オセロ，対戦

```
 1  program Othello6;
 2  {$APPTYPE CONSOLE}
 3  uses SysUtils;
 4
 5  type
 6    TMasuNo = 0..99;                        // 盤のマスの番号
 7    TYouso  = (Soto,Kara,Kuro,Siro);        // 盤の要素（外，空，●，○）
 8    TBan    = array [TMasuNo] of TYouso;    // 盤（10 × 10）
 9    TPlayer = Kuro..Siro;                   // プレイヤー（要素名で表す）
10    TKosuu  = array [TPlayer] of Integer;   // 各プレイヤーのコマの個数
11    TMorC   = array [TPlayer] of Char;      // M (an) かC (omputer)
12    TMasus  = array [1..64] of TMasuNo;     // マスたち
13
14  var
15    Ban    : TBan;                          // 盤
16    Teban  : TPlayer;                       // 手番（今度コマを置く番の人）
17    Aite   : TPlayer;                       // 手番でない方の人
18    Kosuu  : TKosuu;                        // コマの個数
19    MorC   : TMorC;                         // プレイするのは誰か
20    OkeruMasu : TMasus;                     // コマを置けるマス全部
21    OkeruSuu  : Integer;                    // その個数
22
23  procedure SyokiSettei;
24            { 初期設定                          }
25            { 盤を開始状態にして，手番を●にする }
26    var
27      No  : TMasuNo;      // マスの番号
28    begin
29      for No := 0 to 99 do
30        begin
31          Ban[No] := Kara;
32        end;
33      for No := 0 to 9 do
34        begin
35          Ban[00+No] := Soto;
36          Ban[90+No] := Soto;
37          Ban[No*10+0] := Soto;
38          Ban[No*10+9] := Soto;
39        end;
40      Ban[44] := Kuro;
41      Ban[45] := Siro;
42      Ban[54] := Siro;
43      Ban[55] := Kuro;
44      Kosuu[Kuro] := 2;
45      Kosuu[Siro] := 2;
46      Teban := Kuro;
47      Aite  := Siro;
48    end; {SyokiSettei}
49
50  procedure WriteBan;
51            { 盤を書く }
52    var
53      No : TMasuNo;
54    begin
55      WriteLn;
56      Write(’’:22);
```

```
 57      for No := 0 to 9 do
 58        begin
 59          Write(No:2);
 60        end;
 61      WriteLn;
 62      for No := 0 to 99 do
 63        begin
 64          if No mod 10 = 0
 65            then Write(No:20, '  ');
 66          case Ban[No] of
 67            Soto : Write('■');
 68            Kara : Write('・');
 69            Kuro : Write('●');
 70            Siro : Write('○');
 71          end;
 72          if No mod 10 = 9
 73            then WriteLn;
 74        end;
 75      Write('':22);
 76      Write('  ●', Kosuu[Kuro]:2, '個  ');
 77      Write('  ○', Kosuu[Siro]:2, '個  ');
 78      WriteLn;
 79      if Teban = Kuro
 80        then WriteLn('●の番です')
 81        else WriteLn('○の番です');
 82    end; {WriteBan}
 83
 84  function  ReadMasuNo : TMasuNo;
 85            { どこに置くか問い合わせて，マスの番号読む     }
 86            { マスの番号でない数を入れたら再度問い合わせる }
 87    var
 88      No  : Integer;     // 読んだ数
 89    begin
 90      repeat
 91        Write('どこに置きますか [nn] ? ');
 92        ReadLn(No);
 93      until (0 <= No) and (No <= 99);
 94      ReadMasuNo := No;
 95    end; {ReadMasuNo}
 96
 97  function Uragaeseru(No : TMasuNo; Muki : Integer) : Boolean;
 98           { Ban[No] にコマを置いたとして，Muki の方向の相手のコマを裏返せるか}
 99           {          -11 -10  -9          }
