



目录

一、游戏背景.....

二、游戏描述.....

游戏单位.....

地图元素.....

战斗系统.....

属性附表.....

三、平台使用说明

四、AI 编写指导.....

1.接口说明

2.移动.....

3.攻击.....

4.技能.....

联系我们.....

3

4

4

5

6

7

12

16

16

18

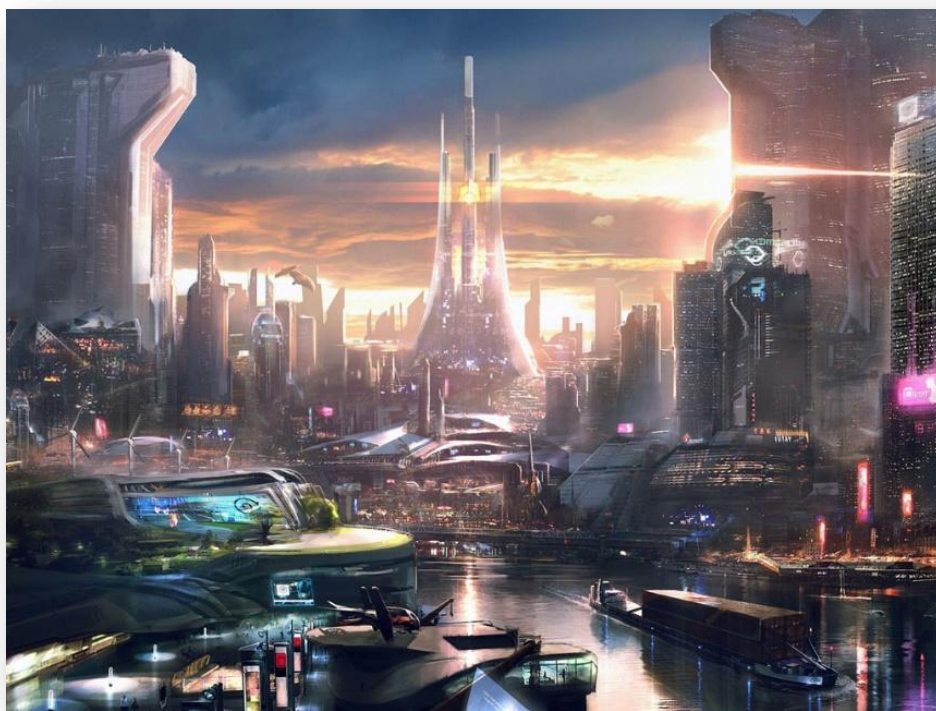
19

19

20

一、游戏背景

几亿年后的未来，地球上的各个板块由于漂移又聚到了一起，经历大灾难后的人类在新的盘古大陆上面建立了新的联合国家。与此同时，当新的宇宙 11 维次元的秘密被揭开之时，人类发现了蜷缩在四维之外的其他维度的相似世界，其发展进程与盘古大陆类似，但科技略微落后。利用量子隧穿效应及强电磁场时空扭曲原理制成的往返于两个世界之间的“魔镜”，掀起了大航海时代以来最大规模的殖民风潮。人们将“魔镜”所连通的理想乡称为“镜界”(Mirror)



那是一个科技十分发达的时代，战场上，人们将通过改造的生化士兵投入战斗，他们拥有着不输于高科技武器的威力，当“镜界”里的人们掌握了这种技术之后，反抗殖民的斗争就开始了……

而你，作为被选中的少年（-_-||），将运用你在“计算机程序设计基础”这门课上学到的知识，设计出 AI 操控你的士兵和将领，战胜敌人，保卫你的家园！

二、游戏描述

游戏需要双方队员编写 AI 程序控制给定的单位在我们的平台上对战。在对战开始时双方均有一定数量的士兵（不超过 10 个）具体兵种与数量确定,并可选择一名将领带领作战。游戏的目标是在一定回合内消灭对方全部的单位（超出回合数比赛仍未结束则比较双方的积分）。



