全国高校生プログラミングコンテスト

CHaser2011

ステップアップヒント3

12 競技サーバを起動するバッチファイルの作成

今まで競技サーバを起動するには、バッチファイル（CHaser2011.bat）をダブルクリックしてコマンドを入力していましたが、競技を行うたびにコマンドを入力しなくても済むよう「サーバを起動するバッチファイル」を作ります。

program フォルダの下に、CHaser2011server.bat を作り、メモ帳などでバッチファイル 1 のように入力します（CHaser2011.bat をコピーして利用すると簡単です)。

マップファイルを複数作った場合は、バッチファイルをコピーしてマップファイル名の部分を変更して保存しておくと便利です。

さらに、様々なマップで競技を行うときはバッチファイル 2 の CHaser2011serverSelectMap.bat を利用すると便利です。これはバッチファイルを実行するとマップ名を入力してサーバを起動させる方法です。

ただし、いずれの場合も*バッチファイルはマップファイルと同じディレクトリ上に置く必要があります*。

バッチファイル 1（CHaser2011server.bat）

```
@echo off
echo ------------------------------------------------------------
echo　CHaser2011 開発環境（JDK6 Update24） 2011.7.4
echo ------------------------------------------------------------

set JAVA_HOME=C:¥Program Files¥Java¥jdk1.6.0_24
set PATH=%PATH%;%JAVA_HOME%¥bin;
set CLASSPATH=.;%JAVA_HOME%¥program¥edu2010b.jar

start /b java edu.procon.Server2010 -Fsample01.map
```

サーバを起動するコマンド
（sample01.map の場合）

バッチファイル２（CHaser2011serverSelectMap.bat）

```
@echo off
echo ------------------------------------------------------------------
echo　 CHaser2011 開発環境（JDK6 Update24） 2011.5.19
echo ------------------------------------------------------------------

set JAVA_HOME=C:¥Program Files¥Java¥jdk1.6.0_24
set PATH=%PATH%;%JAVA_HOME%¥bin;
set CLASSPATH=.;%JAVA_HOME%¥program¥edu2010b.jar
set /p MAP="マップ名を入力してください"

start /b java edu.procon.Server2010 -F%MAP%
```

入力されたマップ名を MAP へ格納する

サーバ起動のコマンドへマップ名を引渡す

13 サンプルプログラム3（アイテム回収と相手への処理を追加したプログラム）

①プログラムの保存先

program フォルダの下に「sample2011_03」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「sample2011_03.java」です。

②プログラムの動作

・クール専用のプログラムで、基本の動きは sample2011_02 と同じです。
・相手が上下左右にいた場合、その方向にブロックをのせます（put）。
・アイテムが上下左右にあった場合、その方向に移動（walk）し、アイテムを回収します。
・old_mode という変数を作り、1 つ前のモードを記憶させています。

