

# 这不重要队（编号 6）的 QT 大作业报告

## ——篮球比赛记录系统

戴航 霍炘均 陈缙开

### 一、程序功能介绍

该程序的功能是描述、记录与回看篮球比赛的具体过程，从而为具体的篮球比赛记录工作减少工作量，以下的具体介绍围绕这点展开。

#### 1、篮球比赛的描述

在主界面中进入“记录新比赛”，输入比赛两方的人数以及球队初始信息（主要是球员名称和号码）以进行初始化。



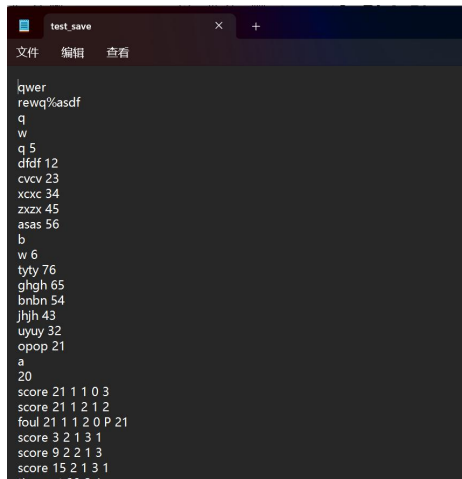
初始化之后，开始正式记录比赛。可以看到，界面上显示了相关按钮。通过按钮的相应操作可以进行记录（例如通过按“三分”按钮，再按球员姓名可以记该球员进三分球）。



结束记录只需要使用左上角的“结束记录”按钮即可。

## 2、篮球比赛的记录

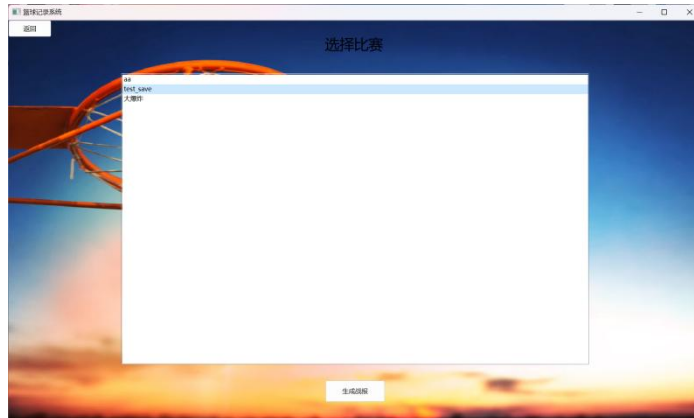
如下图所示，在进行篮球比赛的描述的同时，该程序会同时以.txt 格式记录相应的操作，并进行记录，从而方便后续的回放、调用。



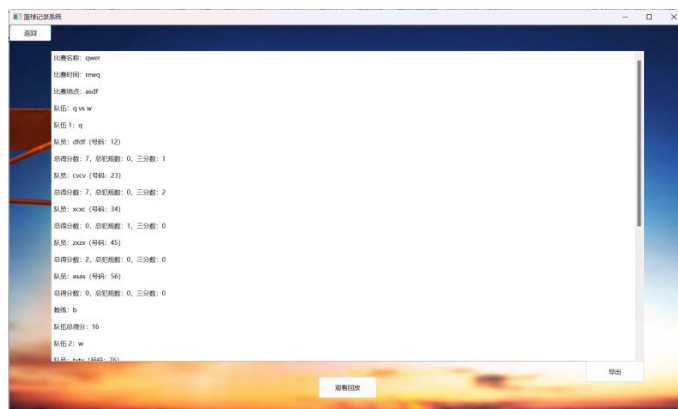
```
qwer  
rewq%asdf  
q  
w  
q 5  
dfdf 12  
cvcv 23  
xcxc 34  
zxzx 45  
asas 56  
b  
w 6  
tyty 76  
ghgh 65  
bnbn 54  
jhjh 43  
uyuy 32  
opop 21  
a  
20  
score 21 1 1 0 3  
score 21 1 2 1 2  
foul 21 1 1 2 0 P 21  
score 3 2 1 3 1  
score 9 2 2 1 3  
score 15 2 1 3 1
```

