全国高校生プログラミングコンテスト

CHaser2009

ステップアップヒント２

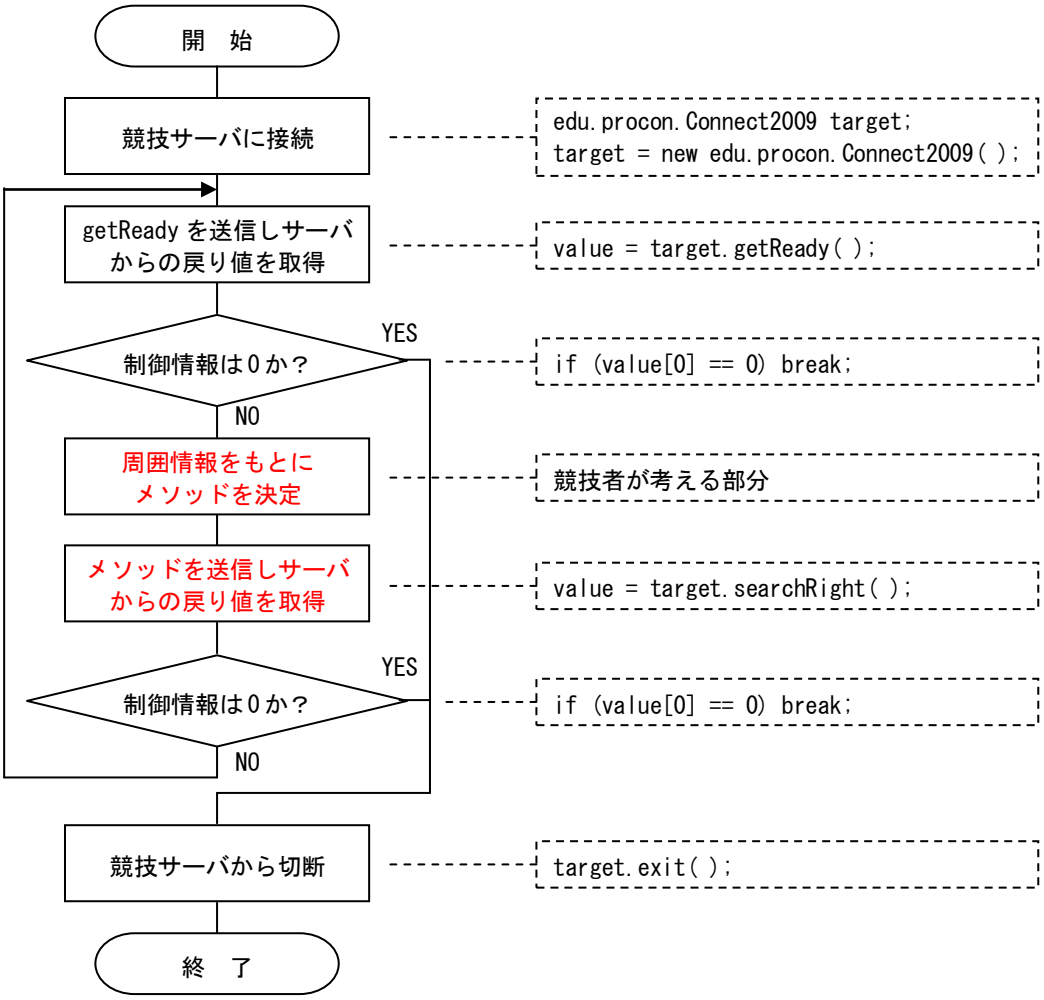
【プログラムの流れ】

みなさんが作るクライアントプログラムと競技サーバのやりとりは、getReady メソッドを送信して周囲情報を取得した後、動作メソッド（walk 系、look 系、search 系、put 系）を選んで送信し最後に周囲情報を取得という流れを繰り返します。

