

[| 일반 소스 공유 | >](#)

(공유) 포커[텍사스 홀덤] v1.8 by seori : 완성!!


seori 챗봇 마스터 🧡 1:1 채팅

2020.02.07. 04:04 조회 55

댓글 4 URL 복사 ⋮

첨부파일 1

텍사스 홀덤! 드디어 완성하였습니다!

(업데이트 내역)

v1.5 도움말을 추가하고 각종 오류들을 수정하였습니다.

v1.6 첫 베팅의 레이즈 최대 횟수를 줄였습니다.

v1.7 올인/사이드베팅 기능을 완성하였습니다.

v1.8 필요없는 스크립트들을 삭제하였습니다. (100줄 이상 줄어들었습니다. 나름 최적화..?)

주석을 달았습니다.

포커를 홀덤으로 다시 바꾸었고 일부 오타가 수정되었습니다.

```

1  const scriptName="홀덤.js"
2  var cards=["s 14", "s 2", "s 3", "s 4", "s 5", "s 6", "s 7", "s 8", "s 9", "s 10", "s 11", "s 12", "s 13", "d 14", "d 2", "d 3", "d 4", "d 5", "d 6", "d 7", "d 8", "d 9", "d 10", "d 11", "d 12", "d 13", "h 14", "h 2", "h 3", "h 4", "h 5", "h 6", "h 7", "h 8", "h 9", "h 10", "h 11", "h 12", "h 13", "c 14", "c 2", "c 3", "c 4", "c 5", "c 6", "c 7", "c 8", "c 9", "c 10", "c 11", "c 12", "c 13"]
3  //카드들입니다. s: 스페이드, d: 다이아몬드, h: 하트, c: 클로버. 숫자는 A가 등급이 가장 높으므로 14부터 시작합니다.
4  var poker=false; var Gstart=false;
5  var player=[]; var coin=[]; var pot=[]; //선수의 게임 진행 정보
6  var player2=[]; var chips=[]; var pot2=[]; //폴드한 선수의 정보 저장
7  var card1; var card2; var cardlist=[]; var common=[]; //카드관련
8  var basicchips=100; var basicseed=1; //시드관련
9  var isBet=false; var beting=[]; var isCheck=0; var status=0; var money=0; var call=0; var m=0;
10 var pc="";
11 var x="";
12 var R;
13 var isFirstgame=true;
14 var allsee="\u200b".repeat(500);
15 var nagagi=[];
16 var preChat=[];
17 var turnoff=false; var onoff=[]; //종료관련
18 var allin=[]; var side=[]; //올인관련
19
20 function shuffle(a) { //배열을 섞는 함수. 카드 셔플에 이용.
21   var j, x, i;
22   for (i = a.length - 1; i > 0; i--) {
23     j = Math.floor(Math.random() * (i + 1));
24     x = a[i]; a[i] = a[j]; a[j] = x;
25   }
26   return a;
27 }
28
29 function seol(moyang){ //게임이 끝나고 '카드모양'에 등급을 매길 때 이용됩니다.
30   return moyang.replace(/s/g,3).replace(/d/g,2).replace(/h/g,1).replace(/c/g,0);
31 }
32
33 function lee(sutja){ //족보 등급 계산을 위한 '숫자' 등급을 매기는 함수입니다.
34   var isStraight=0; var sut=[]; var ja=[];
35
36   for (var i=0;i<sutja.length;i++){
37     sut[i]=Number(sutja[i]);
38   }
39   if (sut[0]>10) return 0;
40   if (sut[1]>10) return 0;
41   if (sut[2]>10) return 0;
42   if (sut[3]>10) return 0;
43   if (sut[4]>10) return 0;
44   if (sut[5]>10) return 0;
45   if (sut[6]>10) return 0;
46   if (sut[7]>10) return 0;
47   if (sut[8]>10) return 0;
48   if (sut[9]>10) return 0;
49   if (sut[10]>10) return 0;
50   if (sut[11]>10) return 0;
51   if (sut[12]>10) return 0;
52   if (sut[13]>10) return 0;
53   if (sut[14]>10) return 0;
54   if (sut[15]>10) return 0;
55   if (sut[16]>10) return 0;
56   if (sut[17]>10) return 0;
57   if (sut[18]>10) return 0;
58   if (sut[19]>10) return 0;
59   if (sut[20]>10) return 0;
60   if (sut[21]>10) return 0;
61   if (sut[22]>10) return 0;
62   if (sut[23]>10) return 0;
63   if (sut[24]>10) return 0;
64   if (sut[25]>10) return 0;
65   if (sut[26]>10) return 0;
66   if (sut[27]>10) return 0;
67   if (sut[28]>10) return 0;
68   if (sut[29]>10) return 0;
69   if (sut[30]>10) return 0;
70   if (sut[31]>10) return 0;
71   if (sut[32]>10) return 0;
72   if (sut[33]>10) return 0;
73   if (sut[34]>10) return 0;
74   if (sut[35]>10) return 0;
75   if (sut[36]>10) return 0;
76   if (sut[37]>10) return 0;
77   if (sut[38]>10) return 0;
78   if (sut[39]>10) return 0;
79   if (sut[40]>10) return 0;
80   if (sut[41]>10) return 0;
81   if (sut[42]>10) return 0;
82   if (sut[43]>10) return 0;
83   if (sut[44]>10) return 0;
84   if (sut[45]>10) return 0;
85   if (sut[46]>10) return 0;
86   if (sut[47]>10) return 0;
87   if (sut[48]>10) return 0;
88   if (sut[49]>10) return 0;
89   if (sut[50]>10) return 0;
90   if (sut[51]>10) return 0;
91   if (sut[52]>10) return 0;
92   if (sut[53]>10) return 0;
93   if (sut[54]>10) return 0;
94   if (sut[55]>10) return 0;
95   if (sut[56]>10) return 0;
96   if (sut[57]>10) return 0;
97   if (sut[58]>10) return 0;
98   if (sut[59]>10) return 0;
99   if (sut[60]>10) return 0;
100  if (sut[61]>10) return 0;
101  if (sut[62]>10) return 0;
102  if (sut[63]>10) return 0;
103  if (sut[64]>10) return 0;
104  if (sut[65]>10) return 0;
105  if (sut[66]>10) return 0;
106  if (sut[67]>10) return 0;
107  if (sut[68]>10) return 0;
108  if (sut[69]>10) return 0;
109  if (sut[70]>10) return 0;
110  if (sut[71]>10) return 0;
111  if (sut[72]>10) return 0;
112  if (sut[73]>10) return 0;
113  if (sut[74]>10) return 0;
114  if (sut[75]>10) return 0;
115  if (sut[76]>10) return 0;
116  if (sut[77]>10) return 0;
117  if (sut[78]>10) return 0;
118  if (sut[79]>10) return 0;
119  if (sut[80]>10) return 0;
120  if (sut[81]>10) return 0;
121  if (sut[82]>10) return 0;
122  if (sut[83]>10) return 0;
123  if (sut[84]>10) return 0;
124  if (sut[85]>10) return 0;
125  if (sut[86]>10) return 0;
126  if (sut[87]>10) return 0;
127  if (sut[88]>10) return 0;
128  if (sut[89]>10) return 0;
129  if (sut[90]>10) return 0;
130  if (sut[91]>10) return 0;
131  if (sut[92]>10) return 0;
132  if (sut[93]>10) return 0;
133  if (sut[94]>10) return 0;
134  if (sut[95]>10) return 0;
135  if (sut[96]>10) return 0;
136  if (sut[97]>10) return 0;
137  if (sut[98]>10) return 0;
138  if (sut[99]>10) return 0;
139  if (sut[100]>10) return 0;
140  if (sut[101]>10) return 0;
141  if (sut[102]>10) return 0;
142  if (sut[103]>10) return 0;
143  if (sut[104]>10) return 0;
144  if (sut[105]>10) return 0;
145  if (sut[106]>10) return 0;
146  if (sut[107]>10) return 0;
147  if (sut[108]>10) return 0;
148  if (sut[109]>10) return 0;
149  if (sut[110]>10) return 0;
150  if (sut[111]>10) return 0;
151  if (sut[112]>10) return 0;
152  if (sut[113]>10) return 0;
153  if (sut[114]>10) return 0;
154  if (sut[115]>10) return 0;
155  if (sut[116]>10) return 0;
156  if (sut[117]>10) return 0;
157  if (sut[118]>10) return 0;
158  if (sut[119]>10) return 0;
159  if (sut[120]>10) return 0;
160  if (sut[121]>10) return 0;
161  if (sut[122]>10) return 0;
162  if (sut[123]>10) return 0;
163  if (sut[124]>10) return 0;
164  if (sut[125]>10) return 0;
165  if (sut[126]>10) return 0;
166  if (sut[127]>10) return 0;
167  if (sut[128]>10) return 0;
168  if (sut[129]>10) return 0;
169  if (sut[130]>10) return 0;
170  if (sut[131]>10) return 0;
171  if (sut[132]>10) return 0;
172  if (sut[133]>10) return 0;
173  if (sut[134]>10) return 0;
174  if (sut[135]>10) return 0;
175  if (sut[136]>10) return 0;
176  if (sut[137]>10) return 0;
177  if (sut[138]>10) return 0;
178  if (sut[139]>10) return 0;
179  if (sut[140]>10) return 0;
180  if (sut[141]>10) return 0;
181  if (sut[142]>10) return 0;
182  if (sut[143]>10) return 0;
183  if (sut[144]>10) return 0;
184  if (sut[145]>10) return 0;
185  if (sut[146]>10) return 0;
186  if (sut[147]>10) return 0;
187  if (sut[148]>10) return 0;
188  if (sut[149]>10) return 0;
189  if (sut[150]>10) return 0;
190  if (sut[151]>10) return 0;
191  if (sut[152]>10) return 0;
192  if (sut[153]>10) return 0;
193  if (sut[154]>10) return 0;
194  if (sut[155]>10) return 0;
195  if (sut[156]>10) return 0;
196  if (sut[157]>10) return 0;
197  if (sut[158]>10) return 0;
198  if (sut[159]>10) return 0;
199  if (sut[160]>10) return 0;
200  if (sut[161]>10) return 0;
201  if (sut[162]>10) return 0;
202  if (sut[163]>10) return 0;
203  if (sut[164]>10) return 0;
204  if (sut[165]>10) return 0;
205  if (sut[166]>10) return 0;
206  if (sut[167]>10) return 0;
207  if (sut[168]>10) return 0;
208  if (sut[169]>10) return 0;
209  if (sut[170]>10) return 0;
210  if (sut[171]>10) return 0;
211  if (sut[172]>10) return 0;
212  if (sut[173]>10) return 0;
213  if (sut[174]>10) return 0;
214  if (sut[175]>10) return 0;
215  if (sut[176]>10) return 0;
216  if (sut[177]>10) return 0;
217  if (sut[178]>10) return 0;
218  if (sut[179]>10) return 0;
219  if (sut[180]>10) return 0;
220  if (sut[181]>10) return 0;
221  if (sut[182]>10) return 0;
222  if (sut[183]>10) return 0;
223  if (sut[184]>10) return 0;
224  if (sut[185]>10) return 0;
225  if (sut[186]>10) return 0;
226  if (sut[187]>10) return 0;
227  if (sut[188]>10) return 0;
228  if (sut[189]>10) return 0;
229  if (sut[190]>10) return 0;
230  if (sut[191]>10) return 0;
231  if (sut[192]>10) return 0;
232  if (sut[193]>10) return 0;
233  if (sut[194]>10) return 0;
234  if (sut[195]>10) return 0;
235  if (sut[196]>10) return 0;
236  if (sut[197]>10) return 0;
237  if (sut[198]>10) return 0;
238  if (sut[199]>10) return 0;
239  if (sut[200]>10) return 0;
240  if (sut[201]>10) return 0;
241  if (sut[202]>10) return 0;
242  if (sut[203]>10) return 0;
243  if (sut[204]>10) return 0;
244  if (sut[205]>10) return 0;
245  if (sut[206]>10) return 0;
246  if (sut[207]>10) return 0;
247  if (sut[208]>10) return 0;
248  if (sut[209]>10) return 0;
249  if (sut[210]>10) return 0;
250  if (sut[211]>10) return 0;
251  if (sut[212]>10) return 0;
252  if (sut[213]>10) return 0;
253  if (sut[214]>10) return 0;
254  if (sut[215]>10) return 0;
