

```

1 #include <iostream>
2 #include <string.h>
3 #include <string>
4 #include <ios>
5 #include <iomanip>
6 #include <random>
7
8 using namespace std;
9
10
11 //グローバル変数
12 int CNT;
13
14 //プレイヤー(キャラ)の情報格納用構造体
15 struct MEMBER{
16     char Name[16]; //プレイヤー名
17     int Type; //0なら人,1ならCOM
18     int Strength; //COMの強さ(=dep)
19 };
20
21 //プロトタイプ宣言
22 bool GetPlayer(int *pos, int END, int array[][32], int times, MEMBER member[3], int Pnum);
23 bool GetAI(int *pos, int END, int array[][32], int times, MEMBER member[3], int Pnum);
24 void GetOperate(int pos, int END, int dep);
25 void GetMemory(int array[][32], int n, MEMBER member[3]);
26 int random(int s, int e);
27
28 int main(void){
29
30     int pos = 0; //現状態
31     int endPoint; //ゲームの終点
32     int style; //プレイ人数選択用
33     int level; //レベル選択用
34     int turn = 0; //ターン数
35     int memory[3][32] = {}; //ゲームログ
36     int order; //プレイ順序
37
38     MEMBER member[3] = {}; //MEMBER構造体の宣言
39     MEMBER temporary[1] = {}; //入れ替えようの一時保管変数
40     MEMBER *p; //操作用ポインタ
41     MEMBER *tmp; //同上
42     p = member; //ポインタ割り当て
43     tmp = temporary; //同上
44
45     std::cout << '\n' << "##GameSetting##" << '\n';
46
47     //プレイ人数選択
48     std::cout << "Please select the number of players" << '\n';
49     std::cout << "1) Human : 0 | COM : 3 " << '\n';
50     std::cout << "2) Human : 1 | COM : 2 " << '\n';
51     std::cout << "3) Human : 2 | COM : 1 " << '\n';
52     std::cout << "4) Human : 3 | COM : 0 " << '\n';
53     while (1) {
54         std::cout << ">> ";
55         std::cin >> style;
56         if (style > 0 && style < 5) {
57             std::cout << '\n';
58             break;
59         }
60         std::cout << "!! Please choose a number from 1 to 4 !!" << '\n';
61     }
62
63     //プレイヤー名入力
64     for (size_t i = 0; i < style-1; i++) {
65         std::cout << "Please enter the name of Player_" << i+1 << '\n' << ">> ";
66         std::cin >> (p+i)->Name;

```

```

67     (p+i)->Type = 0;
68 }
69 //COM名入力
70 for (size_t i = style; i < 4; i++) {
71     std::cout << "Please enter the name of COM_" << i-(style-1) << '\n' << ">> ";
72     std::cin >> (p+i-1)->Name;
73     (p+i-1)->Type = 1;
74 }
75
76 //ゲームの終了点の選択
77 std::cout << '\n' << "Please select the end point of the game" << '\n' << ">> ";
78 std::cin >> endPoint;
79
80 //COMのレベル選択
81 std::cout << '\n';
82 for (size_t i = style; i < 4; i++) {
83     while (1) {
84         std::cout << "Please choose the strength of the " << (p+i-1)->Name << '\n';
85         std::cout << "1:weak | 2:middle | 3:strong " << '\n' << ">> ";
86         std::cin >> level;
87         if (level > 0 && level < 4) {
88             break;
89         }
90         std::cout << "!! Please choose a number from 1 to 3 !!" << '\n';
91     }
92     if (level == 1) {
93         //1手先読み
94         (p+i-1)->Strength = 2;
95     } else if (level == 2) {
96         //4手先読み
97         (p+i-1)->Strength = 8;
98         //depが偶数になるよう調整
99         if ((p+i-1)->Strength % 2 != 0) {
100             (p+i-1)->Strength += 1;
101         }
102     } else if (level == 3) {
103         //全手先読み
104         (p+i-1)->Strength = endPoint/2;
105         //depが偶数になるよう調整
106         if ((p+i-1)->Strength % 2 != 0) {
107             (p+i-1)->Strength += 1;
108         }
109     }
110 }
111
112 //プレイ順の選択
113 std::cout << '\n' << "Decide the order of play at random" << '\n';
114 order = random(0, 5);
115
116 //プレイ順に構造体を入れ替え
117 switch (order) {
118     case 0:
119         //0->1->2
120         break;
121
122     case 1:
123         //1->2->0
124         *tmp = *p;
125         *p = *(p+1);
126         *(p+1) = *tmp;
127         *tmp = *(p+1);
128         *(p+1) = *(p+2);
129         *(p+2) = *tmp;
130         break;
131
132     case 2:
133         //2->0->1
134         *tmp = *p;
135         *p = *(p+1);
136         *(p+1) = *tmp;
137         *tmp = *p;
138         *p = *(p+2);

