目次

―概要―

この章では，プログラミングコンテスト競技部門のルールについて解説します．ゲームの基本となるため，ぜひ目を通しておいて下さい．また，中にはルールとして知ってるのと知らないのでは全然ゲームの難易度が違ってくるルールもあるので，隅から隅まで詳しく読んで少しでも優勝のために頑張って下さい．

1　説明に使う用語・記号について

ゲームに使う基本的な用語や記号と実際のゲームとの対応を説明します。

プレイヤ	ゲームに接続するプログラム (つまり，今回の大会で皆さんに書いてもらうプログラム) とそれを使って対戦する人です．
サーバ	接続されて，対戦の状況などを管理するプログラムのことです．
COOL	ゲームの先攻側です．**COOL 用のポートに接続したプログラム**が先攻になります．
HOT	ゲームの後攻側です．**HOT 用のポートに接続したプログラム**が後攻になります．
ターン	自分が 1 回行動する一連の動作 (getReady，周辺情報の取得，メソッド，周辺情報の取得) をターンと呼びます．
フィールド	ゲームを行う場所のことです．
周辺情報	自分の居るマスを含めた周囲 9 マスの状態と ゲームが終了しているか否かの情報のことです．
メソッド	自分のターン中に 1 つ行える動作のことです．

また，この章ではメソッドの説明など，言葉だけではイメージが掴みづらい部分が多いです．なので，これから使われる図と用語について対応表を書いておきます (そして，COOL を自分として書いていますが，HOT でも同様です)．

自分 (COOL) 

相手 (HOT) 

普通の床 

ブロック 

アイテム 

2　ルール

2.1　基本的なルール

- ⋆ 競技は，サーバに接続された 2 人のプレイヤによる 1 対 1 での対戦になります．
- ⋆ 先に接続した側を先攻になり，後に接続した側が後攻になります．
- ⋆ 勝利条件 (2.2 節を参照) を先に満たしたプレイヤが勝利となります．
- ⋆ 競技に使用するフィールドは，15×17(縦 15 マス，横 17 マス) です．
- ⋆ アイテム・ブロックは，マップの中心から点対称になる様に配置されます．
- ⋆ アイテムを取得したとき，元居たマスは以下の様になります (下記の図を参照)．

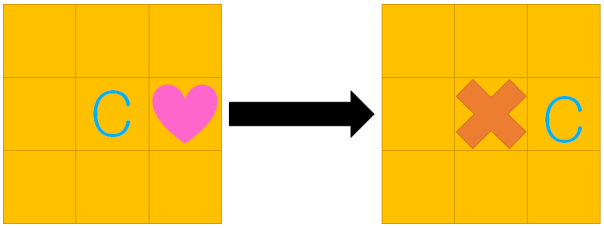


図 1: 動作の例

2.2　勝利条件

- ⋆ 決められた，制限ターンになったときに，
  アイテムを相手よりも多く集めている．
- ⋆ ブロックを対戦相手の上に乗せる．
- ⋆ 対戦相手の上下左右をブロックで囲み，動けなくする．
- ⋆ 対戦相手がブロックの上に移動する．
- ⋆ 対戦相手が試合続行不可能になる．

この条件で勝負がつかなかった場合は引き分けとなります．

3　ゲームで行える行動

3.1　getReady

自分のターンがくるまで待機します．自分のターンが来たとき，サーバから与えられる周辺情報を取得します．

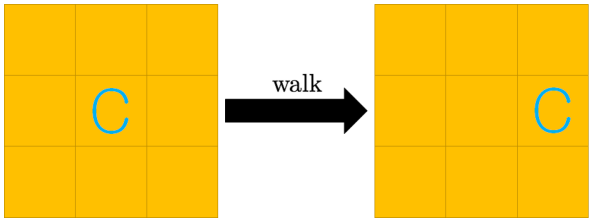
また，ターンの間で起動した場合，試合が続行出来ないのでターンの間は起動しない様にして下さい！

3.2　メソッド

メソッドは，“[動作][Up, Right, Left, Down のどれか]” という形で書かれています (例 :“walkRight”)．今回，載せる図などはすべて “Right” での動作です．ほかの “Up” などは適時読み替えて下さい．