首先我们来具体看一看游戏有些什么**单位**。

游戏中共有 6 种普通的兵种和三种不同的英雄，大家可以看文档后面的[表格](#)来看不同单位的不同属性值，包括生命、攻击、防御、攻击范围、移动力等等。

1. 生命：每个单位的生命值小于等于 0 将被判定为死亡，之后不能再行动；
2. 攻击防御：用于计算攻击的伤害，伤害的计算公式详见后文；
3. 攻击范围：每个单位每回合只能攻击自己攻击范围内的对方单位，攻击范围为对方与己方的格点距离（即横纵坐标差的绝对值之和），行动顺序见后文；

4. 移动力：由于是回合制战棋类游戏，每个回合只能移动各士兵一次，移动范围由移动力与地形的消耗移动力决定，每移出一个地形单位会消耗相应数值的移动力，在移动力未用完的情况下可以继续移动，移动一次后至下一个回合，移动力会恢复。
5. 行动顺序：每一回合由英雄先开始移动，再依据行动顺序双方轮流进行行动，相同兵种固定一种顺序进行（如始终由剑士 1 先动，后剑士 2 行动，两队穿插进行）
6. 技能：三种英雄除了上述的几种属性之外，还各自拥有一项技能。技能说明详见[附录二](#)。特殊的兵种会有技能用于辅助作战。对己方士兵使用的辅助技能只能对相邻单位使用（即格点距离为 1）

除此之外，不同的普通兵种间存在着天生的相生相克，也就是说，伤害的计算还必须考虑单位间的相克性系数。具体参见附录。

接下来我们介绍一下游戏里的**地图元素**。

我们的地图采用不超过 20×20 的矩阵棋盘式地图，就是所谓的“走格子”游戏——地图是由一格一格的离散的地图元素组成，一个格子里只能呆一个单位。每个地图元素有相应的属性，包括消耗的移动力、地形加成效果和积分奖励。

1. 消耗移动力：表示单位在该地图元素上移出后所消耗的移动力；



2. 加成效果：表示单位在该地形上所触发的特殊效果，包括增加在该地形上的单位的属性；
3. 积分奖励：有积分奖励属性的地图元素可以使处于该地图单元上的单位的一方获得额外的加分。

有关地图上的移动的说明：除特殊兵种外，可以通过己方队员到达前面的位置，但不可以通过对方队员（即对方队员位置等效于屏障）。具体的地图按元素属性和参数参见文档后的表格与附录。

最后是我们的战斗系统

前面提到过，游戏采用回合制，每个回合依据攻击顺序由两方的不同兵种交替行动，每次行动分为移动和攻击或使用技能，即在每回合开始使轮到行动的单为可以在允许范围内移动（或干脆停在原地），然后选择攻击或者使用技能或者干脆就待机。

战斗流程示例如下：

回合开始

1. A 队士兵 1 移动 -> A 队士兵 1 战斗 或 技能 或 待机
2. B 队士兵 1 移动 -> B 队士兵 1 战斗 或 技能 或 待机
3. A 队士兵 2 移动 -> A 队士兵 2 战斗 或 技能 或 待机
4. B 队士兵 2 移动 -> B 队士兵 2 战斗 或 技能 或 待机

…往复至回合结束

（说明：1.上述士兵包括英雄，即英雄视作士兵的一种进行操作

2.若 A 队士兵 2 死亡，则 B 队士兵 1 行动后立即有 B 对士兵 2 行动，其余士兵死亡时同理）

当某一单位选择对方作为攻击对象时，先对对方造成伤害，如果攻击方在防守方攻击范围内且防守方受到伤害后仍存活，防守方会给予攻击方攻击伤害（反击）。下面是几项数值的计算式：

伤害 = (进攻方攻击 - 防守方防御) × 相克性

胜负条件：







- 1.若在一定回合内由一方势力全部被另一方消灭，直接将该方判负

2.若到一定回合（暂定 30 回合）内双方都有力量剩余，则以双方积分判定胜负

$$\text{积分} = \text{英雄剩余血量} \times 5 + \text{士兵剩余总血量} \times 3 + \text{累计地形积分奖励}$$

