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４

【サンプルプログラム４】mode と step を使ったプログラム

program フォルダの下に「sample2009_04」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_04.java」です。

【動作】

・スタート直後に上下左右をサーチし、アイテムのある方向に進むようにします。
・アイテムが上下左右になかった場合、今まで通り上に進みます。
・新たに step という変数を作り、それぞれの mode 内で step の数値に応じた動作ができるようにします。
・ステータス（アイテム数、ターン数、モード番号、ステップ番号）を画面に表示させます。

【sample2009_04.java】

```
1  public class sample2009_04 {
2
3      public static void main(String[] args) {
4          int[] value;
5          value = new int[10];
6
7          int mode = 10,     /* 現在のモード */
8              old_mode = 1,  /* 前のモード */
9              step = 1,      /* ステップ */
10             turn = 0,      /* ターン数 */
11             item = 0,      /* アイテム数 */
12             i;             /* カウンタ変数 */
13
14         /********** 競技サーバに接続する **********/
15         edu.procon.Connect2009 target;
16         target = new edu.procon.Connect2009("全プロ４" );
17
18         System.out.print("サーバと接続しました。");
19
20         while (true) {
21             value = target.getReady( );
22             if (value[0] == 0) break;
23
24             turn++; /* ターン数＋１ */
25             System.out.print("¥nturn=" + turn + "、mode=" + mode + "、step=" + step);
26
27             /********** 周囲に相手があるかチェック **********/
28             if(value[2]==1 || value[4]==1 || value[6]==1 || value[8]==1){
29                 old_mode = mode;
30                 mode = 90;
31             }
32
33             /********** 周囲にアイテムがあるかチェック **********/
```

```
34              if(value[2]==3 || value[4]==3 || value[6]==3 || value[8]==3){
35                  old_mode = mode;
36                  mode = 20;
37              }
38
39              /********** mode で分岐する **********/
40              switch (mode){
41                  case 1: /* 壁にぶつかるまで上に移動する */
42                      if(value[2] == 2){              /* 上にブロックがあったら */
43                          value = target.walkRight( ); /* 右に移動し */
44                          mode = 2;                   /* mode を 2 に変更する */
45                      }
46                      else{
47                          value = target.walkUp( );    /* そうでなければ、上に移動する */
48                      }
49                      break;
50                  case 2: /* ブロックにぶつかるまで右に移動する */
51                      if(value[6] == 2){
52                          value = target.walkDown( );
53                          mode = 3;
54                      }
55                      else{
56                          value = target.walkRight( );
57                      }
58                      break;
59                  case 3: /* ブロックにぶつかるまで下に移動する */
60                      if(value[8] == 2){
61                          value = target.walkLeft( );
62                          mode = 4;
63                      }
64                      else{
65                          value = target.walkDown( );
66                      }
67                      break;
68                  case 4: /* ブロックにぶつかるまで左に移動する */
69                      if(value[4] == 2){
70                          value = target.walkUp( );
71                          mode = 1;
72                      }
73                      else{
74                          value = target.walkLeft( );
75                      }
76                      break;
77
78                  case 10: /* スタート直後、４方向をサーチして移動する方向を決める */
79                      switch (step){
80                          case 1:
81                              value = target.searchUp();    /* 上をサーチする */
82                              System.out.print("：上サーチ");
83                              for(i=1; i<10; i++){
84                                  if(value[i] == 3){ /* 上方向にアイテムがあったら */
85                                      System.out.print("：上方向にアイテムあり。");
86                                      mode = 1;      /* 上に行くモードにして */
87                                      break;         /* ループを抜ける */
88                                  }
89                              }
90                              if(i==10){
91                                  step = 2;       /* アイテムがなかったら、ステップ 2 へ */
92                              }
93                              break;
94                          case 2:
95                              value = target.searchRight();    /* 右をサーチする */
96                              System.out.print("：右サーチ");
97                              for(i=1; i<10; i++){
```

```
98                              if(value[i] == 3){ /* 右方向にアイテムがあったら */
99                                  System.out.print("：右方向にアイテムあり。");
100                                 mode = 2;       /* 右に行くモードにして */
101                                 step = 1;       /* ステップを１に戻し */
102                                 break;          /* ループを抜ける */
103                             }
104                         }
105                         if(i==10){
106                             step = 3;       /* アイテムがなかったら、ステップ３へ */
107                         }
108                         break;
109                     case 3:
110                         value = target.searchDown();    /* 下をサーチする */
111                         System.out.print("：下サーチ");
112                         for(i=1; i<10; i++){
113                             if(value[i] == 3){ /* 下方向にアイテムがあったら */
114                                 System.out.print("：下方向にアイテムあり。");
115                                 mode = 3;       /* 下に行くモードにして */
116                                 step = 1;       /* ステップを１に戻し */
117                                 break;          /* ループを抜ける */
118                             }
119                         }
120                         if(i==10){
121                             step = 4;       /* アイテムがなかったら、ステップ３へ */
122                         }
123                         break;
124
125                     case 4:
126                         value = target.searchLeft();    /* 左をサーチする */
127                         System.out.print("：左サーチ");
128                         for(i=1; i<10; i++){
129                             if(value[i] == 3){ /* 左方向にアイテムがあったら */
130                                 System.out.print("：左方向にアイテムあり。");
131                                 mode = 4;       /* 左に行くモードにして */
132                                 step = 1;       /* ステップを１に戻し */
133                                 break;          /* ループを抜ける */
134                             }
135                         }
136                         if(i==10){
137                             step = 1;       /* アイテムがなかったら、ステップ１へ */
138                             mode = 1;       /* 上に行くモードにする */
139                         }
140                         break;
141                 }
142                 break;
143
144             case 20: /* 周囲にアイテムがあったら、取りに行く */
145                 /* 上にアイテムがあったら */
146                 if(value[2] == 3)       value = target.walkUp();
147                 /* 左にアイテムがあったら */
148                 else if(value[4] == 3) value = target.walkLeft();
149                 /* 右にアイテムがあったら */
150                 else if(value[6] == 3) value = target.walkRight();
151                 /* 下にアイテムがあったら */
152                 else if(value[8] == 3) value = target.walkDown();
153                 mode = old_mode;
154                 item++; /* アイテム数＋１ */
155                 System.out.print("：item=" + item); /* アイテム数の表示 */
156                 break;
157             case 90: /* 周囲に相手がいたら、put する */
158                 /* 上に相手がいたら */
159                 if(value[2] == 1)       value = target.putUp();
160                 /* 左に相手がいたら */
161                 else if(value[4] == 1) value = target.putLeft();
```

```
162                     /* 右に相手がいたら */
163                     else if(value[6] == 1) value = target.putRight();
164                     /* 下に相手がいたら */
165                     else if(value[8] == 1) value = target.putDown();
166                     mode = old_mode;
167                     break;
168
169             }
170             /********** 制御情報が 0 だったら終了する **********/
171             if(value[0] == 0) break;
172         }
173         /********** 競技サーバから切断する **********/
174         target.exit( );
175     }
176 }
```

