码0
```

结论分析与体会:

本次实验我独立设计了司令部类，维护了内部变量，完成了生成武士的功能，深化了自己的面向对象思维，提高了自己的编程能力，锻炼了面向对象综合设计能力，也更加期待以后的挑战!

就实验过程中遇到的问题及解决处理方法，自拟 1－3 道问答题：

1. 如何判断能不能继续生成武士?

答：可以通过司令部剩余的总生命值和生成武士所需要的最小生命值比较，如果比它小，则不能继续生成武士, Stop 变量置为 true，否则继续进行循环.

2. 如何初始化武士不同的信息?

答：对于名称 string 类型可以通过 map，对于数值 int 类型可以通过数组来维护，将武士的类型编号 1~5 对应到不同的数值上去.