255  if (sut[216]>10) return 0;
256  if (sut[217]>10) return 0;
257  if (sut[218]>10) return 0;
258  if (sut[219]>10) return 0;
259  if (sut[220]>10) return 0;
260  if (sut[221]>10) return 0;
261  if (sut[222]>10) return 0;
262  if (sut[223]>10) return 0;
263  if (sut[224]>10) return 0;
264  if (sut[225]>10) return 0;
265  if (sut[226]>10) return 0;
266  if (sut[227]>10) return 0;
267  if (sut[228]>10) return 0;
268  if (sut[229]>10) return 0;
269  if (sut[230]>10) return 0;
270  if (sut[231]>10) return 0;
271  if (sut[232]>10) return 0;
272  if (sut[233]>10) return 0;
273  if (sut[234]>10) return 0;
274  if (sut[235]>10) return 0;
275  if (sut[236]>10) return 0;
276  if (sut[237]>10) return 0;
277  if (sut[238]>10) return 0;
278  if (sut[239]>10) return 0;
279  if (sut[240]>10) return 0;
280  if (sut[241]>10) return 0;
281  if (sut[242]>10) return 0;
282  if (sut[243]>10) return 0;
283  if (sut[244]>10) return 0;
284  if (sut[245]>10) return 0;
285  if (sut[246]>10) return 0;
286  if (sut[247]>10) return 0;
287  if (sut[248]>10) return 0;
288  if (sut[249]>10) return 0;
289  if (sut[250]>10) return 0;
290  if (sut[251]>10) return 0;
291  if (sut[252]>10) return 0;
292  if (sut[253]>10) return 0;
293  if (sut[254]>10) return 0;
294  if (sut[255]>10) return 0;
295  if (sut[256]>10) return 0;
296  if (sut[257]>10) return 0;
297  if (sut[258]>10) return 0;
298  if (sut[259]>10) return 0;
299  if (sut[260]>10) return 0;
300  if (sut[261]>10) return 0;
301  if (sut[262]>10) return 0;
302  if (sut[263]>10) return 0;
303  if (sut[264]>10) return 0;
304  if (sut[265]>10) return 0;
305  if (sut[266]>10) return 0;
306  if (sut[267]>10) return 0;
307  if (sut[268]>10) return 0;
308  if (sut[269]>10) return 0;
309  if (sut[270]>10) return 0;
310  if (sut[271]>10) return 0;
311  if (sut[272]>10) return 0;
312  if (sut[273]>10) return 0;
313  if (sut[274]>10) return 0;
314  if (sut[275]>10) return 0;
315  if (sut[276]>10) return 0;
316  if (sut[277]>10) return 0;
317  if (sut[278]>10) return 0;
318  if (sut[279]>10) return 0;
319  if (sut[280]>10) return 0;
320  if (sut[281]>10) return 0;
321  if (sut[282]>10) return 0;
322  if (sut[283]>10) return 0;
323  if (sut[284]>10) return 0;
324  if (sut[285]>10) return 0;
325  if (sut[286]>10) return 0;
326  if (sut[287]>10) return 0;
327  if (sut[288]>10) return 0;
328  if (sut[289]>10) return 0;
329  if (sut[290]>10) return 0;
330  if (sut[291]>10) return 0;
331  if (sut[292]>10) return 0;
332  if (sut[293]>10) return 0;
333  if (sut[294]>10) return 0;
334  if (sut[295]>10) return 0;
335  if (sut[296]>10) return 0;
336  if (sut[297]>10) return 0;
337  if (sut[298]>10) return 0;
338  if (sut[299]>10) return 0;
339  if (sut[300]>10) return 0;
340  if (sut[301]>10) return 0;
341  if (sut[302]>10) return 0;
342  if (sut[303]>10) return 0;
343  if (sut[304]>10) return 0;
344  if (sut[305]>10) return 0;
345  if (sut[306]>10) return 0;
346  if (sut[307]>10) return 0;
347  if (sut[308]>10) return 0;
348  if (sut[309]>10) return 0;
349  if (sut[310]>10) return 0;
350  if (sut[311]>10) return 0;
351  if (sut[312]>10) return 0;
352  if (sut[313]>10) return 0;
353  if (sut[314]>10) return 0;
354  if (sut[315]>10) return 0;
355  if (sut[316]>10) return 0;
356  if (sut[317]>10) return 0;
357  if (sut[318]>10) return 0;
358  if (sut[319]>10) return 0;
359  if (sut[320]>10) return 0;
360  if (sut[321]>10) return 0;
361  if (sut[322]>10) return 0;
362  if (sut[323]>10) return 0;
363  if (sut[324]>10) return 0;
364  if (sut[325]>10) return 0;
365  if (sut[326]>10) return 0;
366  if (sut[327]>10) return 0;
367  if (sut[328]>10) return 0;
368  if (sut[329]>10) return 0;
369  if (sut[330]>10) return 0;
370  if (sut[331]>10) return 0;
371  if (sut[332]>10) return 0;
372  if (sut[333]>10) return 0;
373  if (sut[334]>10) return 0;
374  if (sut[335]>10) return 0;
375  if (sut[336]>10) return 0;
376  if (sut[337]>10) return 0;
377  if (sut[338]>10) return 0;
378  if (sut[339]>10) return 0;
379  if (sut[340]>10) return 0;
380  if (sut[341]>10) return 0;
381  if (sut[342]>10) return 0;
382  if (sut[343]>10) return 0;
383  if (sut[344]>10) return 0;
384  if (sut[345]>10) return 0;
385  if (sut[346]>10) return 0;
386  if (sut[347]>10) return 0;
387  if (sut[348]>10) return 0;
388  if (sut[349]>10) return 0;
389  if (sut[350]>10) return 0;
390  if (sut[351]>10) return 0;
391  if (sut[352]>10) return 0;
392  if (sut[353]>10) return 0;
393  if (sut[354]>10) return 0;
394  if (sut[355]>10) return 0;
395  if (sut[356]>10) return 0;
396  if (sut[357]>10) return 0;
397  if (sut[358]>10) return 0;
398  if (sut[359]>10) return 0;
399  if (sut[360]>10) return 0;
400  if (sut[361]>10) return 0;
401  if (sut[362]>10) return 0;
402  if (sut[363]>10) return 0;
403  if (sut[364]>10) return 0;
404  if (sut[365]>10) return 0;
405  if (sut[366]>10) return 0;
406  if (sut[367]>10) return 0;
407  if (sut[368]>10) return 0;
408  if (sut[369]>10) return 0;
409  if (sut[370]>10) return 0;
410  if (sut[371]>10) return 0;
411  if (sut[372]>10) return 0;
412  if (sut[373]>10) return 0;
413  if (sut[374]>10) return 0;
414  if (sut[375]>10) return 0;
415  if (sut[376]>10) return 0;
416  if (sut[377]>10) return 0;
417  if (sut[378]>10) return 0;
418  if (sut[379]>10) return 0;
419  if (sut[380]>10) return 0;
420  if (sut[381]>10) return 0;
421  if (sut[382]>10) return 0;
422  if (sut[383]>10) return 0;
423  if (sut[384]>10) return 0;
424  if (sut[385]>10) return 0;
425  if (sut[386]>10) return 0;
426  if (sut[387]>10) return 0;
427  if (sut[388]>10) return 0;
428  if (sut[389]>10) return 0;
429  if (sut[390]>10) return 0;
430  if (sut[391]>10) return 0;
431  if (sut[392]>10) return 0;
432  if (sut[393]>10) return 0;
433  if (sut[394]>10) return 0;
434  if (sut[395]>10) return 0;
435  if (sut[396]>10) return 0;
436  if (sut[397]>10) return 0;
437  if (sut[398]>10) return 0;
438  if (sut[399]>10) return 0;
439  if (sut[400]>10) return 0;
440  if (sut[401]>10) return 0;
441  if (sut[402]>10) return 0;
442  if (sut[403]>10) return 0;
443  if (sut[404]>10) return 0;
444  if (sut[405]>10) return 0;
445  if (sut[406]>10) return 0;
446  if (sut[407]>10) return 0;
447  if (sut[408]>10) return 0;
448  if (sut[409]>10) return 0;
449  if (sut[410]>10) return 0;
450  if (sut[411]>10) return 0;
451  if (sut[412]>10) return 0;
452  if (sut[413]>10) return 0;
453  if (sut[414]>10) return 0;
454  if (sut[415]>10) return 0;
455  if (sut[416]>10) return 0;
456  if (sut[417]>10) return 0;
457  if (sut[418]>10) return 0;
458  if (sut[419]>10) return 0;
459  if (sut[420]>10) return 0;
460  if (sut[421]>10) return 0;
461  if (sut[422]>10) return 0;
462  if (sut[423]>10) return 0;
463  if (sut[424]>10) return 0;
464  if (sut[425]>10) return 0;
465  if (sut[426]>10) return 0;
466  if (sut[427]>10) return 0;
467  if (sut[428]>10) return 0;
468  if (sut[429]>10) return 0;
469  if (sut[430]>10) return 0;
470  if (sut[431]>10) return 0;
471  if (sut[432]>10) return 0;
472  if (sut[433]>10) return 0;
473  if (sut[434]>10) return 0;
474  if (sut[435]>10) return 0;
475  if (sut[436]>10) return 0;
476  if (sut[437]>10) return 0;
477  if (sut[438]>10) return 0;
478  if (sut[439]>10) return 0;
479  if (sut[440]>10) return 0;
480  if (sut[441]>10) return 0;
481  if (sut[442]>10) return 0;
482  if (sut[443]>10) return 0;
483  if (sut[444]>10) return 0;
484  if (sut[445]>10) return 0;
485  if (sut[446]>10) return 0;
486  if (sut[447]>10) return 0;
487  if (sut[448]>10) return 0;
488  if (sut[449]>10) return 0;
489  if (sut[450]>10) return 0;
490  if (sut[451]>10) return 0;
491  if (sut[452]>10) return 0;
492  if (sut[453]>10) return 0;
493  if (sut[454]>10) return 0;
494  if (sut[455]>10) return 0;
495  if (sut[456]>10) return 0;
496  if (sut[457]>10) return 0;
497  if (sut[458]>10) return 0;
498  if (sut[459]>10) return 0;
499  if (sut[460]>10) return 0;
500  if (sut[461]>10) return 0;
501  if (sut[462]>10) return 0;
502  if (sut[463]>10) return 0;
503  if (sut[464]>10) return 0;
504  if (sut[465]>10) return 0;
505  if (sut[466]>10) return 0;
506  if (sut[467]>10) return 0;
507  if (sut[468]>10) return 0;
508  if (sut[469]>10) return 0;
509  if (sut[470]>10) return 0;
510  if (sut[471]>10) return 0;
511  if (sut[472]>10) return 0;
512  if (sut[473]>10) return 0;
513  if (sut[474]>10) return 0;
514  if (sut[475]>10) return 0;
515  if (sut[476]>10) return 0;
516  if (sut[477]>10) return 0;
517  if (sut[478]>10) return 0;
518  if (sut[479]>10) return 0;
519  if (sut[480]>10) return 0;
520  if (sut[481]>10) return 0;
521  if (sut[482]>10) return 0;
522  if (sut[483]>10) return 0;
523  if (sut[484]>10) return 0;
524  if (sut[485]>10) return 0;
525  if (sut[486]>10) return 0;
526  if (sut[487]>10) return 0;
527  if (sut[488]>10) return 0;
528  if (sut[489]>10) return 0;
529  if (sut[490]>10) return 0;
530  if (sut[491]>10) return 0;