③サンプルプログラム３（sample2011_03.java）

```
1  /*******************************************************
2     sample2011_03.java
3  *******************************************************/
4  public class sample2011_03 {
5     public static void main(String[] args) {
6        int[] value;
7        value = new int[10];
8        int mode = 1, // 現在のモード
9            old_mode = 1; // 前のモード
10
11       /********** 競技サーバに接続する **********/
12       edu.procon.Connect2010 target;
```

```
13      target = new edu.procon.Connect2010("サンプ 3 ", args[0] ,Integer.parseInt(args[1]));
14      while (true) {
15        value = target.getReady( );
16        if (value[0] == 0) break;
17
18        /********** 周囲（上下左右）に相手がいるかチェック **********/
19        if(value[2]==1 || value[4]==1 || value[6]==1 || value[8]==1){
20          mode = 90;    // mode を 90 に変更する
21        }
22
23        /********** 周囲（上下左右）にアイテムがあるかチェック **********/
24        if(value[2]==3 || value[4]==3 || value[6]==3 || value[8]==3){
25          old_mode = mode;    // 現在のモードを old_mode に保存する
26          mode = 20;    // mode を 20 に変更する
27        }
28
29        /********** mode の値で分岐する **********/
30        switch (mode) {
31          case 1:    // ブロック（壁）にぶつかるまで下に移動する
32            if(value[8] != 2){    // 下が壁でなければ、
33              value = target.walkDown( );    // 下に移動する
34            }else{
35              value = target.walkRight( );    // 下が壁ならば、右に移動し、
36              mode = 2;    // mode を 2 に変更する
37            }
38            break;
39
40          case 2:    // ブロック（壁）にぶつかるまで右に移動する
41            if(value[6] != 2){
42              value = target.walkRight( );
43            }else{
44              value = target.walkUp( );
45              mode = 3;
46            }
47            break;
48
49          case 3:    // ブロック（壁）にぶつかるまで上に移動する
50            if(value[2] != 2){
51              value = target.walkUp( );
52            }else{
53              value = target.walkLeft( );
54              mode = 4;
55            }
56            break;
57
58          case 4:    // ブロック（壁）にぶつかるまで左に移動する
59            if(value[4] != 2){
60              value = target.walkLeft( );
61            }else{
62              value = target.walkDown( );
64              mode = 1;
65            }
66            break;
67
68          case 20:    // 周囲にアイテムがあったら、取りに行く
69            // 上にアイテムがあったら、上に移動する
70            if(value[2] == 3) value = target.walkUp( );
71            // 左にアイテムがあったら、左に移動する
72            else if(value[4] == 3) value = target.walkLeft( );
```

```
73                  // 右にアイテムがあったら、右に移動する
74                  else if(value[6] == 3) value = target.walkRight( );
75                  // 下にアイテムがあったら、下に移動する
76                  else value = target.walkDown( );
77                  mode = old_mode;    // モードを元に戻す
78                  break;
79              case 90:    // 周囲に相手がいたら、ブロックをのせる
80                  // 上に相手がいたら、上にブロックをのせる
81                  if(value[2] == 1) value = target.putUp();
82                  // 左に相手がいたら、左にブロックをのせる
83                  else if(value[4] == 1) value = target.putLeft();
84                  // 右に相手がいたら、右にブロックをのせる
85                  else if(value[6] == 1) value = target.putRight();
86                  // 下に相手がいたら、下にブロックをのせる
87                  else value = target.putDown();
88                  break;
89          }
90          /********** 制御情報が 0 だったら while ループを終了する **********/
91          if(value[0] == 0) break;
92      }
93      /********** 競技サーバから切断する **********/
94      target.exit( );
95  }
96  }
```

⑤プログラム説明

・変数の宣言

```
8       int mode = 1, // 現在のモード
9           old_mode = 1; // 前のモード
```

動作の種類を記憶させる mode を宣言します。old_mode はアイテム回収処理をした後に元のモードに戻るために必要です。

・チーム名

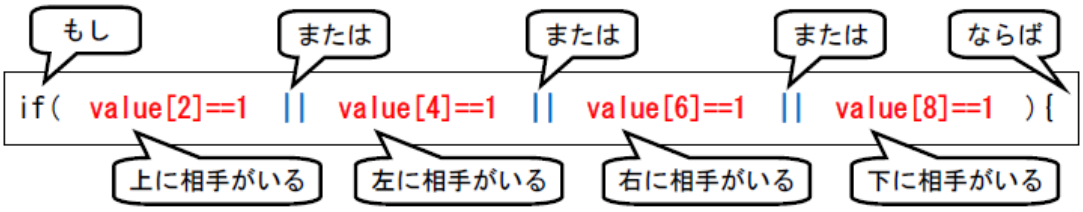
```
13      target = new edu.procon.Connect2010("サンプ３", args[0] ,Integer.parseInt(args[1]));
```

edu.procon.Connect2010 クラスを使って競技サーバに接続します。どのプログラムが対戦しているかわかるようにするため、チーム名を「サンプ３」にしました（変更しても構いません）。