3.若在结束时双方积分相同，以后攻者获胜（视为防御成功）


表格一：地形介绍

地形	描述	图片
平原	大陆上随处可见的平地	
山地	高低起伏的山丘影响士兵的移动，同时便于防守	
树林	巨树参天的战场，回避能力提高	
屏障	耸立的山峦，不允许一般士兵通过	
炮塔	双方设置的远程兵器，被占领后只有特定士兵才会使用	
遗迹	文明时代的古城，偶尔会有增加能力的宝物	
魔镜	利用电磁场的空间转换装置，可以传送到战场上任意位置	

（具体参数参考附录，具体数值以最新发布头文件为准）

表格二：兵种介绍

兵种	描述	攻击能力	防御能力	图片
能量剑士	使用注入了能量的宝剑的近战单位，闻名于高超的剑术	★★★	★★★	
生化突击手	生化改造后的变异士兵，优秀的体能是他们能日行百里	★★	★★★★★	
等离子狙击手	使用先进的等离子步枪的狙击手们往往能在远处杀死敌人	★★	★★★	
战机	装备了新型武器的作战机械，能无视地形傲游整个地图	★	★★★	
坦克	旧时代的作战武器，拥有超高的防御，却相对较为笨重	★★	★★★★★	

治疗师	专门负责补给维修的士兵，可以回复单位生命	0	★★★	
-----	----------------------	---	-----	---

（具体参数参考附录，具体数值以最新发布头文件为准）

表格三：英雄将领介绍

	描述	攻击能力	防御能力	图片
生化狂战士	攻守兼备的优秀生化战士，关键时刻可以分享自己的生命给作战的同伴	★★★★★	★★★★★	
暗杀者	共和国里最具声望的能量杀手，身经百战，常常能一击毙命	★★★★★	★★★★	
大法师	大法师实际上是个科学怪人。他们装备了先进的武器，可进行远程的拟魔法攻击，并且可以增加我方能力	★★★★★	★★★★★	

（具体数值见附录）

（注：附录数值仅供参考，具体数值详见最新发布的游戏参数包）

附录 1：地形参数

地形	消耗移动力	效果	特殊效果	积分奖励
----	-------	----	------	------

平原	1			0
山地	2	防守+1		0
树林	3	10%对方攻击不中		0
屏障	1：战机		除战机外不能通过	0
炮塔	1		狙击手可实现远程攻击，攻击范围 2-10，攻击力不变	2（连续占有 5 回合）
遗迹	1		每回合随机出现镜子碎片（可增加单位的攻击/防御/移动力，士兵最大+3，英雄最大+5）被采到后回合会出现	3（获得一次碎片）
魔镜	1		移动目的地为魔镜时会传送到固定位置，不能再进行攻击或技能	1（进行一次传送）

附录二：兵种参数

兵种	生命	攻击	防御	攻击范围	移动	特技	行动顺序
能量剑士	25	18	12	1	6		1
生化突击手	25	17	13	1	7		2
等离子狙击手	25	17	12	2	6	可操控炮塔	3
战机	21	15	12	1	8	可翻越屏障	4

坦克	30	17	14	1	5		5
治疗师	21	0	12	0	6	回复邻近一单位生命10	6

附录三：相克性

攻 守	剑士	突击手	狙击手	战机	坦克	治疗师
能量剑士	1	0.5	1	0.5	1.5	1
生化突击手	1.5	1	1	1	0.5	1
等离子狙击手	1	1	1	2	2	1
战机	1.5	1	1	1	0.5	1
坦克	0.5	1.5	1	1.5	1	1

附录四：英雄

	生命	攻击	防御	攻击范围	移动	技能
生化狂战士	55	20	14	1	5	牺牲生命 5，回复邻近一单位攻击数值的生命
暗杀者	40	20	13	1	6	10%伤害×2
大法师	45	18	14	1-2	7	使邻近一单位 5 回合内攻击、防御 +1，