```

```

38     sut.push(Number(sutja[1]));
39     ja.push(Number(sutja[i]));
40 }
41 for (var i=0;i<sutja.filter(e => e == 14).length;i++){
42     sutja.push(1); sut.push(1); ja.push(1);
43 } //추후 백스트레이트 계산을 위해 배열에 A(14)가 포함된 경우 1을 추가해줍니다.
44
45 for (var i=14;i>1;i--) if (sutja.filter(e => e == i).length==4) return [9, i]; //포카드
46
47 for (var i=14;i>1;i--){
48     if (sutja.filter(e => e == i).length==3){
49         for (k=0;k<3;k++) sut.splice(sut.indexOf(i),1);
50         for (var j=14;j>1;j--) if (sut.filter(e => e == j).length>=2) return [8,i+0.01*j];
51     }
52 } //플하우스
53
54 for (var i=14;i>9;i--){
55     if (sutja.filter(e => e == i).length>=1){
56         isStraight+=1;
57         if (isStraight==5) return [6,0];
58     } else {
59         isStraight=0;
60         break;
61     }
62 } //마운틴
63
64 for (var i=5;i>0;i--){
65     if (sutja.filter(e => e == i).length>=1){
66         isStraight+=1;
67         if (isStraight==5) return [5,0];
68     } else {
69         isStraight=0;
70         break;
71     }
72 } //백스트레이트
73
74 for (var p=13; p>5; p--){
75     for (var i=0;i<5;i++){
76         if (sutja.filter(e => e == (p-i)).length >=1 ){
77             isStraight+=1;
78             if (isStraight==5) return [4,p];
79         } else break;
80     }isStraight=0;
81 } //스트레이트
82
83 for (var i=14;i>1;i--){ if (sutja.filter(e => e == i).length==3) return [3,i]; } //트리플
84
85 for (var i=14;i>1;i--){
86     if (sutja.filter(e => e == i).length==2){
87         for (k=0;k<2;k++) ja.splice(ja.indexOf(i),1);
88         for (var j=i-1;j>1;j--) if (ja.filter(e => e == j).length==2) return [2,i+0.01*j];
89         return [1,i]; //원페어
90     }
91 }
92
93 var ans=[0,Math.max.apply(null, sutja)];
94 return ans; //탐(노페어)
95 }
96
97 function grade(CARD){ //모양 등급인 '플러시'의 경우를 생각하여 족보의 진짜 등급을 계산합니다.
98     var Flushshape; var shape=[]; var isFlush=false;
99     var num=[]; var card=[]; var Realgrade=[[[]];
100
101     for (var i=0; i<CARD.length; i++){
102         shape.push(seol(CARD[i].split(" ")[0]));
103         num.push(CARD[i].split(" ")[1]);
104         card.push(Number(num[i])*10+Number(shape[i]));
105     }
106