【プログラム説明】

①変数の宣言

```
7           int mode = 10,     /* 現在のモード */
8               old_mode = 1,  /* 前のモード */
9               step = 1,      /* ステップ */
10              turn = 0,      /* ターン数 */
11              item = 0,      /* アイテム数 */
12              i;             /* カウンタ変数 */
```

スタート直後に上下左右をサーチしアイテムのある方向を探索する新しいモード（モード 10）を実行させるので、mode の初期値は 10 となります。

step の初期値は 1 とします。turn は現在のターン数、item は取得したアイテム数、i は for 文のカウンタ変数です。

②チーム名

```
16      target = new edu.procon.Connect2009("全プロ４" );
```

edu.procon.Connect2009 クラスを使って競技サーバに接続します。「サンプルプログラム４」なので、チーム名を「全プロ４」にしました（変更しても構いません）。

③接続成功のメッセージを表示

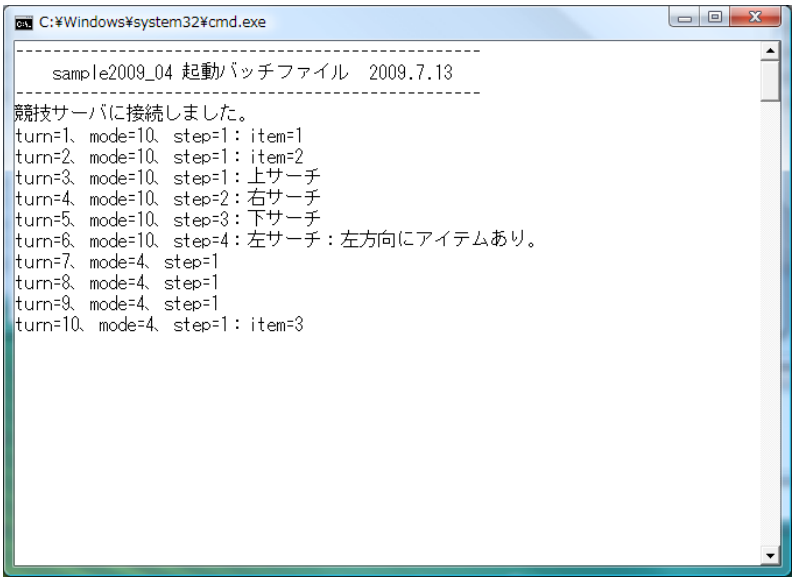
```
18      System.out.print("競技サーバに接続しました。");
```