三、平台使用说明：

运行 mirror.exe，就会看到我们高大上的主界面：



选择单人游戏，进入这个界面：



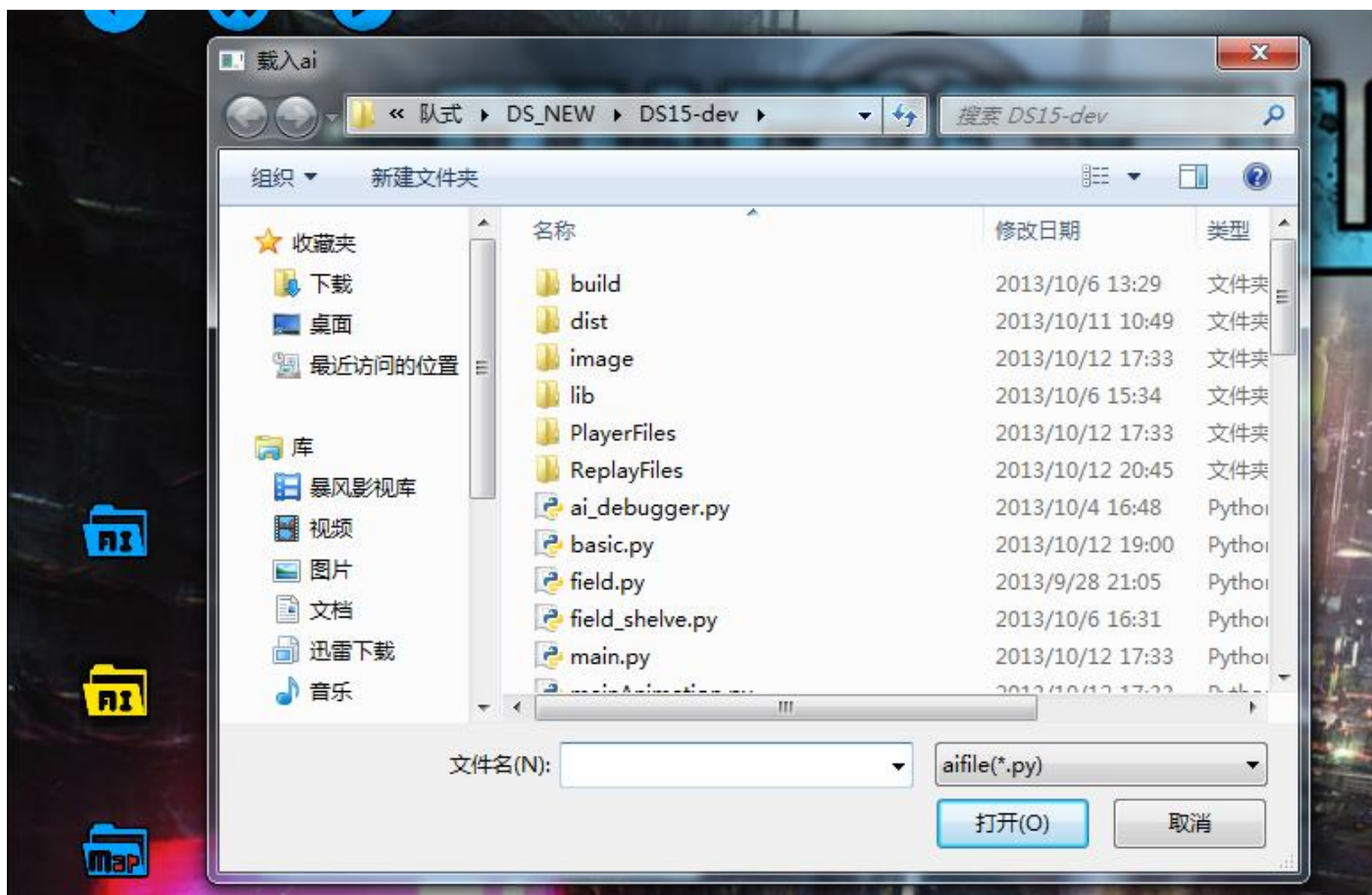
在这里可以选择 AI 对战或人机对战，设置方式类似，先点击

选择 AI 文件



点击

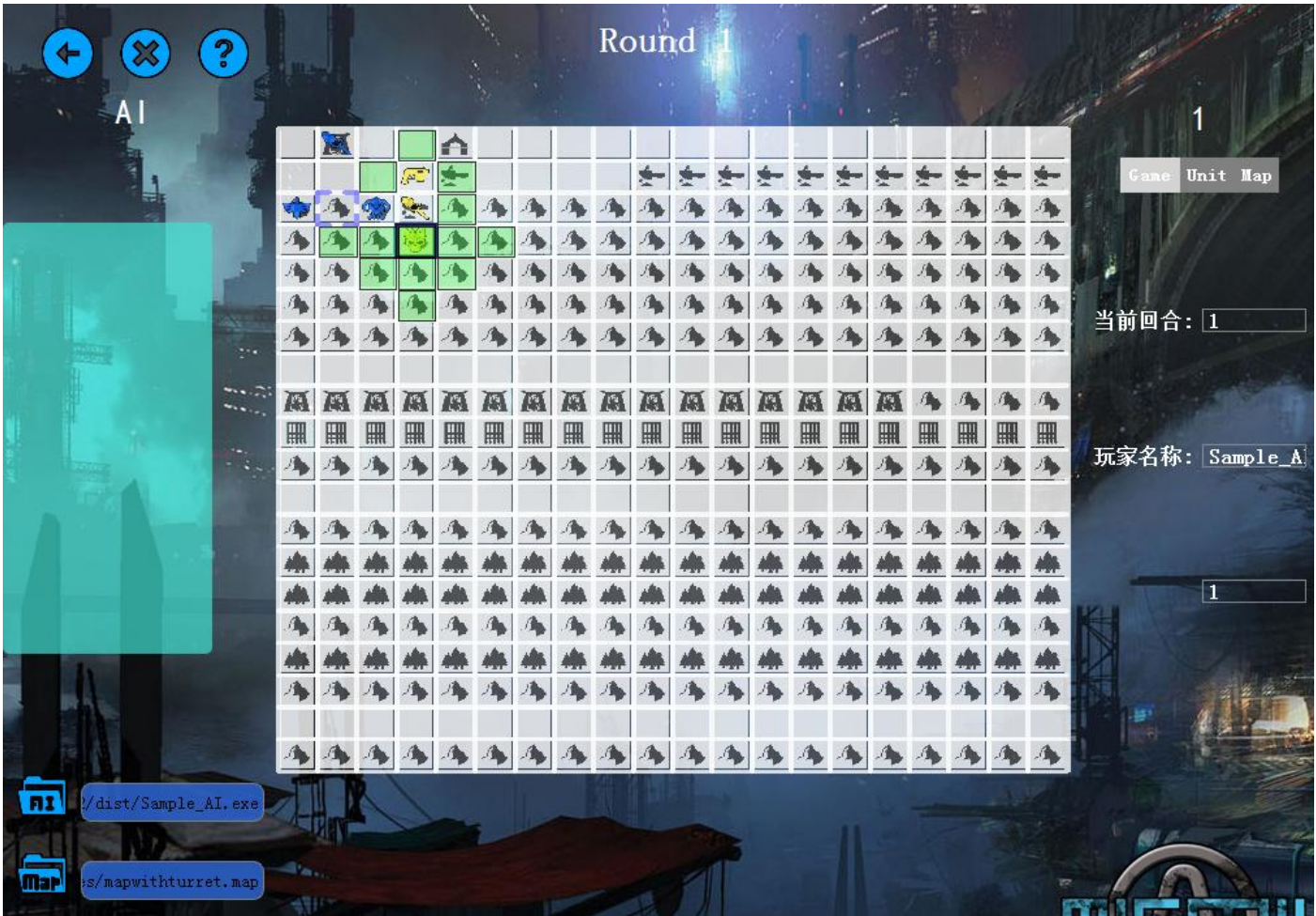
选择地图文件



点 开始运行。神奇的界面就出现了！！！！



想看吗，在下页呢



四、AI 编写指导：

1. 接口说明：

(1) Info

我们为选手提供了储存游戏信息的 info 结构体，选手可以查看 info 的成员来了解地图、士兵状态等信息。Info 类型为 game_info 结构体，具体有以下成员：

```
typedef struct game_info { // 游戏信息结构体，每回合选手从中获取必要的信息
    int team_number;           // 队伍号(0或1)
    int map[COORDINATE_X_MAX][COORDINATE_Y_MAX]; // 地图各点类型
    int map_size[2];           // 地图行[0]、列[1]
    int mirror_number;         // 魔镜数量
    Mirror mir[MIRROR_MAX];    // 各处的镜子位置及出口
    int soldier_number[2];     // 各方单位数量
    int move_id;               // 移动单位，表示本队伍中第move_id个人
    int score[2];              // 两队当前积分(不包括存活士兵所占)
    int turn;                  // 当前总回合数
    int temple_number;         // 神庙数量
    Temple temple[TEMPLE_MAX]; // 各神庙的位置及状态