```

```

107     for (var i=0; i<4; i++){
108         if (shape.filter(e => e == i).length>=5){
109             isFlush=true;
110             Flushshape=i;
111             break;
112         }
113     } //족보가 플러시인지 아닌지 판별합니다.
114
115     if (isFlush==true){ //플러시일 경우
116         for(var i=0;i<shape.length; i++){
117             if (shape[i]!=Flushshape){
118                 card.splice(i,1);
119                 num.splice(i,1);
120             }
121         }
122         if (lee(num)[0]==6) Realgrade[0].push(12); //로티플
123         else if (lee(num)[0]==5) Realgrade[0].push(11); //백티플
124         else if (lee(num)[0]==4) Realgrade[0].push(10); //스티플
125         else Realgrade[0].push(7); //플러시
126     } else Realgrade[0].push(lee(num)[0]); //플러시가 아닐 경우 숫자 등급을 그대로 반환합니다.
127
128     num.sort((a,b)=>b-a);
129     var nnum=0;
130     card.sort((a,b)=>b-a);
131     var sshape=0;
132     for (var i=0;i<num.length;i++){
133         nnum+=Math.pow(0.01,i)*num[i];
134         sshape+=Math.pow(0.001,i)*card[i];
135     }
136     Realgrade[0].push(lee(num)[1]);
137     Realgrade.push(nnum);
138     Realgrade.push(sshape);
139     return Realgrade;
140     /* [[족보 등급, 등급에 해당하는 숫자], 하이 카드, 하이 카드에 따른 모양 등급]으로 전체 등급을 반환
141     등급이 무승부가 날 시 우승자를 판별하기 위함입니다.*/
142 }
143
144 function sh(shs){ //내용 출력을 위한 함수입니다.
145     return shs.replace(/s /g,"♠").replace(/d /g,"♦").replace(/h /g,"♥").replace(/c /g,"♣").
146 }
147
148 function outgrade(put){ //족보 등급 출력을 위한 함수입니다.
149     var first=Number(put[1]).toFixed(0);
150     var second=Math.round((Number(put[1])-first)*100);
151     var third=Number(put[0]).toFixed(0).replace("12", "로티플").replace("11", "백티플").replace("10", "스티플");
152
153     var ans=first+", "+second+" "+third;
154
155     if (first==0) return third;
156     else if (second==0) return first+" "+third;
157     else return ans;
158 }
159
160 function response(room, msg, sender, isGroupChat, replier, ImageDB, packageName, threadId){
161
162     if (preChat[room]==msg) return 0; //도배(따닥챗)방지
163     preChat[room]=msg;
164
165     var cmd=msg.split(" ")[0];
166     var data=msg.replace(cmd+" ", "");
167     if (msg=="/홀덤") replier.reply("텍사스 홀덤 도움말\n(명령어, 룰 설명)+"allsee+"\n\n[명령어]\n");
168     if (msg=="/홀덤 생성" && isGroupChat==true){
169         if (poker==true) replier.reply("이미 생성된 방이 있습니다.");
170         else {
171             poker=true;
172             R=room;
173             replier.reply("방이 생성되었습니다.\n저에게 관톡으로 '/홀덤 참가'라고 보내주시면 참가가능합니다.");
174         }
175     } //방을 만듭니다.