```

```

139     *(p+2) = *tmp;
140     break;
141
142     case 3:
143         //1->0->2
144         *tmp = *p;
145         *p = *(p+1);
146         *(p+1) = *tmp;
147         break;
148
149     case 4:
150         //0->2->1
151         *tmp = *(p+1);
152         *(p+1) = *(p+2);
153         *(p+2) = *tmp;
154         break;
155
156     case 5:
157         //2->1->0
158         *tmp = *p;
159         *p = *(p+2);
160         *(p+2) = *tmp;
161         break;
162 }
163
164 // プレイ順の表示
165 std::cout << "1st : " << p->Name << '\n';
166 std::cout << "2nd : " << (p+1)->Name << '\n';
167 std::cout << "3rd : " << (p+2)->Name << '\n';
168
169 //メインループ開始
170 std::cout << '\n' << "##GameStart##" << '\n';
171 while (1){
172     //1番手
173     if (p->Type == 0) { // 構造体のType部で人かCOMか判断
174         if ( GetPlayer(&pos, endPoint, memory, turn, member, 0) ){
175             GetMemory(memory, turn, member);
176             break;
177         }
178     }else{
179         if ( GetAI(&pos, endPoint, memory, turn, member, 0) ){
180             GetMemory(memory, turn, member);
181             break;
182         }
183     }
184
185     //2番手
186     if ((p+1)->Type == 0) {
187         if ( GetPlayer(&pos, endPoint, memory, turn, member, 1) ){
188             GetMemory(memory, turn, member);
189             break;
190         }
191     }else{
192         if ( GetAI(&pos, endPoint, memory, turn, member, 1) ){
193             GetMemory(memory, turn, member);
194             break;
195         }
196     }
197
198     //3番手
199     if ((p+2)->Type == 0) {
200         if ( GetPlayer(&pos, endPoint, memory, turn, member, 2) ){
201             GetMemory(memory, turn, member);
202             break;
203         }
204     }else{
205         if ( GetAI(&pos, endPoint, memory, turn, member, 2) ){
206             GetMemory(memory, turn, member);
207             break;
208         }
209     }
210 }

```

```

211 // 現在状態の表示
212 GetMemory(memory, turn, member);
213 turn++;
214 }
215 return 0;
216 }
217
218 // プレイヤーの入力関数
219 bool GetPlayer(int *pos, int END, int array[][32], int times, MEMBER member[3], int Pnum){
220
221     int choice;
222
223     std::cout << '\n' << "Now number is " << *pos << '\n';
224
225     while (1){
226         if (*pos < END-1) {
227             std::cout << "Please push number (1 or 2) " << member[Pnum].Name << '\n';
228             std::cout << "-> ";
229             std::cin >> choice;
230             if (choice == 1 || choice == 2) {
231                 break;
232             }
233             std::cout << "!! You can only enter 1 or 2 !!" << '\n';
234         }else if(*pos == END-1){
235             std::cout << "Please push number 1 " << member[Pnum].Name << '\n';
236             std::cout << "-> ";
237             std::cin >> choice;
238             if (choice == 1) {
239                 break;
240             }
241             std::cout << "!! You can only enter 1 !!" << '\n';
242         }
243     }
244
245     // 現状態を更新しログを保存
246     *pos += choice;
247     array[Pnum][times] = *pos;
248
249     if (*pos >= END){
250         std::cout << member[Pnum].Name << " Lose..." << '\n';
251         return true;
252     }
253
254     return false;
255 }
256
257 // AIの選択関数
258 bool GetAI(int *pos, int END, int array[][32], int times, MEMBER member[3], int Pnum){
259
260     int one, two; // 1,2それぞれの勝率を格納
261     int operate; // 最終的な決定
262     int dep = member[Pnum].Strength; // COMの強さごとに決められた探索深度
263
264     std::cout << '\n' << member[Pnum].Name << " thiking now..." << '\n';
265     // 先読み
266     for (size_t i = 1; i < 3; i++) {
267         // AIが 1or2 を選んだ場合の勝ち数を計測
268         CNT = 0;
269         GetOperate(*pos+i, END, dep);
270         if (i == 1){
271             one = CNT;
272         }else if(i == 2){
273             two = CNT;
274         }
275     }
276
277     if (*pos > END-3) {
278         operate = 1;
279     }else if (*pos < END/2) {
280         operate = random(1, 2);
281     }else{
282         // 勝率が高い方を選択