100           { Muki は  -1   *   +1   の８方向 }
101           {          +9  +10 +11          }
102    var
103      Ok   : Boolean;
104      N    : TMasuNo;
105    begin
106      N := No+Muki;                              // Muki 方向に１マス進む
107      if Ban[N] <> Aite                          // 相手のコマでなければだめ
108        then Ok := False
109        else begin
110               repeat
111                 N := N+Muki;                    // Muki 方向に１マス進む
112               until Ban[N] <> Aite;             // 相手のコマでなくなるまで続ける
113               Ok := Ban[N] = Teban;             // それが自分のコマならＯＫ
114             end;
```

```
115     Uragaeseru := Ok;
116   end; {Uragaeseru}
117
118 function  Okeru(No : TMasuNo) : Boolean;
119           { Ban[No] にコマを置けるか            }
120   var
121     Ok : Boolean;
122   begin
123     Ok := False;
124     if Ban[No] = Kara
125       then Ok := True;
126     Okeru := Ok and (Uragaeseru(No,-11) or Uragaeseru(No,-10) or
127                      Uragaeseru(No,-9)  or Uragaeseru(No,-1)  or
128                      Uragaeseru(No,+1)  or Uragaeseru(No,+9)  or
129                      Uragaeseru(No,+10) or Uragaeseru(No,+11));
130   end; {Okeru}
131
132 procedure UragaesetaraUragaesu(No : TMasuNo; Muki : Integer);
133           { Muki 方向の相手のコマを裏返せたら裏返す }
134   var
135     N  : TMasuNo;
136   begin
137     if Uragaeseru(No,Muki)
138       then begin
139              N := No+Muki;              // １つ隣は相手のコマのはず
140              repeat
141                Ban[N] := Teban;         // 手番のコマに変える
142                Inc(Kosuu[Teban]);       // 手番のコマの数を増やす
143                Dec(Kosuu[Aite]);        // 相手のコマの数を減らす
144                N := N+Muki;             // 次のマスに進む
145              until Ban[N] = Teban;      // 手番のコマまで来たら終わる
146            end;
147   end; {UragaesetaraUragaesu}
148
149 procedure Oku(No : TMasuNo);
150           { Ban[No] に手番のコマを置く }
151   begin
152     Ban[No] := Teban;
153     UragaesetaraUragaesu(No,-11);
154     UragaesetaraUragaesu(No,-10);
155     UragaesetaraUragaesu(No,-9);
156     UragaesetaraUragaesu(No,-1);
157     UragaesetaraUragaesu(No,+1);
158     UragaesetaraUragaesu(No,+9);
159     UragaesetaraUragaesu(No,+10);
160     UragaesetaraUragaesu(No,+11);
161     Kosuu[Teban] := Kosuu[Teban]+1;
162   end; {Oku}
163
164 procedure ListUpOkeruMasu;
165           { コマを置けるマスをリストアップする }
166   var
167     No : TMasuNo;            // 調べるマスの番号
168     K  : Integer;            // OkeruMasu に入れたマスの個数
169   begin
170     // すべてのマスを調べて，コマを置けるマスを OkeruMasu に記憶する
171     // 記憶した個数を OkeruSuu に記憶する
172     K := 0;
```

```
173     for No := 11 to 88 do
174       begin
175         if Okeru(No)
176           then begin
177                  Inc(K);
178                  OkeruMasu[K] := No;
179                end;
180       end;
181     OkeruSuu := K;
182   end; {ListUpOkeruMasu}
183
184 function  SelectMasuNo : TMasuNo;
185           { コンピュータが置く場所を選ぶ          }
186           { コマを置けるますの中からランダムに選ぶ }
187   var
188     R  : Integer;