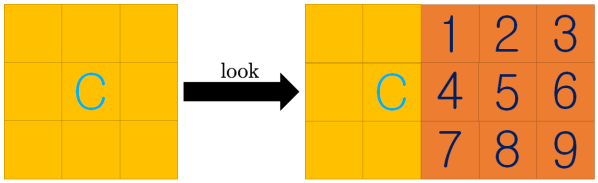
walk

指定された上下左右のどれかの方向に移動します．メソッド実行後に取得される周辺情報は移動後の周辺情報です．



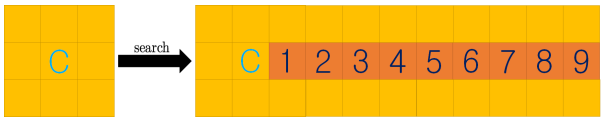
look

指定された上下左右のどれかの方向に対して，正方形状に 9 マスの情報を取得します (下記の図を参照)．メソッド実行後に取得される周辺情報は look で取得した情報です．周辺情報割当ての順番は図に割り振られている番号と一緒です．



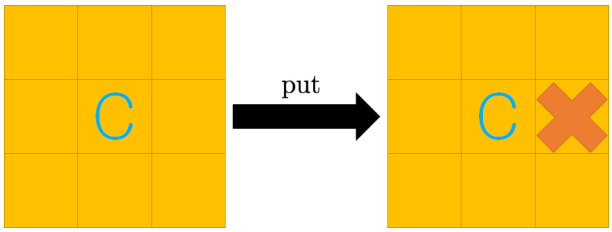
search

指定された上下左右のどれかの方向に対して，直線上に 9 マスの情報 (下記の図を参照) を取得します．メソッド実行後に取得される周辺情報は search で取得した情報です． 周辺情報割当ての順番は図に割り振られている番号と一緒です．



put

指定された上下左右のどれかの方向に対して，ブロックを配置します．メソッド実行後に取得される周辺情報は，ブロックをおいた後の周辺情報です．



4　周辺情報

サーバから送信される周辺情報については，具体的には以下の通りの様な仕様となっています．

(1) 10 桁の 0 から 3 までの数字です．
(2) 1 桁目には，ゲームの状態 (終了しているかどうか) を表しています．0 が終了状態で，1 が続行している状態です．
(3) 2 桁目から 10 桁目には対応する自分の周囲 9 マスの情報 (図 2 参照) が保存されています (5 番目が自分の現在居る位置です)．また，各桁に 0 から 3 までの数字が割り当てられて，以下に示す様な意味になります．

図 2: 周辺情報の割当図

表 1: 周辺の状態割り当て表

割り当てられている数値	状態
0	なにもなし
1	相手プレイヤがいる
2	ブロックがある
3	アイテムがある

以下に，実際の状態と値の対応表を書きます．

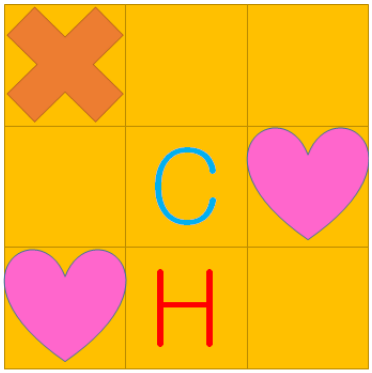


図 3: 周辺情報の例

表 2: 周辺情報の値

周辺情報の桁	値
1	1
2	2
3	0
4	0
5	0
6	0
7	3
8	3
9	1
10	0

5　具体的な通信の仕様

Warning!：この章は，自分で新しくライブラリを作成する人向けの章です．それ以外の人は読まなくても問題有りません．

別の言語などで，独自のライブラリを構築する場合，以下の仕様にのっとって作ればサーバとの通信が行えます．

(1) サーバとの接続を確立 (接続のポート番号は前述の値を参照して下さい)，
(2) サーバへチーム名を送信します (ここで，先頭 5 文字目以降は無視されます)．
(3) サーバが送信する “@” を受け取ります．
(4) “gr\r\n” をサーバに送信します (これを，本テキストでは getReady と呼んでいます)．
(5) サーバ側から送信される 10byte の文字列を受け取ります (これを，本テキストでは周辺情報と呼び，実装では value に格納しています)．例としては，“1000000000” となります．
(6) サーバにメソッドを表現する 2 文字とその末尾に “\r\n” を付加した文字列を送信します．表現の対応表は，以下の通りです．

