选手接口文档

1.选手接口
void updateAge()　升级时代
void construct(BuildingType building_type, Position pos, Position soldier_pos) 建造命令 参数一次是建造的建筑类型，建造的位置和出兵位置(非造兵建筑空缺这个参数)
void upgrade(int unit_id)　升级单位命令，参数为目标单位 id
void sell(int unit_id)　出售命令，参数为单位 id
void toggleMaintain(int unit_id)　　修理命令，参数为目标单位 id

2.游戏静态信息
选手通过 bool 变量 flag 获取自己是 0/1 号玩家
选手通过变量 bool** map 来获取地图信息，其中 map 是一个 200*200 的 bool 数组。

3.获取游戏的全局动态信息

```
struct State
{
    int turn;
    int winner;
    _resource resource[2];
    Age age[2];
    vector<Building> building[2];
    vector<Solider> soldier[2];
};
```

游戏信息出存在这样一个结构体中，玩家可以通过指针 state 获取这些信息。
turn 是当前游戏回合数目，winner 为当前胜利者，其他为选手的资源和建筑，兵种
其中 resource age building soldier 这几个数组分别储存了两个选手的信息，其中[0]是 player0 的信息，[1]是 player1 的信息
其中 building[i] solider[i]储存的是装有结构体 Building，Soldier 实例的向量（STL 的 vector）。

4.对应资源信息结构体的实例

```
struct _resource
{
    int building_point;
    int resource;
};

struct Position {
     int x;
```

```
    int y;
};


struct Soldier {
    SoldierName soldier_name;     //soldier 的类型
    int heal;           //soldier 的血量
    Position pos;     //soldier 的位置
    int flag; //soldier 的阵营
    int unit_id;  //soldier 的 ID
};

struct Building {
    BuildingType building_type;   //building 的类型
    int heal;      //building 的血量
    Position pos;     //building 的位置
    int flag; //building 的阵营
    int unit_id;  //building 的 id
    bool maintain;   //building 是否被维修
};
```