    Soldier_Basic soldier[SOLDIERS_NUMBER][2]; // 双方单位的信息
}Game_Info;
```

- info.team_number: 类型 int，值为 0 或 1，表示自己所属队伍编号
- info.map_size[]: 类型 int 数组，map_size[0]代表地图的行数，map_size[1]代表地图的列数。
- info.map[]: 类型 二维 int 数组，map[i][j]表示坐标 (i,j) 处的地形，i 的范围是 0 ~ map_size[0]-1，j 的范围是 0 ~ map_size[1]-1，具体地形对应的数字如下：

```
// 地图类型编号
const int PLAIN = 0;    // 平原
const int MOUNTAIN = 1; // 山地
const int FOREST = 2;   // 森林
const int BARRIER = 3; // 屏障
const int TURRET = 4;   // 炮塔
const int TEMPLE = 5;   // 遗迹
const int MIRROR = 6;   // 魔镜
```

每种地形的具体效果参见《规则文档》

- info.mirror_number: 类型 int，表示地图上的镜子数
- info.mirror[]: 类型 结构体 Mirror 的数组，Mirror 包含 inPos 和 outPos 两个成员，表示镜子的入口出口位置。例如 info.mirror[i].inPos.x，info.mirror[i].inPos.y 表示 i 号镜子入口的坐标。
- info.soldier_number[]: 类型 int 数组，下标只有 0 或 1，info.soldier_number[0]表示 0 号队的各类士兵（包括英雄）总个数。

- `info.soldier[][]`: 类型 结构体 `Soldier_Basic` 的数组，第一个下标表示士兵编号，第二个下标是队伍编号。`Soldier_Basic` 内容如下：

```
typedef struct soldier_basic { // 基本单位结构体，包含所有基本的属性