## 3、篮球比赛的回看

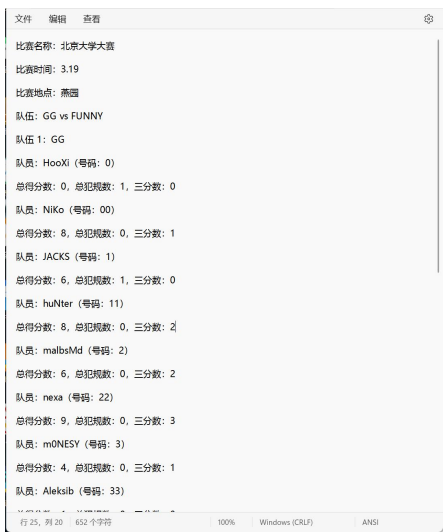
选择篮球比赛场次：



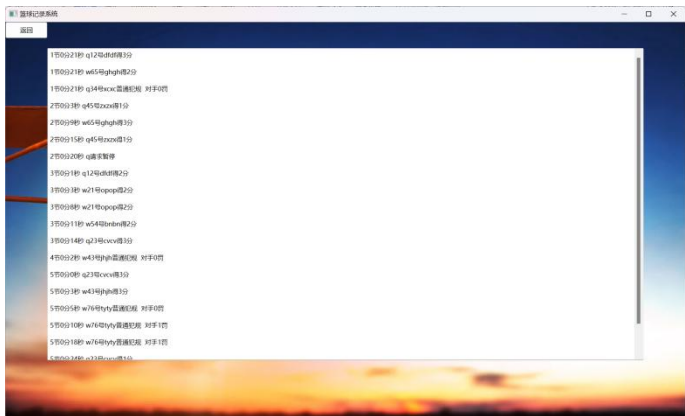
生成战报：里面统计了在参与比赛人员的行为（得分、犯规等）



导出战报：将生成的战报导出成为.txt 文件

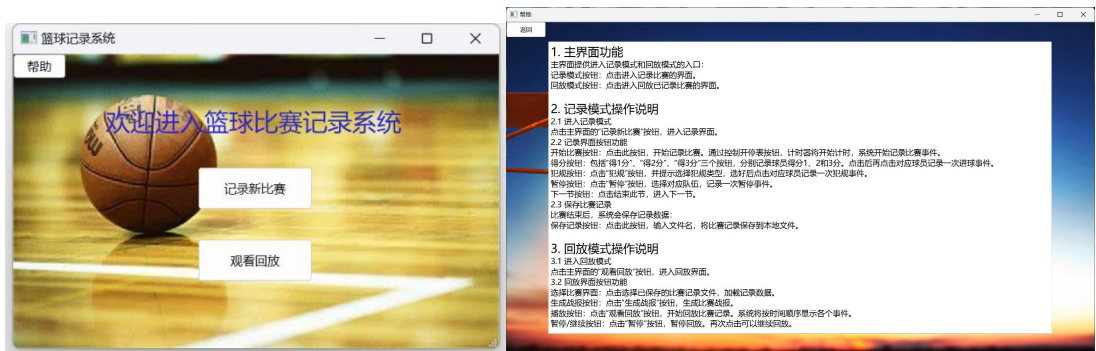


进行回放：



4、帮助界面

为了方便使用，我们还在主界面的左上角设置了“帮助”按钮，用来引导使用者更快上手



二、项目各模块与类设计细节

## 模块设计：

### 1、主窗口 (mainwindow.cpp, mainwindow.h, mainwindow.ui):

这是用户与程序交互的初始界面。它包含记录比赛按钮和回放比赛按钮，用于在不同功能之间导航。

### 2、帮助 (help.cpp, help.h, help.ui):

提供有关如何使用应用程序的指导。此界面中包含用户可用的功能和特性的信息。

### 3、记录模式 (record\_mode.cpp, record\_mode.h, record\_mode.ui):

实际进行记录比赛的模式。用户可以输入有关比赛的详细信息，例如比赛地点、球队信息等重要信息。

### 4、记录 (record.cpp, record.h, record.ui):

提供核心记录功能。随着比赛的进行，实时记录数据。

### 5、回放 (replay.cpp, replay.h, replay.ui):

允许用户回放已经记录的比赛数据。通过回放被记录的事件和统计数据来帮助用户复盘和分析比赛。

### 6、保存 (save.cpp, save.h, save.ui):

将记录的游戏数据保存到文件，确保所有输入的数据都被保存，以便将来参考或分析。

在 .cpp 文件中实现数据处理、比赛逻辑和其他功能。然后设计 UI：使用 Qt Designer 创建 .ui 文件，定义应用程序的布局 and 外观，再在 C++ 中加载和连接 UI。在不同 UI 的关联事件以及同一 UI 的触发事件中采用经典的信号和槽方法传递数据。

## 类设计：

定义类的方法：使用头文件定义应用程序所需的类和方法。

### 1、Game 类

每个 **Game** 类包含了比赛的名称、地点、时间以及比赛球队。

## 2、Team 类

每个 **Team** 类代表一个队伍，包含了队伍人数、队员指针组以及教练，包含的参数有队伍得分数、队伍犯规数等，同时也有队伍得分、犯规等事件函数方便操作。