189     No : TMasuNo;
190   begin
191     ListUpOkeruMasu;                    // コマを置けるマスをリストアップする
192     R := Random(OkeruSuu)+1;            // １～OkeruSuu の乱数
193     No := OkeruMasu[R];                 // 乱数で選んだマス
194     WriteLn(No, ' に置きます');
195     SelectMasuNo := No;
196   end; {SelectMasuNo}
197
198 procedure OneMove;
199           { １手行う }
200   var
201     No   : TMasuNo;
202   begin
203     repeat
204       if MorC[Teban] = 'M'
205         then No := ReadMasuNo
206         else No := SelectMasuNo;
207     until Okeru(No);
208     Oku(No);
209   end; {OneMove}
210
211 procedure Hanten(var Player : TPlayer);
212           { 白黒反転する }
213   begin
214     case Player of
215       Kuro : Player := Siro;
216       Siro : Player := Kuro;
217     end;
218   end; {Hanten}
219
220 function  DokonimoOkenai : Boolean;
221           { どこにも置けないか }
222   var
223     N : TMasuNo;
224   begin
225     N := 10;                      // N=10 番目から始めて
226     repeat                        // 繰り返す
227       Inc(N);                     // 次のマスへ進む
228     until (N > 88) or Okeru(N);   // 置けるマスが見つかるか，すべて調べ尽くすまで
229     DokonimoOkenai := N > 88;
230   end; {DokonimoOkenai}
```

```
231
232 procedure Game;
233           { 1ゲーム行う }
234   var
235     Pass : Integer;               // 連続してパスした回数
236   begin
237     SyokiSettei;
238     WriteBan;
239     Pass := 0;
240     repeat
241       if DokonimoOkenai
242         then begin
243                WriteLn('どこにも置けません');
244                Inc(Pass);
245              end
246         else begin
247                OneMove;
248                Pass := 0;         // パスした回数をリセット
249              end;
250       Hanten(Teban);
251       Hanten(Aite);
252       WriteBan;
253     until (Pass >= 2) or                    // 2人が連続してパスをしたか
254           (Kosuu[Kuro] = 0) or              // ○だけになったか
255           (Kosuu[Siro] = 0) or              // ●だけになったか
256           (Kosuu[Kuro]+Kosuu[Siro] = 64);  // 埋め尽くしたら終り
257     if Kosuu[Kuro] = Kosuu[Siro]
258       then WriteLn('引き分け')
259       else if Kosuu[Kuro] > Kosuu[Siro]
260              then WriteLn('●の勝ち')
261              else WriteLn('○の勝ち');
262   end; {Game}
263
264 procedure ReadPlayer;
265           { 誰と誰が対戦するか，問い合わせて読む }
266   var
267     P    : TPlayer;
268   begin
269     WriteLn('プレイするのは人 (M) ですかコンピュータ (C) ですか');
270     for P := Kuro to Siro do
271       begin
272         repeat
273           case P of
274             Kuro : Write('●');
275             Siro : Write('○');
276           end;
277           Write('はどっち [M/C] ? ');
278           ReadLn(MorC[P]);
279           MorC[P] := UpCAse(MorC[P]);             // 小文字だったら大文字にする
280         until (MorC[P] = 'M') or (MorC[P] = 'C');
281       end;
282   end; {ReadPlayer}
283
284 var
285   YesNo : String;
286
287 begin {Main}
288   Randomize;
```

```
289    repeat
290      ReadPlayer;
291      Game;
292      WriteLn;
293      Write('もう1度しますか [y/n] ? ');
294      ReadLn(YesNo);
295    until YesNo = 'n';
296  end.
