**ProjectFIFA**

**——2018世界杯预演及文字直播**

16337019 陈浩玮 16337017 陈徳毅

**1.问题说明及基本实现方法**

模拟世界杯，首先要了解世界杯的比赛流程，有以下几个步骤：

确定32支入围队伍 -> 分组抽签 -> 小组赛，确定16支出现队伍 -> 16强赛 -> 8强赛 -> 半决赛 -> 季军赛及决赛。

根据足球赛性质，设计以下类：

(1) Player

类成员: string:name, string:team, string:pos(位置), int:num(球衣号)。

(2) Team

类成员: string:country(所属国)，int:rank（世界排名），int:group（小组赛分组），string:rgn（所属地区），vector<Player>:GK,DF,MF,FW（所有位置球员集合）。

(3) Match

类成员: Team:teamA,TeamB(比赛双方)，string:venue(场馆)，int:date（日期）。

类行为：两个球队常规比赛，返回双方比分；两个队点球，返回比分。

根据世界杯流程，设计以下类：

(1) Draw

传入32个队，根据世界排名放在4个罐子内；根据世界杯规则，随机分成8个组，将分组结果传回main函数；打印罐子、分组结果，并将结果保存在文件。

(2) Result

记录各球队比赛结果，在小组赛结束后统计数据，确定出线球队并将其传回main函数；在所有比赛结束后生成所有球队排名；记录各球员进球数，在所有比赛结束后生成射手榜；将所有相关数据保存到文件。

(3) GroupStage

传入分组结果，根据文件中记录的真实世界杯日程安排日程；打印日程，并保存在文件中；按照日程进行比赛；

(4) KnockoutStage

传入16强球队，根据文件中记录的真实世界杯日程安排日程，进行所有淘汰赛阶段比赛；每个阶段结束后打印晋级图；将排名为1~4，5~8，8~16的球队分别传回main函数，用于给Result类生成最终排名。

类流程图：

从文件读入所有队伍信息

allTeams

Draw对象

抽签：

groups[8]

GroupStage对象

小组赛：

比赛结果

小组赛结果：

Result对象

top16

KnockoutStage对象

淘汰赛：

rank1\_4

rank4\_8

rank8\_16

Result对象

最终数据统计：

**2.类实现说明**

**2.1 Team类**

(1) **bool** operator<(**const** **Team** &*other*) **const;**

重载的小于号运算符：除东道主外，按世界排名比较，东道主与其他队比较总是返回true。

**procedure** operator<(**Team**:*other*):

**if**(*other*为俄罗斯) **then** **return** **False**;

**else** **if**(*this*为俄罗斯) **then** **return** **True**;

**else** **return** (*this*->*rank* < *other*.*rank*);

(2) **static** **void** readin(**vector**<**Team**> &);

静态方法：从文件中读入所有队伍信息：

**2.2 Match类，模拟一场比赛的模型**

(1) **void** randomPick(**const** **Team** &team, **vector**<**Player**> &ply) **const**;

给一个队随机挑选11名队员，将挑选结果存入ply中。各位置人数按 GK:MF:DF:FW = 1:4:4:2挑选。下面以挑选DF为例说明挑选过程：

**vector**<**Player**> *DF*;

将该队所有DF球员放入该数组;

**for** *i* **from** 0 **to** 4:

*num* := rand() % sizeof(*DF*); // num取[0,sizeof(DF))中任意整数

*ply*.push\_back(*DF*[*num*]);

将*DF*[*num*]删除;

(2) **pair**<**int**, **int**> Match::match(**Result** &res) **const**;

两队进行比赛。Result类对象用于记录比赛结果。比赛结果结果要求随机，可在模拟前先确定两个队伍进球数地期望值，再用poisson分布生成两队实际进球数。使用poisson分布的原因有以下几点：

1.poisson分布产生的结果为非负整数，符合进球数的要求。

2.相比正态分布，poisson分布只有一个参数，方便使用。而正态分布有两个参数，第二个参数的值不好确定。

为了更真实地模拟世界杯，该比赛结果须同两队世界排名有关。即两队进球数的期望值要与排名有关。但这么设计有个缺点：若两队排名都很高，两队比赛时进球数可能非常多，不符合实际情况。所以稍微修改一下：将（A队进球数期望值 – B队进球数期望值）设定为AB两队排名差的函数，设定两队总进球数期望为4。这样进球数之差就可唯一确定两队比分的期望。

记A = A队进球数期望，B = B队进球数期望，d = A - B

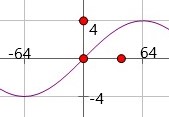
则A + B = 4, 得A = (4 + d) / 2, B = 4 - A。

记得分差函数为:

d = f(x)

则d取值范围为[-4,4],x取值范围为[-64,64]。（排名最低的队伍为65名）。

该函数应该为奇函数。根据直观印象，当排名差变大时，比分差变大的速率应该要减慢。正弦函数的[-pi/2,pi/2]上的图像符合上述要求，故采用正弦函数。对正弦函数作伸缩变换得到得分差函数。



该正弦函数周期T = 256，记f(x) = 4sin(kx)，则k = 2\*pi / T = pi / 128。

x为两队排名差，即x = rankof(B) – rankof(A)。最终函数为：

d = 4sin((rankof(B)-rankof(A))\*pi/128)

A队进球数期望为：

A = (4+d)/2 = 2 + 2sin((rankof(B)-rankof(A))\*pi/128)

B队进球数期望为：

B = 4 – A

例如，若A队为德国队(Rank 1), B队为巴西队(Rank 2)，则德国队进球数期望值为A = 2 + 2sin((2-1)\*pi/128) = 2.05, 巴西队进球数期望值为B = 4 – A = 1.95。胜率基本持平。

再如若B队为冰岛队(Rank 21)，则德国队进球数期望值为A = 2 + 2sin((21-1)\*pi/128) = 2.94，冰岛队为B = 4 – A = 1.06。若B队为日本队(Rank 44)，则A = 2 + 2sin((44-1)\*pi/128) = 3.74，B = 0.26。

一场足球比赛按90分钟模拟，每10分钟内两队进球数期望为分别为A/9，B/9，每个10分钟用poisson随机数生成两队在该10分钟内的进球数。再随机决定这些球是哪几个球员踢进的。

**for** i **from** 0 **to** 9:

**int** *goalofA* := poisson(*A*/9) // A队在该10分钟内进球数

**int** *goalofB* := poisson(*B*/9) // B队在该10分钟内进球数

**for** j **from** 0 **to** *goalofA*:

在除守门员外的所有A队上场球员中随机选一个作为该球打进者。

**for** j **from** 0 **to** *goalofB*:

在除守门员外的所有B队上场球员中随机选一个作为该球打进者。

(3) **pair**<**int**, **int**> Match::penalties() **const**;

两队进行点球大战，返回两队比分。世界杯中点球大战的进球数与结果均不记录在积分榜上，所以不需要用Result类对象记录结果。比赛结果同样要求随机。可分别确定两队的进球率，每次射球时以该队进球率随机产生一个bool值，该bool值为true时表明该球射进，否则表示没进。

为了令点球结果同样与球队排名有关，两队进球率应为排名差的函数。根据上三届世界杯中点球大战的数据，获胜的一方进球率约为80%，所以该函数设定排名高的那支球队进球率为80%。至于排名低的球队，排名每下降一名，进球率下降1%。

记A,B分别为A,B队进球率，不妨设A队排名比B队高