```

```

176
177 if (Gstart==false && poker==true && cmd=="/칩" && isNaN(data)==false){
178     if ((data-Math.floor(data))==0){
179         if (50<=data && data<=1000){
180             basicchips=Number(data);
181             replier.reply("기본 칩이 "+data+"개로 설정되었습니다.");
182             coin.fill(basicchips);
183             chips.fill(basicchips);
184             if (basicseed/basicchips<0.002){
185                 basicseed=Math.cell(basicchips*0.002);
186                 replier.reply("최저시드("+basicseed+"개)로 자동 설정되었습니다.");
187             } else if (basicseed/basicchips>0.05){
188                 basicseed=Math.floor(basicchips*0.05);
189                 replier.reply("최고시드("+basicseed+"개)로 자동 설정되었습니다.");
190             }
191         } else replier.reply("50에서 1000 사이의 자연수를 입력해주세요.");
192     } else replier.reply("자연수를 입력해주세요.");
193 } //게임 시작전에 기본적으로 가지게 되는 칩 개수를 설정합니다.
194
195 if (Gstart == false && poker==true && cmd=="/시드" && isNaN(data)==false){
196     if ((data-Math.floor(data))==0){
197         basicseed=Number(data);
198         if (basicseed/basicchips<0.002){
199             basicseed=Math.cell(basicchips*0.002);
200             replier.reply("최저시드("+basicseed+"개)로 자동 설정되었습니다.");
201         } else if (basicseed/basicchips>0.05){
202             basicseed=Math.floor(basicchips*0.05);
203             replier.reply("최고시드("+basicseed+"개)로 자동 설정되었습니다.");
204         } else replier.reply("시드가 "+data+"개로 설정되었습니다.");
205     } else replier.reply("자연수를 입력해주세요.");
206 } //시드금액을 설정합니다. 방장이 자동 SB가 됩니다.
207
208 if (msg=="/홀덤 참가"){
209     if (Gstart==false){
210         if (poker==true){
211             if (room==sender){
212                 if (player.length<5){
213                     if (player.indexOf(sender)===-1){
214                         player.push(sender);
215                         player2.push(sender);
216                         coin.push(basicchips);
217                         chips.push(basicchips);
218
219                         pot.push(0);
220                         pot2.push(0);
221                         cardlist.push([]);
222                         replier.reply("정상 참가 처리 되었습니다.");
223                         replier.reply(R, "현재 참가자: "+player);
224                     } else replier.reply(sender + "님은 이미 참가하셨습니다.");
225                 } else replier.reply("최대 인원은 5명입니다.");
226             } else replier.reply("관적으로 보내주셔야 참가가 인정됩니다.");
227         } else replier.reply("아직 방이 만들어지지 않았습니다.");
228     } else replier.reply("이미 게임이 진행중입니다.");
229 } //게임에 참가합니다.
230
231 if (msg=="/홀덤 시작" && room==R && player.indexOf(sender)!=-1){ // 게임을 시작합니다.
232     if (player.length>=2){
233         if (poker==true){
234             if (Gstart==false){
235                 Gstart=true;
236                 if (isFirstgame==false && basicseed<Math.floor(0.05*basicchips)){
237                     replier.reply("시드를 올리시겠습니까?\n업 (숫자) 또는 /킵 으로 설정 가능");
238                     while (isFirstgame==false){}
239                 } /*두 번째 이상의 게임부터 시드를 올릴 지 말 지 선택에 대한 질문입니다.
240                     이미 최고시드일 경우에는 묻지 않습니다.*/
241                 shuffle(cards); //카드 섞기
242                 replier.reply("게임을 시작합니다.");
243                 for(var i=0; i<=player.length-1; i++){
244                     card1=cards[cards.length-1];