    int kind, life, strength, defence, move_range;
    int attack_range[2]; // attack_range[1][0]表示攻击范围上下限
    int duration; // 单位被英雄3的技能强化的时间
    Position pos;
}Soldier_Basic;
```

例如 `info.soldier[4][1].life` 表示 1 号队第 4 号士兵的生命值，`info.soldier[3][0].pos.x` 表示 0 号队第 3 号士兵的位置的 x 坐标。

- `info.temple_number`: 类型 `int`，表示地图上的神庙数
- `info.temple[]`: 类型 结构体 `Temple` 的数组，`Temple` 成员如下：

```
typedef struct temple { // 神庙结构体
    Position pos;
    int state; // 表示神庙神符状态，0：无碎片；1：攻击+1；2：移动+1；3：防御+1；
}Temple;
```

例如 `info.temple[3].pos.x` 代表 3 号神庙位置的 x 坐标，`info.temple[0].state` 表示 0 号神庙的状态。

- `info.sore[]`: 类型 `int` 数组。`Info.sore[i]`表示 i 号队当前的得分
- `info.turn`: 类型 `int`。已进行的回合数。
- `info.move_id`: 类型 `int`。表示选手当前需要操作的单位（士兵或英雄）编号

(2) cmd

`cmd` 是选手下达命令的结构体变量，其成员如下：

```
typedef struct command { // 选手操作,每回合传给逻辑
    Position destination; // 要移动的目的地
    Cmd_Order order; // wait:待机, attack:攻击, skill:施放技能
    int target_id; // 目标单位
} Command;
```