```

```

283     if (one > two) {
284         operate = 1;
285     }else if (one < two){
286         operate = 2;
287     }else{
288         // どちらでも同じならランダムで決定
289         operate = random(1,2);
290     }
291 }
292
293 // 現状態を更新しログを保存
294 *pos += operate;
295 array[Pnum][times] = *pos;
296
297 std::cout << member[Pnum].Name << " selected " << operate << '\n';
298
299 if (*pos >= END){
300     std::cout << member[Pnum].Name << " Lose..." << '\n';
301     return true;
302 }
303 return false;
304 }
305
306 //AIのオペレーター関数
307 void GetOperate(int pos, int END, int dep){
308     // 再帰を繰り返し指定の深さまできたら終了
309     if (dep == 0) {
310         return;
311     }
312
313     if (dep % 2 == 0) {
314         // depが偶数=人間のターン
315         for (size_t i = 2; i < 5; i++) {
316             // posが終了点を超えていたら終了
317             if (pos + i >= END-2) {
318                 return;
319             }
320             GetOperate(pos+i, END, dep-1);
321         }
322     }else{
323         // depが奇数=AIのターン
324         for (size_t i = 1; i < 3; i++) {
325             if (pos + i >= END-2) {
326                 // 終了点の数値があればグローバル変数CNTをインクリメント
327                 if (pos + i == END-2) {
328                     CNT++;
329                 }
330                 return;
331             }
332             GetOperate(pos+i, END, dep-1);
333         }
334     }
335 }
336
337 // ゲームログの表示保存関数
338 void GetMemory(int array[][32], int n, MEMBER member[3]){
339
340     std::cout << '\n' << "#####";
341     for (size_t i = 0; i < n; i++) {
342         std::cout << "####";
343     }
344     std::cout << '\n';
345
346     for (size_t i = 0; i < 3; i++) {
347
348         if (i == 0) {
349             std::cout << std::left << std::setw(10) << member[0].Name << "...";
350         }else if (i == 1){
351             std::cout << std::left << std::setw(10) << member[1].Name << "...";
352         }else if (i == 2){
353             std::cout << std::left << std::setw(10) << member[2].Name << "...";
354         }

```

```

355
356     for (size_t j = 0; j < n+1; j++) {
357         if (array[i][j] != 0){
358             std::cout << std::right << std::setw(3) << array[i][j];
359         }else{
360             std::cout << std::right << std::setw(3) << "--";
361         }
362         if(j == n){
363             std::cout << " ";
364         }else{
365             std::cout << ",";
366         }
367     }
368     std::cout << '\n';
369 }
370
371 std::cout << "#####";
372 for (size_t i = 0; i < n; i++) {
373     std::cout << "####";
374 }
375 std::cout << '\n';
376 }
377
378 // 乱数生成
379 int random(int s, int e){
380     std::random_device rnd;          // 非決定的な乱数生成器を生成
381     std::mt19937 mt(rnd());          // メルセンヌ・ツイスタの32ビット版、引数は初期シード値
382     std::uniform_int_distribution<> r(s, e);          // 指定範囲の一様乱数
383     return r(mt);
384 }
385

```