## 3、Player 类

每个球员可以通过一个 **Player** 类的对象来表示。这个类包含球员名字、比赛表现数据等属性。每个 **Player** 也有自己的得分、犯规等事件函数，随队伍一起调用。

## 4、Event 类

比赛事件如得分、犯规和换人可以通过一个 **Event** 类来表示。这个类包含事件类型、时间和相关球员等属性。

使用指针动态分配和管理 **Player** 和 **Event** 对象。这允许灵活的内存管理，特别是在球员和事件数量不固定时。同时，使用指针访问和更新球员统计数据和事件详情。这确保了对象的更改在所有使用这些指针的地方都得到反映。

## 5、Score 类

**Event** 的派生类，用于记录得分事件。

## 6、Foul 类

**Event** 的派生类，用于记录犯规事件。

## 7、Timeout 类

**Event** 的派生类，用于记录暂停事件。

# 三、组内分工

类设计：戴航

类实现：戴航、陈缙开

UI 设计与实现：霍炘珣

代码实现：戴航、霍炘珣、陈缙开

## 四、项目总结与反思

### 1、设计背景

技术需求来源于生活。某协会（某 KUBA）某部门某人深受某项单调乏味的某工作的某种折磨，具体而言，该工作是实时记录篮球比赛事件以及赛后总结战报。在实时记录比赛事件时，常常由于人手不足而忙得不可开交，从而导致记录字迹极为潦草且十分容易出错甚至无法完全记录所有事件。在赛后总结战报的时候，总会因为该项工作单一且枯燥的特性而哈欠连天。出于以上原因，一款能够减轻某工作的工作量并且具有令某协会满意的功能的程序是非常需要的。

### 2、工作总结

为了解决这些问题，我们开发了一款能够仅通过鼠标点击便可快速记录比赛事件的程序，其记录事件的效率是手写的若干倍，同时还能自动生成战报，大大减少了工作量，可以一定程度上解决某协会人手不足的问题。

在开始工作之前，我们先设计了程序运行中所需要的类和函数，并将它们统一放进一个头文件方便取用。在实现程序的时候，我们选择 UI 和程序分头进行工作推进，在双方均完成某一阶段的工作之后再进行对接，这样使工作过程少有冲突，使得工作进行更有效率。达到一定阶段后，我们便开始测试程序运行的正确性和合理性，使程序被不断修正，最后能够令人在使用时舒适从容。

### 3、工作反思

通过对篮球比赛记录系统的制作，我们处理了许多从未遇见过的问题，在处理这些问题的过程中，我们对于类与指针的理解也更加深刻、运用更加熟练。在此次工作中，预先设计的较为高级的内容未能得到充分的实现，这需要更多的实践经验，让我们深刻认识到将技术应用于实践的时候需要跨过一道鸿沟，我们需要更多的实践来加强理论运用的能力。