297
```

オセロ，勝率計算

```
 1  program Othello7;
 2  {$APPTYPE CONSOLE}
 3  uses SysUtils;
 4
 5  type
 6    TMasuNo = 0..99;                        // 盤のマスの番号
 7    TYouso  = (Soto,Kara,Kuro,Siro);        // 盤の要素（外，空，●，○）
 8    TBan    = array [TMasuNo] of TYouso;    // 盤（10 × 10）
 9    TPlayer = Kuro..Siro;                   // プレイヤー（要素名で表す）
10    TKosuu  = array [TPlayer] of Integer;   // 各プレイヤーのコマの個数
11    TMorC   = array [TPlayer] of Char;      // M (an) かC (omputer)
12    TMasus  = array [1..64] of TMasuNo;     // マスたち
13
14  var
15    Ban    : TBan;                          // 盤
16    Teban  : TPlayer;                       // 手番（今度コマを置く番の人）
17    Aite   : TPlayer;                       // 手番でない方の人
18    Kosuu  : TKosuu;                        // コマの個数
19    MorC   : TMorC;                         // プレイするのは誰か
20    OkeruMasu : TMasus;                     // コマを置けるマス全部
21    OkeruSuu  : Integer;                    // その個数
22    KuroKati  : Integer;
23    SiroKati  : Integer;
24    Hikiwake  : Integer;
25
26  procedure SyokiSettei;
27            { 初期設定                          }
28            { 盤を開始状態にして，手番を●にする }
29    var
30      No  : TMasuNo;      // マスの番号
31    begin
32      for No := 0 to 99 do
33        begin
34          Ban[No] := Kara;
35        end;
36      for No := 0 to 9 do
37        begin
38          Ban[00+No] := Soto;
39          Ban[90+No] := Soto;
40          Ban[No*10+0] := Soto;
41          Ban[No*10+9] := Soto;
42        end;
43      Ban[44] := Kuro;
44      Ban[45] := Siro;
45      Ban[54] := Siro;
46      Ban[55] := Kuro;
47      Kosuu[Kuro] := 2;
48      Kosuu[Siro] := 2;
49      Teban := Kuro;
50      Aite  := Siro;
51    end; {SyokiSettei}
52
53  function Uragaeseru(No : TMasuNo; Muki : Integer) : Boolean;
54           { Ban[No] にコマを置いたとして，Muki の方向の相手のコマを裏返せるか}
55           {          -11 -10  -9          }
56           { Muki は  -1   *   +1   の８方向 }
```

```
 57          {          +9  +10 +11          }
 58    var
 59      Ok   : Boolean;
 60      N    : TMasuNo;
 61    begin
 62      N := No+Muki;                              // Muki 方向に１マス進む
 63      if Ban[N] <> Aite                          // 相手のコマでなければだめ
 64        then Ok := False
 65        else begin
 66               repeat
 67                 N := N+Muki;                    // Muki 方向に１マス進む
 68               until Ban[N] <> Aite;             // 相手のコマでなくなるまで続ける
 69               Ok := Ban[N] = Teban;             // それが自分のコマならＯＫ
 70             end;
 71      Uragaeseru := Ok;
 72    end; {Uragaeseru}
 73
 74  function  Okeru(No : TMasuNo) : Boolean;
 75            { Ban[No] にコマを置けるか           }
 76    var
 77      Ok : Boolean;
 78    begin
 79      Ok := False;
 80      if Ban[No] = Kara
 81        then Ok := True;
 82      Okeru := Ok and (Uragaeseru(No,-11) or Uragaeseru(No,-10) or
 83                       Uragaeseru(No,-9)  or Uragaeseru(No,-1)  or
 84                       Uragaeseru(No,+1)  or Uragaeseru(No,+9)  or
 85                       Uragaeseru(No,+10) or Uragaeseru(No,+11));
 86    end; {Okeru}
 87
 88  procedure UragaesetaraUragaesu(No : TMasuNo; Muki : Integer);
 89            { Muki 方向の相手のコマを裏返せたら裏返す }
 90    var
 91      N  : TMasuNo;
 92    begin
 93      if Uragaeseru(No,Muki)
 94        then begin
 95               N := No+Muki;              // １つ隣は相手のコマのはず
 96               repeat
 97                 Ban[N] := Teban;         // 手番のコマに変える
 98                 Inc(Kosuu[Teban]);       // 手番のコマの数を増やす
 99                 Dec(Kosuu[Aite]);        // 相手のコマの数を減らす
100                 N := N+Muki;             // 次のマスに進む