```

```

245         cards.pop();
246         card2=cards[cards.length-1];
247         cards.pop();
248         cardlist[i].push(card1);
249         cardlist[i].push(card2);
250         var ans="핸드 카드를 받았습니다.\n"+cardlist[i]+" \n현재 등급: "+outgrade;
251         replier.reply(player[i],sh(ans));
252     } //처음에 받게 되는 핸드 카드를 받습니다.
253     java.lang.Thread.sleep(1000);
254     replier.reply("처음 기본 칩은 "+basicchips+"개입니다. "+basicseed+"/"+basics
255     coin[0]-=basicseed;
256     coin[1]-=basicseed*2;
257     chips[0]-=basicseed;
258     chips[1]-=basicseed*2;
259     money=basicseed*3;
260     pot[0]=basicseed;
261     pot2[0]=basicseed;
262     pot[1]=basicseed*2;
263     pot2[1]=basicseed*2;
264     //SB와 BB는 시드금액을 지불합니다.
265
266     for (var i=0;i<2*player.length;i++) beting.push(player[i%player.length]);
267     beting.push(player[0]);
268     beting.push(player[1]);
269     beting.splice(0,2);
270     //첫 베팅의 순서:SB의 좌측부터 시작
271     pc="";
272     for (var k=0; k<=player.length-1;k++) pc=pc+player[k]+": "+pot[k]+"개 베팅/
273     replier.reply(beting[m]+"님부터 베팅을 시작합니다.\n/콜, /폴드, /레이즈 (숫자
274     isBet=true; //베팅시작
275     } else replier.reply("이미 게임이 시작되었습니다.");
276     } else replier.reply("아직 방이 만들어지지 않았습니다.");
277     } else replier.reply("최소 참가자는 2명입니다.");
278 }
279
280 if (isFirstgame==false && cmd=="/업" && isNaN(data)==false){ //시드 올릴까
281     data=Number(data);
282     basicseed+=Math.floor(data);
283     if (basicseed/basicchips>0.05){
284         basicseed=Math.floor(basicchips*0.05);
285         replier.reply("최고시드("+basicseed+"개)로 자동 설정되었습니다.");
286         isFirstgame=true;
287     } else {
288         replier.reply("시드가 "+data+"개 올라 "+basicseed+"개가 되었습니다.");
289         isFirstgame=true;
290     }
291 } else if (isFirstgame==false && msg=="/킵") isFirstgame=true; //올리지 말까
292
293 if (isBet==true){ //베팅 스크립트
294     if (msg == "/체크"){ //체크
295         if (isCheck==0 && sender==beting[m] && room==R && status>=1){
296             replier.reply("체크! 다음 "+beting[m+1]+"님 베팅해주세요.\n/콜, /폴드, /레이즈 (
297             m+=1;
298             isCheck+=1;
299         }
300     } else if (msg=="/콜" && sender==beting[m] && room==R){ //콜
301         if (status==0 || isCheck!=0){
302
303             if (allin.indexOf(sender)==-1 && (Math.max.apply(null, pot)-pot[player.indexOf(
304                 var whole=coin[player.indexOf(sender)];
305                 coin[player.indexOf(sender)]=0;
306                 chips[player2.indexOf(sender)]=0;
307                 money += whole;
308                 pot[player.indexOf(sender)]+=whole;
309                 pot2[player2.indexOf(sender)]+=whole;
310                 side.push(money);
311                 allin.push(sender);
312                 replier.reply(sender+"님 올인! 사이드베팅이 적용됩니다."); //콜 금액이 보유금
313             } else if (allin.indexOf(sender)!=-1) replier.reply(sender+"님은 올인하셨으므로