従って、クライアントプログラムは図 2 のフローチャートのようになります。

みなさんは、サーバから得た周囲情報をもとにメソッドを選んで送信する部分を考えます。

【フローチャート】　　**【プログラム】**サンプルプログラムの抜粋



【図 2】プログラムの大まかな流れ

【サーバからの戻り値】

getReady メソッドを送信した場合、サーバから送られてくる戻り値は、

制御情報（1）＋　周囲情報（9）＝　10 個の整数データ

となっています。戻り値は value という 10 個の領域を持つ配列に格納されます。

制御情報と周囲情報の値は表 1 と表 2 の通りです。

【表 1】制御情報の値

制御情報	
0	通信終了
1	通信続行

【表 2】周囲情報の値

周囲情報	
0	なし（床）
1	相手
2	ブロック
3	アイテム

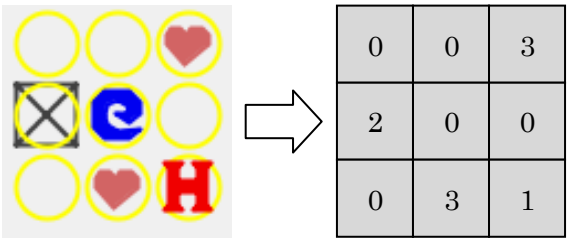
制御情報は 0 または 1 で、0 の場合、通信終了（試合終了）となり、exit メソッドでサーバから切断しプログラムを終了します。

周囲情報は、図 3 のように自分の周囲の 9 マスを左上から右下まで順番に値が格納されます。

例えば、図 4 の位置で getReady をした場合、表 2 より配列 value の値は図 5 のようになります。また、周囲 5（自分自身）は 0 となります。

周囲 1	周囲 2	周囲 3
周囲 4	周囲 5	周囲 6
周囲 7	周囲 8	周囲 9

【図 3】周囲情報の位置



0	0	3
2	0	0
0	3	1

【図 4】getReady をした場合の値

value[0]（制御情報）	1
〃 [1]（周囲 1）	0
〃 [2]（ 〃 2）	0
〃 [3]（ 〃 3）	3
〃 [4]（ 〃 4）	2
〃 [5]（ 〃 5）	0
〃 [6]（ 〃 6）	0
〃 [7]（ 〃 7）	0
〃 [8]（ 〃 8）	3
〃 [9]（ 〃 9）	1

【図 5】配列 value の値

※動作メソッド（walkUp など）を実行した後もサーバから戻り値が送信されます。
順序は競技仕様（CHaser2009rule.pdf）をご覧ください。

【サンプルプログラム２】モードを使ったプログラム

program フォルダの下に「sample2009_02」という名前のフォルダを作り、次のプログラムを入力します。プログラム名は「sample2009_02.java」です。

【動作】

・まず、ブロックにぶつかるまで上に移動します。
・次に右に移動し、ブロックの内側を時計回りに移動します。
・ターン制限（121 ターン）まで移動し、ゲームオーバーになります。
・mode（モード）という変数を作り、動作の種類を記憶させます。

【sample2009_02.java】

```
1  public class sample2009_02 {
2      public static void main(String[] args) {
3          int[] value;
4          value = new int[10];
5
6          int mode = 1;
7
8          /********** 競技サーバに接続する **********/
9          edu.procon.Connect2009 target;
10         target = new edu.procon.Connect2009("全プロ２" );
11
12         while (true) {
13             value = target.getReady( );
14             if (value[0] == 0) break;
15
16             /********** mode で分岐する **********/
17             switch (mode){
18                 case 1:  /* ブロックにぶつかるまで上に移動する */
19                     if(value[2] == 2){  /* 上にブロックがあったら */
20                         value = target.walkRight( );  /* 右に移動し */
21                         mode = 2;  /* mode を 2 に変更する */
22                     }
23                     else{  /* そうでなければ、上に移動する */
24                         value = target.walkUp( );
25                     }
26                     break;
27                 case 2:  /* ブロックにぶつかるまで右に移動する */
28                     if(value[6] == 2){
29                         value = target.walkDown( );
30                         mode = 3;
31                     }
32                     else{
33                         value = target.walkRight( );
34                     }
35                     break;
36                 case 3:  /* ブロックにぶつかるまで下に移動する */
```

```
37                      if(value[8] == 2){
38                          value = target.walkLeft( );
39                          mode = 4;
40                      }
41                      else{
42                          value = target.walkDown( );
43                      }
44                      break;
45                  case 4:  /* ブロックにぶつかるまで左に移動する */
46                      if(value[4] == 2){
47                          value = target.walkUp( );
48                          mode = 1;
49                      }
50                      else{
51                          value = target.walkLeft( );
52                      }
53                      break;
54              }
55              /********** 制御情報が 0 だったら終了する **********/
56              if(value[0] == 0) break;
57          }
58          /********** 競技サーバから切断する **********/
59          target.exit( );
60      }
61  }
```