101               until Ban[N] = Teban;      // 手番のコマまで来たら終わる
102             end;
103    end; {UragaesetaraUragaesu}
104
105  procedure Oku(No : TMasuNo);
106            { Ban[No] に手番のコマを置く }
107    begin
108      Ban[No] := Teban;
109      UragaesetaraUragaesu(No,-11);
110      UragaesetaraUragaesu(No,-10);
111      UragaesetaraUragaesu(No,-9);
112      UragaesetaraUragaesu(No,-1);
113      UragaesetaraUragaesu(No,+1);
114      UragaesetaraUragaesu(No,+9);
```

```
115      UragaesetaraUragaesu(No,+10);
116      UragaesetaraUragaesu(No,+11);
117      Kosuu[Teban] := Kosuu[Teban]+1;
118    end; {Oku}
119
120  procedure ListUpOkeruMasu;
121            { コマを置けるマスをリストアップする }
122    var
123      No : TMasuNo;
124    begin
125      OkeruSuu := 0;
126      for No := 11 to 88 do
127        begin
128          if Okeru(No)
129            then begin
130                   Inc(OkeruSuu);
131                   OkeruMasu[OkeruSuu] := No;
132                 end;
133        end;
134    end; {ListUpOkeruMasu}
135
136
137  function  SelectMasuNo : TMasuNo;
138            { コンピュータが置く場所を選ぶ          }
139            { コマを置けるますの中からランダムに選ぶ }
140    var
141      R  : Integer;
142      No : TMasuNo;
143    begin
144      ListUpOkeruMasu;                    // コマを置けるマスをリストアップする
145      R := Random(OkeruSuu)+1;            // １～OkeruSuu の乱数
146      No := OkeruMasu[R];                 // 乱数で選んだマス
147      SelectMasuNo := No;
148    end; {SelectMasuNo}
149
150  procedure OneMove;
151            { １手行う }
152    var
153      No   : TMasuNo;
154    begin
155      No := SelectMasuNo;
156      Oku(No);
157    end; {OneMove}
158
159  procedure Hanten(var Player : TPlayer);
160            { 白黒反転する }
161    begin
162      case Player of
163        Kuro : Player := Siro;
164        Siro : Player := Kuro;
165      end;
166    end; {Hanten}
167
168  function  DokonimoOkenai : Boolean;
169            { どこにも置けないか }
170    var
171      N : TMasuNo;
172    begin
```

```
173     N := 10;                        // N=10 番目から始めて
174     repeat                          // 繰り返す
175       Inc(N);                       // 次のマスへ進む
176     until (N > 88) or Okeru(N);     // 置けるマスが見つかるか，すべて調べ尽くすまで
177     DokonimoOkenai := N > 88;
178   end; {DokonimoOkenai}
179
180 procedure Game;
181           { １ゲーム行う }
182   var
183     Pass : Integer;                 // 連続してパスした回数
184   begin
185     SyokiSettei;
186     Pass := 0;
187     repeat
188       if DokonimoOkenai
189         then begin
190                Inc(Pass);
191              end
192         else begin
193                OneMove;
194                Pass := 0;           // パスした回数をリセット
195              end;
196       Hanten(Teban);
197       Hanten(Aite);
198     until (Pass >= 2) or                    // ２人が連続してパスをしたか
199           (Kosuu[Kuro] = 0) or              // ○だけになったか
200           (Kosuu[Siro] = 0) or              // ●だけになったか
201           (Kosuu[Kuro]+Kosuu[Siro] = 64);   // 埋め尽くしたら終り
202   end; {Game}
203
204 var
205   K : Integer;
206
207 begin {Main}
208   Randomize;
209   MorC[Kuro] := 'C';
210   MorC[Siro] := 'C';
211   KuroKati := 0;
212   SiroKati := 0;
213   Hikiwake := 0;
214   for K := 1 to 1000 do
215     begin
216       Game;
217       if Kosuu[Kuro] = Kosuu[Siro]
218         then Inc(Hikiwake)
219         else if Kosuu[Kuro] > Kosuu[Siro]
220                then Inc(KuroKati)
221                else Inc(SiroKati);
222       Write(' ●勝ち':8, KuroKati:4);
223       Write(' ○勝ち':8, SiroKati:4);
224       WriteLn(' 引分け':8, Hikiwake:4);
225     end;
226   ReadLn;
227 end.
228
```