```

```

314     else {
315         coin[player.indexOf(sender)]=coin[player.indexOf(sender)]-Math.max.apply(null, chips);
316         chips[player2.indexOf(sender)]=chips[player2.indexOf(sender)]-Math.max.apply(null, chips);
317         money = money+ Math.max.apply(null, pot)-pot[player.indexOf(sender)];
318         pot[player.indexOf(sender)]=Math.max.apply(null, pot);
319         pot2[player2.indexOf(sender)]=Math.max.apply(null, pot2);
320     }
321     pc="";
322     for (var k=0; k<=player.length-1;k++) pc=pc+player[k]+": "+pot[k]+"개 베팅/"+"co
323     pc=pc.trim();
324     call +=1; //콜 횟수
325
326     if (call==player.length-1){ //콜을 마지막으로 베팅 완료
327         replier.reply("베팅이 완료되었습니다.\n\nPOT: "+money+"개\n\n"+pc.trim());
328         isBet=false;
329         status+=1;
330         call=0;
331         isCheck=0;
332     } else { //베팅이 완료되지 않았을 시 다음 주자로 넘어갑니다.
333         replier.reply("다음 "+beting[m+1]+"님 베팅해주세요.\n/콜, /폴드, /레이즈 (숫자)");
334         m+=1;
335         isCheck+=1;
336     }
337 }
338 } else if (msg=="/폴드" && sender==beting[m] && room==R){ //폴드
339     if (player.length != 2) { //플레이어가 2인 이상일때
340         coin.splice(player.indexOf(sender),1);
341         pot.splice(player.indexOf(sender),1);
342         cardlist.splice(player.indexOf(sender),1);
343         player.splice(player.indexOf(sender),1);
344         if (allin.indexOf(sender)!=-1){
345             side.splice(allin.indexOf(sender),1);
346             allin.splice(allin.indexOf(sender),1);
347         }
348         for (var i=0;i<=2;i++) beting.splice(beting.indexOf(sender),1);
349         pc="";
350         for (var k=0; k<=player.length-1;k++) pc=pc+player[k]+": "+pot[k]+"개 베팅/"+"co
351         replier.reply(sender+"님 폴드(다이)하셨습니다.");
352
353         if (nagagi.indexOf(sender)!=-1){
354             chips.splice(player2.indexOf(sender),1);
355             pot2.splice(player2.indexOf(sender),1);
356             player2.splice(player2.indexOf(sender),1);
357
358             replier.reply(sender+"님이 나가셨습니다.\n남은 참가자: "+player2);
359             nagagi.splice(nagagi.indexOf(sender),1);
360         } //폴드 시 게임에서 완전 나가집니다.
361
362         if (call==player.length-1){ //폴드를 마지막으로 베팅이 완료될 경우
363             replier.reply("베팅이 완료되었습니다.\n\nPOT: "+money+"개\n\n"+pc.trim());
364             isBet=false;
365             status+=1;
366             call=0;
367             isCheck=0;
368         } else { //베팅이 완료되지 않았을 경우
369             m=m-Math.floor(m/(player.length+1));
370             if (isCheck==0) replier.reply("다음 "+beting[m]+"님 베팅해주세요.\n/체크, /폴드, /레이즈 (숫자)");
371             //첫 주자가 폴드할 경우
372             else replier.reply("다음 "+beting[m]+"님 베팅해주세요.\n/콜, /폴드, /레이즈 (숫자)");
373             //아닐 경우
374         }
375     } else { //플레이어가 2명이었을 경우. 폴드 시 남은 1명이 자동 승리하게 됩니다.
376         player.splice(player.indexOf(sender),1);
377         chips[player2.indexOf(player[0])]+=money;
378         pc="현재 칩 보유량";
379         for (var k=0; k<=player2.length-1;k++) pc=pc+"\n"+player2[k]+": "+chips[k]+"개"
380         replier.reply("나머지 선수가 모두 폴드하여 "+player+"님 승리입니다!\n\n"+pc);
381         if (nagagi.length!=0){
382             for (i=0;i<nagagi.length;i++){