`cmd` 的使用是在 `ai.cpp` 文件中的 `AI_main()`函数中进行的，例如：

`cmd.order = wait` 就是下令要对当前士兵进行等待（不攻击不施法）操作。

(3) move_range()

学长怎么会忍心坑学弟呢？“零基础”我们说到做到，于是我们直接给选手们提供了这个神奇的函数，只需把 `info`、要查询的单位队伍号和士兵编号作为参数传进去，就能返回该单位的移动范围（可移动到的位置的坐标组成的数组）

具体函数如下：

```
int move_range(Game_Info &gameInfo, int team, int id, Position *tmp )
```

传进去的 tmp 是由选手的一个 Position 数组，建议长度 300，用来储存移动范围。函数会返回一个整数表示该数组长度。

下面给出用该函数计算本队当前要操作的单位的移动范围的示例：

```
Command AI_main()
{
    Command cmd;
    //选手在这里写自己的AI主函数

    int len_move;    //用来接收移动范围数组的长度
    Position move_rg[300]; //传进去的tmp

    len_move = move_range(info, info.team_number, info.move_id, move_rg);

    return cmd;
}
```

是不是很高大上呢 $O(n_n)O$ 。



PS：其实上面这些东西都是 basic.h 和 fun.cpp 中的定义与说明，建议少年们好好看看里面的内容

2. 移动操作：

由于每次操作时，平台已经在 info.move_id 中告诉选手当前行动的单位，所以选手只需要指定移动目的地即可。当前单位可移动范围可以由 move_range() 函数得到，选手应确保指定的目的地在可移动范围内。这里给个简单的示例，向旁边（x 坐标减一的位置）移动一步，示例中并未判断目的地是否在可移动范围内，为了锻炼选手的能力，我把这个问题留给你们自己解决（喂，明明是你懒了好不好！）：

```

Command AI_main()
{
    Command cmd;
    //选手在这里写自己的AI主函数
    if (info.soldier[info.move_id][info.team_number].pos.x > 0) //确认没有站在x=0的边界处
        cmd.destination.x = info.soldier[info.move_id][info.team_number].pos.x - 1;

    cmd.destination.y = info.soldier[info.move_id][info.team_number].pos.y;
    return cmd;
}

```

3. 攻击操作:

把 order 设为 attack，选定 target_id 就可以了。给一个攻击对方 0 号单位的示例：

```

Command AI_main()
{
    Command cmd;
    //选手在这里写自己的AI主函数
    cmd.order = attack;
    cmd.target_id = 0;
    return cmd;
}

```

4. 技能操作:

请选手先认真阅读《规则文档》了解各英雄的技能。其实释放技能和攻击操作类似，这里给一个将军英雄给血最少的小弟加血的技能示例：

```

Command AI_main()
{
    Command cmd;
    //选手在这里写自己的AI主函数
    int min_life = 100, min_life_id = -1; //记录最少的血和对应的士兵号

    for (int i=0; i<info.soldier_number[info.team_number]; i++)
        if (i != info.move_id) //不能是当前单位即将军自己
            if ( (info.soldier[i][info.team_number].life > 0)
                && (info.soldier[i][info.team_number].life < min_life) ) //这个单位活着且血更少
            {
                min_life = info.soldier[i][info.team_number].life;
                min_life_id = i;
            }
    if (min_life_id != -1) //如果找的到活着的且血最少的小弟。。。
    {
        cmd.order = skill;
        cmd.target_id = min_life_id;
    }
    return cmd;
}

```

好了，学长只能帮你们到这了，战术神马的还得靠你们自己思考。命运的车轮已经开始转动了，快去勇敢地战斗吧，少年 $o(\cong v \leq) o \sim \sim$



我们的联系方式：

逻辑组： 朴镜潭 15659218841

平台组： 李步宇 18101360654

界面组： 李栋林 15652774652

展示组： 王康 18001040914

网站组： 池雨泽 15643061003

美工组： 吴旦 18811369861