競技サーバに接続したことを知らせるメッセージをコマンドプロンプトの画面に表示させます。

④ステータスの表示

```
24      turn++; /* ターン数＋１ */
25      System.out.print("¥nturn=" + turn + "、mode=" + mode + "、step=" + step);
```

ターン数をインクリメントし、ステータス（ターン数、モード番号、ステップ番号）を表示させます（画面 21）。print メソッドや println メソッドを使って様々なステータス（状態）を表示することでプログラムの動きを把握することができます。また、プログラムが正常に動かないときその原因を知ることもできます。



【画面 21】ステータス表示

⑤mode=10 の処理

```
78      case 10: /* スタート直後、４方向をサーチして移動する方向を決める */
79          switch (step){
80              case 1:
81                  value = target.searchUp();    /* 上をサーチする */
82                  System.out.print("：上サーチ");
83                  for(i=1; i<10; i++){
84                      if(value[i] == 3){ /* 上方向にアイテムがあったら */
85                          System.out.print("：上方向にアイテムあり。");
86                          mode = 1;       /* 上に行くモードにして */
87                          break;          /* ループを抜ける */
88                      }
89                  }
90                  if(i==10){
91                      step = 2;       /* アイテムがなかったら、ステップ２へ */
92                  }
93                  break;
```

モード 10 は、上下左右をサーチして移動する方向を決めるので、表 5 のように 4 つの手順（step）に分かれます。

まず上をサーチします。次に for 文で周囲情報 value[1]から value[9]にアイテム(3)があるか探索します。もし、アイテムが見つかったら「上方向にアイテムあり」とメッセージを出力した後、mode に上に移動する番号 1 をセットし、break 文で for 文を終了します。このとき、i の値はアイテムが見つかった番号になっています。

もし、アイテムが見つからなかった場合は、for 文が最後まで進んでいるので i は 10 になっています。これを使い if 文で i=10 ならばアイテムがないと判断し次のステップ（step=2）に進みます。

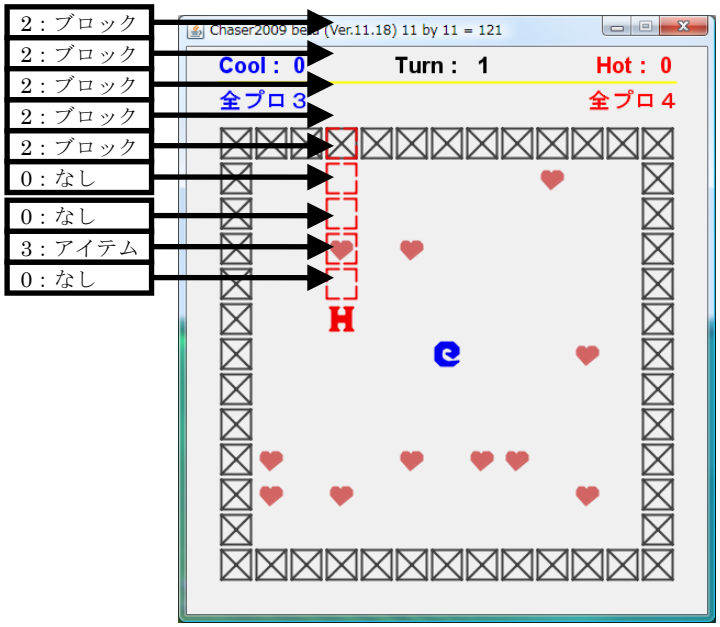
step2 から step4 も同様に行います。

【表 5】モードとステップの種類と動作

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	
10 （新規）	上下左右をサーチしアイテムのある方向を探索する	
	step=1	上をサーチし、アイテムがあれば mode=1 にする
	step=2	右をサーチし、アイテムがあれば mode=2 にする
	step=3	下をサーチし、アイテムがあれば mode=3 にする
	step=4	左をサーチし、アイテムがあれば mode=4 にする
20	周囲（上下左右）のアイテムを回収する	
90	周囲（上下左右）の相手に put する	

【ブロックの外側について】

画面 22 は上にサーチしたときの画面です。serchUp メソッド実行後、サーバからは 9 マス分の周囲情報が送信されますが、ブロックの外側は全てブロック（2）となります。従って、サーバからの戻り値は、「030022222」となります。



【画面 22】サーチがブロックを突き抜けた場合

【発展】

現在のプログラムは、上→右→下→左の順にサーチして 1 つでもアイテムがあったらその方向に進みますが、4 方向をサーチして 1 番アイテムが多い方向に進むなど変更してみましょう。