・周囲（上下左右）に相手がいるかチェック

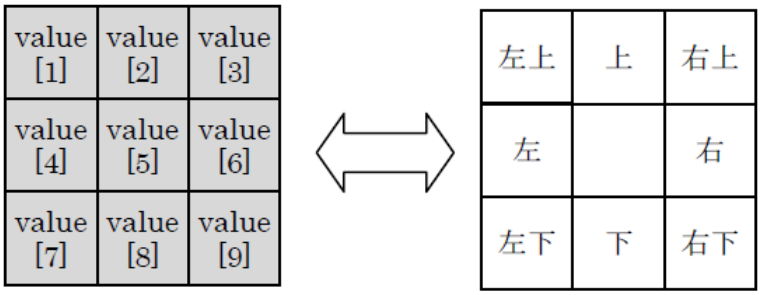
```
18          /********** 周囲（上下左右）に相手がいるかチェック **********/
19          if(value[2]==1 || value[4]==1 || value[6]==1 || value[8]==1){
20              mode = 90;    // mode を 90 に変更する
21          }
```

CHaser2010 は、相手にブロックをのせて（put）勝利することが一番の目的です。サーバから戻り値で相手は「1」なので、if 文を使って判断します。「value[2]==1」は上に相手がいるか？という意味で、「||」は論理和（OR）という意味で論理演算子といいます。



【図１】if 文の解説

図１をまとめると「もし、上下左右に相手がいるならば」という意味になります。
その場合は、mode を 90 にします。



【図２】value の戻り値と上下左右の関係

・mode=90 の処理

```
79          case 90:    // 周囲に相手がいたら、ブロックをのせる
80            // 上に相手がいたら、上にブロックをのせる
81            if(value[2] == 1) value = target.putUp();
82            // 左に相手がいたら、左にブロックをのせる
83            else if(value[4] == 1) value = target.putLeft();
84            // 右に相手がいたら、右にブロックをのせる
85            else if(value[6] == 1) value = target.putRight();
86            // 下に相手がいたら、下にブロックをのせる
87            else value = target.putDown();
88            break;
```

switch 文を使って mode の値で分岐処理を行います。mode=90 の場合、上下左右に相手がいるので、相手のいる方向にブロックをのせて（put）勝利となります。

・周囲（上下左右）にアイテムがあるかチェック

```
23        /********** 周囲（上下左右）にアイテムがあるかチェック **********/
24        if(value[2]==3 || value[4]==3 || value[6]==3 || value[8]==3){
25          old_mode = mode;    // 現在のモードを old_mode に保存する
26          mode = 20;    // mode を 20 に変更する
27        }
```

相手の処理と同様に if 文を使用して周囲にアイテム（3）があるか判断します。アイテムを回収したあと元のモードに戻るため old_mode に現在のモードを格納し、mode を 20 にします。

・mode=20 の処理

```
68              case 20:    // 周囲にアイテムがあったら、取りに行く
69                // 上にアイテムがあったら、上に移動する
70                if(value[2] == 3) value = target.walkUp( );
71                // 左にアイテムがあったら、左に移動する
72                else if(value[4] == 3) value = target.walkLeft( );
73                // 右にアイテムがあったら、右に移動する
74                else if(value[6] == 3) value = target.walkRight( );
75                // 下にアイテムがあったら、下に移動する
76                else value = target.walkDown( );
77                mode = old_mode;    // モードを元に戻す
78                break;
```

mode=20 の場合、上下左右にアイテムがあるので、その方向に移動（walk）します。アイテムを回収したあと、元のモードに戻るため old_mode の値を mode に格納します。
現在、使用しているモードは表 1 の通りです。

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
20（新規）	周囲（上下左右）のアイテムを回収する（walk）
90（新規）	周囲（上下左右）の相手にブロックをのせる（put）

14 発展

今回追加したプログラムは上下左右しか対応していないので、斜め方向（右上、右下、左上、左下）に対応させなければなりません。斜め方向は 1 つのターンでは処理できないので新たにモードを作るべきです。
特に、相手が斜め方向にいる場合、相手も自分がいることがわかるため慎重にメソッドを選択しなければなりません。また、プログラムがループしてしまわないよう注意してください。