表 3: メソッドの表現対応表

	Up	Right	Left	Down
walk	“wu”	“wr”	“wl”	“wd”
look	“lu”	“lr”	“ll”	“ld”
search	“su”	“sr”	“sl”	“sd”
put	“pu”	“pr”	“pl”	“pd”

(7) (5) と同様に周辺情報を受け取ります．
(8) サーバに “#\r\n” を送信します．
(9) 以降，周辺情報の最初の文字が “0” になるまで，(4) からここまでを繰り返す．

以下に概要図を示すので，これを利用して作成に役立てて下さい．

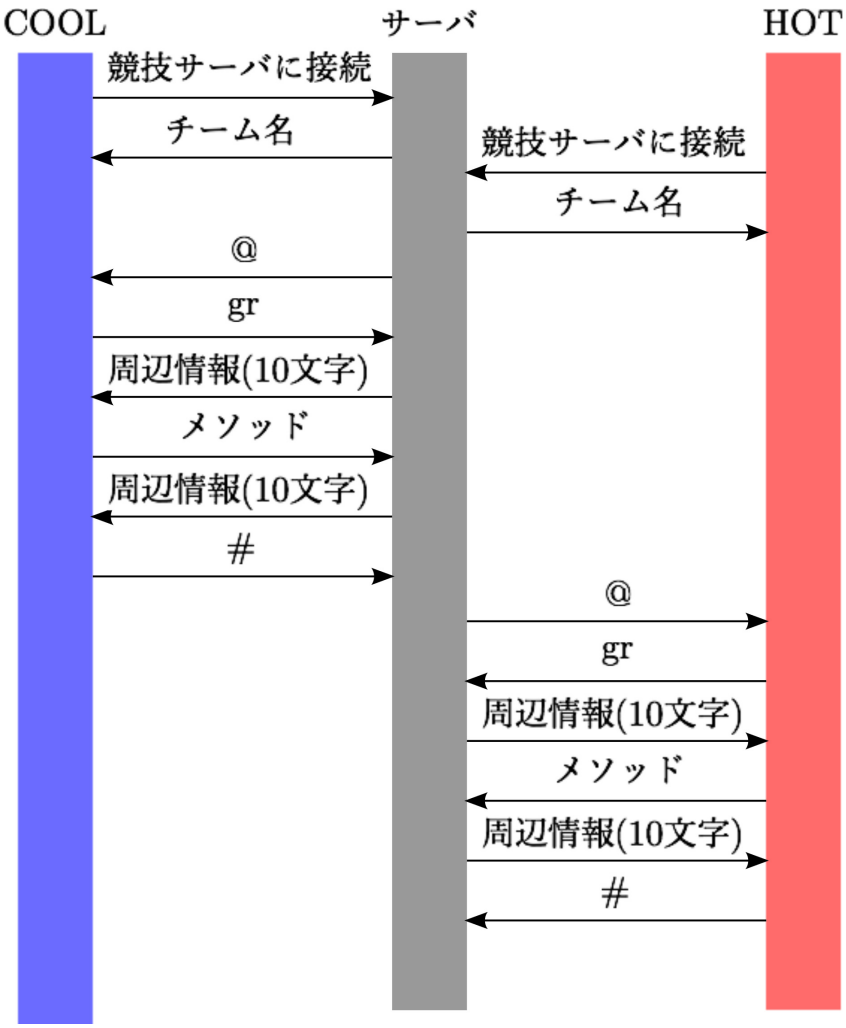

COOL
サーバ
HOT
競技サーバに接続
チーム名
競技サーバに接続
チーム名
@
gr
周辺情報(10文字)
メソッド
周辺情報(10文字)
#
@
gr
周辺情報(10文字)
メソッド
周辺情報(10文字)
#

図 4: 動作模式図