【プログラム説明】

①変数の宣言

```
3       int[] value;
4       value = new int[10];
5
6       int mode = 1;
```

value は、サーバからの戻り値（制御情報 1＋周囲情報 9）を格納する配列です。

mode は、動作の種類を記憶させる変数です。mode を使うことで、どの動作をしているのかを管理することができます。サンプルプログラムは下表のように 4 つのモードを用意しています。この方式ならば、今後プログラムを発展させるときモードを追加するだけで様々な動作ができるようになります。

②競技サーバへの接続

```
8     /********** 競技サーバに接続する **********/
9     edu.procon.Connect2009 target;
10    target = new edu.procon.Connect2009("全プロ２" );
```

edu.procon.Connect2009 クラスを使って競技サーバに接続します。Connect2009 の引数で指定している文字列"全プロ２"はサーバに送信するチーム名で競技画面に表示されます。文字列は全角と半角の区別はなく 4 文字以内です。半角は全角に変換されます。また、5 文字以上の場合、先頭の 4 文字が有効です。

また、ネットワーク上のサーバに接続する場合は IP アドレスを指定します。サーバの IP アドレスが 192.168.101.50 の場合、10 行目を次のように変更します。

```
target = new edu.procon.Connect2009("全プロ２", "192.168.101.50" );
```

③getReady でサーバからの戻り値を得る

```
13    value = target.getReady( );
14    if (value[0] == 0) break;
```

getReady メソッドを送信するとサーバから戻り値が送信されてくるので、配列 value で受け取ります。value[0]に制御情報が格納されているので、0 だったら break し、while ループを抜けてプログラムを終了します。

④mode で分岐する

```
16    /********** mode で分岐する **********/
17    switch (mode){
18        case 1:  /* ブロックにぶつかるまで上に移動する */
19            if(value[2] == 2){  /* 上にブロックがあったら */
20                value = target.walkRight( );  /* 右に移動し */
21                mode = 2;  /* mode を 2 に変更する */
22            }
23            else{  /* そうでなければ、上に移動する */
24                value = target.walkUp( );
25            }
26            break;
27        case 2:  /* ブロックにぶつかるまで右に移動する */
              .
              .
              .
```

switch 文を使って mode の値で分岐処理を行います。mode = 1 の場合、ブロックにぶつかるまで上に移動するので、自分のすぐ上にブロックがあるかを if(value[2] == 2) で判断します。

もし、上がブロックだった場合は次に右に移動するので walkRight メソッドを送信し、mode を 2 にします。ブロックでなければ、walkUp メソッドを送信し上に移動します。同様に mode2 から 4 を作ります。

【表 3】モードの種類と動作

mode	動　作
1	ブロックにぶつかるまで上に移動する
2	ブロックにぶつかるまで右に移動する
3	ブロックにぶつかるまで下に移動する
4	ブロックにぶつかるまで左に移動する

⑤制御情報が 0 だったら終了する

```
55      /********** 制御情報が 0 だったら終了する **********/
56      if(value[0] == 0) break;
```

メソッドを送信するとサーバから戻り値が返ってくるので、終了の判定をします。

⑥競技サーバから切断する

```
58      /********** 競技サーバから切断する **********/
59      target.exit( );
```

exit メソッドで競技サーバから切断します。

※注意

このプログラムは、モードが切り替わる際、移動する方向にブロックがあるかのチェックをしていないので、ブロックにぶつかってしまう場合があります。

これを回避するためにプログラム中で判断を追加しなければなりません。どこにどのような判断を追加したらよいか考えてみてください。