# *PK2048* 编码指南

计算机学院学生会

本文档将于比赛官方网站持续更新，内容以最新版为准。

## 1 游戏介绍

欢迎大家前来挑战 *PK2048* 这款小游戏！如果你玩过 *2048* 这个简单有趣的游戏[1]，你可能会对 *PK2048* 更感兴趣，因为这里不仅有更丰富有趣的玩法，有更紧张精彩的对战，更有人工智能的加盟哦！

你可能曾想过，如果有一个能帮你通关 *2048* 的 AI 该多好。现在梦想该成真啦！你只需要写一个简单的程序，参加到 *PK2048* 中来，你会看到 AI 凭借你的智慧博弈，最终置对手于死地，或者比对手更早达到 2048。

你没有玩过 *2048*？完全没有关系！这个游戏正是因为简单才让人爱不释手；也正是因为简单，每个人都能写出自己独特的 AI 哦。编游戏 AI 太高大上？*PK2048* 正是要让你高大上一把！

闲话少叙，这个文档之后便是游戏的详细规则和编程指南，认真阅读，然后带着你的聪明的小程序来参赛吧！

## 2 详细规则

游戏在两个人间进行。一场比赛分 3 局，最先获得 2 局胜利的一方获得该场比赛的胜利。

游戏局面为一个 $4 \times 4$ 的棋盘，左上标号为 1，之后从左到右、从上到下依次标号到 16。比赛开始时，开始棋盘上 6，7，10，11 格上分别放有

[1] http://gabrielecirulli.github.io/

2，4，4，2。正常情况下，每一回合玩家需要决定向哪个方向移动。移动时，所有有数字的格子向玩家决定的移动方向滑动，直到不能继续滑动为止；如果有数字落到与之相同的数字之上，这个格子会合并到下一个格子中，并把两个数字相加，同时该玩家得到相加后的和对应的分数。数字不允许连加，即一个数字合并后不能再参与之后的合并。玩家的移动必须有意义，即必须有格子移动或者合并，否则视为非法操作。

在一局比赛中，玩家的最终目标是得到更高的分数。除了通过适当的移动，玩家还有两种特殊操作来帮助实现这一目的。每种特殊操作每局仅限使用一次。第一种是干扰，玩家可以放弃一次移动机会，改为指定一个空白格子，在其中放置一个数字 2。第二种是炸弹，玩家可以在某回合开始前在棋盘上现有的任意一个有数的格子中放置炸弹，该格子会立刻变为空白格子，之后玩家继续本回合的移动。

当一名玩家完成他的操作后，该玩家回合结束。随后会在任意一个空白格子处生成一个 2 或 4，生成位置和数字完全随机。之后进入另一名玩家的回合。

当棋盘上出现 2048，或没有合法的操作方式，或一方给出非法操作时，游戏结束。对于前两种情况，得分高者获得一局的胜利；对于后一种，则给出非法操作的玩家直接判负。**如果出现相同分数**，第一种情况下，率先达到 2048 的人获胜；第二种情况判无路可走的人为负；第三种情况不变。

特别注意，如果最后某玩家无法移动，但他手中还有炸弹，游戏不会结束，这时玩家必须将手中的炸弹用掉并进行一步移动。相反地，若一名玩家无法进行任何操作，并且用掉了炸弹，游戏会立刻结束，不管另一位玩家是否有炸弹可用。不要任意给出非法输出！

# 3 程序说明

为本游戏写 AI 程序很简单。程序从标准输入中读取当前状态，输出决策到标准输出流即可。唯一需要特别注意的是：**每次**输出后均需清空缓冲区，否则评测程序无法立刻读到你的输出。方法是：C 语言使用 `fflush(stdout)`，C++ 使用 `cout<<flush`。可以参考下面的样例程序。

## 3.1 输入格式

评测程序会适时给出当前状态的描述。第一行只有一个数，1 表示先手，2 表示后手。第二行是初始状态的描述，之后每次收到程序的决策后，评测程序都会返回更新后的状态描述。状态描述只有一行，共 23 个整数，格式为：首先有 2 个整数，第一个表示格子号，第二个为 2 或者 4, 表示随机生成的数；依照规则不需生成数时这两个数为 0 0。随后有 2 个数，分别表示对方上一回合放置的干扰和炸弹的位置。之后一个数表示对手的移动，1、2、3、4 分别表示上、右、下、左。没有干扰或炸弹时，对应的数为 0；依照规则不能移动的情况下，表示移动的一位为 0。(评测程序保证规则不被破坏)。随后接着 16 个整数 $a_1, a_2, \ldots$，其中 $a_i$ 表示第 $i$ 号格子中当前的数是 $a_i$。特别地，0 表示格中还没有数。最后两个数，分别表示你的和对手的分数。所有的数用空格隔开。

例如，开始时，先手得到的输入会是：

```
1
0 0 0 0 0 0 0 0 0 0 2 4 0 0 4 2 0 0 0 0 0 0 0
```

后手得到的输入则可能是（此时对手已经进行了移动)：

```
2
3 2 0 0 4 0 0 2 0 2 4 0 0 4 2 0 0 0 0 0 0 0 0
```

## 3.2 输出格式

输出每次只有一行，包括 3 个整数。前两个数分别表示干扰、炸弹的放置点的格子号，不放置对应位为 0。第三个数为移动方式，1、2、3、4 分别表示上、右、下、左，不能移动时输出 0。注意，输出必须符合规则，否则直接判负!

例如，如果你的程序某一步决定将 3 号格子炸开并向右移动，你应该输出：

```
0 3 2
```

程序运行是有时间限制的。一个回合的决策时间（从评测机提供输入开始到程序给出输出为止）不得超过 1s，而且由于一些不可预知的原因，时间计量不会十分准确，所以尽量不要做过多运算。

### 3.3 特殊说明

1. 程序将在 Linux 环境下使用 `GCC 4.7.2` 编译。支持数种 `gcc` 标准，但需要在文件开头用注释标注。标注方式为文件**第一行**注明 `//-std=<std>`，其中 `<std>` 可以是 `c99`，`c++11` 中的一种。没有注明采用 `gcc` 默认选项（推荐）。

2. 程序只能有一个源文件。使用多个文件组织程序的，请自行将 `.h`、`.c` 或`.cpp` 合成一个文件。

3. 程序只能使用标准输入输出流，不允许读写任何文件；不允许直接使用 Unix 系统调用，包括但不限于 fork()、exec() 等调用；不允许使用 `#pragma` 编译指令等干涉编译选项的操作。

## 4 样例程序

请在官方网站下载。