```

```

382         for (i=0; i<nagagi.length; i++){
383             player2.splice(player2.indexOf(nagagi[i]),1);
384             chips.splice(player2.indexOf(nagagi[i]),1);
385         }
386         replier.reply(nagagi+"님이 나가셨습니다.\n남은 참가자: "+player2);
387     }
388     Gstart=false;
389     player=[]; coin=[]; pot=[]; pot2=[];
390     cardlist=[]; common=[];
391     allin=[]; side=[];
392     player2.unshift(player2[player2.length-1]);
393     chips.unshift(chips[chips.length-1]);
394     player2.pop();
395     chips.pop();
396     //게임이 끝날 때마다 자리가 한칸씩 밀려나 SB와 BB가 바뀝니다.
397     for ( var i=0; i<=player2.length-1;i++){
398         coin.push(chips[i]);
399         player.push(player2[i]);
400     }
401     for(var j=0; j<=player.length-1; j++){
402         pot.push(0);
403         pot2.push(0);
404         cardlist.push([]);
405     }
406     beting=[]; money=0; call=0; m=0; isBet=false; status=0;
407     x="";
408     isFirstgame=false;
409     nagagi=[];
410     isChecked=0;
411     replier.reply("새로운 참가자가 저에게 랜덤폭으로 '/홀덤 참가'를 보내주시면 참가 처리
412 }
413 } else if (cmd=="레이즈" && sender==beting[m] && room==R){ //레이즈
414     if (isNaN(data) == false){
415         if ((data-Math.floor(data))==0){
416             if ((Math.max.apply(null, pot)-pot[player.indexOf(sender)])<data && data<=
417                 if (m<beting.length-player.length){
418                     if(allin.indexOf(sender)==-1){
419                         data=Number(data);
420                         if (data>=coin[player.indexOf(sender)]) { //레이즈 금액이 보유금
421                             var whole=coin[player.indexOf(sender)];
422                             coin[player.indexOf(sender)]=0;
423                             chips[player2.indexOf(sender)]=0;
424                             money += whole;
425                             pot[player.indexOf(sender)]+=whole;
426                             pot2[player2.indexOf(sender)]+=whole;
427                             side.push(money);
428                             allin.push(sender);
429                             replier.reply(sender+"님 올인! 사이드베팅이 적용됩니다.");
430                         } else { //아닐 경우
431                             pot[player.indexOf(sender)]+=data;
432                             pot2[player2.indexOf(sender)]+=data;
433                             coin[player.indexOf(sender)]-=data;
434                             chips[player2.indexOf(sender)]-=data;
435                             money +=data;
436                         }
437                         call=0; //레이즈 시 콜 횟수가 초기화됩니다.
438                         pc="";
439                         for (var k=0; k<=player.length-1;k++) pc=pc+player[k]+": "+pot[
440                         replier.reply("다음 "+beting[m+1]+"님 베팅해주세요.\n/콜, /폴드,
441                         m+=1;
442                         isChecked+=1;
443                     } else replier.reply("올인한 플레이어는 /콜, /폴드 만 가능합니다.");
444                 } else replier.reply("레이즈 횟수 최대치 달성으로 /콜, /폴드 만 가능합니
445                     //레이즈는 최대 2번(콜과 폴드를 하지 않아야) 가능합니다.
446                 } else replier.reply("베팅금액이 콜 이하이거나 레이즈 최대치("+2*Math.max.ar
447                 } else replier.reply("자연수만 입력해주세요.");
448             } else replier.reply("숫자만 입력해주세요.");
449         }
450     }

```



```

451
452 if (status==1 && isBet==false && common.length==0){ //첫 베팅이 끝났을 때
453     m=0;
454     beting=[];
455     for (var i=0; i<3*player.length; i++) beting.push(player[i%player.length]);
456     for (var i=0; i<3; i++){
457         common.push(cards[cards.length-1]);
458         cards.pop();
459     }
460     for (var i=0; i<player.length; i++) for (var j=0; j<common.length; j++) cardlist[i].push(cc
461 //두번째 이상의 베팅에서는 SB부터 베팅을 하도록 순서가 변경됩니다.
462     var ans="공통 카드 3장을 공개합니다.\n"+common; //공통 카드 공개
463     replier.reply(sh(ans));
464     for (var i=0; i<player.length; i++){
465         var ans="현재 나의 카드\n"+cardlist[i]+"현재 등급: "+outgrade(grade(cardlist[i]))[0]
466         replier.reply(player[i],sh(ans));
467     } //공통 카드가 공개될 때마다 현재 나의 카드와 등급이 공개됩니다.
468     replier.reply(beting[m]+"님부터 베팅을 시작합니다.\n/체크, /폴드, /레이즈 (숫자) 로 베팅이
469     isBet=true; //베팅 다시 시작!
470 }
471
472 if (status>=2 && status<=3 && isBet==false && common.length==status+1){ //두번째, 세번째 베팅
473     m=0;
474     if (status==3 ) x="마지막 ";
475     common.push(cards[cards.length-1]);
476     cards.pop();
477     for (var j=0; j<player.length; j++) cardlist[j].push(common[status+1]);
478     var ans=x+"추가 공통 카드 1장을 공개합니다: "+common[status+1]+"공통 카드: "+common;
479     replier.reply(sh(ans));
480     for (var i=0; i<player.length; i++){
481         var ans="현재 나의 카드\n"+cardlist[i]+"현재 등급: "+outgrade(grade(cardlist[i]))[0]
482         replier.reply(player[i],sh(ans));
483     }
484     replier.reply(beting[m]+"님부터 "+x+"베팅을 시작합니다.\n/체크, /폴드, /레이즈 (숫자) 로
485     isBet=true; //베팅 다시 시작!
486 }
487
488 if (status==4 && isBet==false){ //마지막 베팅이 끝나고 결과를 계산합니다.
489     var gl=[]; //플레이어별 족보 등급 리스트
490     var prize=[]; //플레이어별 획득할 수 있는 칩 개수
491     var winners=[]; //플레이어를 1등부터 순위대로 나열
492     var ult="공통 카드: "+common+"\n\n핸드 카드를 공개합니다."; //각자의 카드 모두 공개
493     var Res="";
494     replier.reply("마지막 베팅이 종료되었습니다.");
495     for (var i=0; i<player.length; i++){
496         var gg=player[i]+": "+outgrade(grade(cardlist[i]))[0]+"n";
497         Res=Res+gg;
498         gg=player[i]+": "+cardlist[i][0]+", "+cardlist[i][1];
499         ult=ult+"\n"+gg;
500     } //족보 등급 출력
501     for (var i=0; i<player.length; i++) gl.push([grade(cardlist[i]),player[i]]);
502     gl.sort((a,b)=>b[0][2]-a[0][2]);
503     gl.sort((a,b)=>b[0][1]-a[0][1]);
504     gl.sort((a,b)=>b[0][0][1]-a[0][0][1]);
505     gl.sort((a,b)=>b[0][0][0]-a[0][0][0]); //등급을 순위대로 나열
506     for (var i=0; i<gl.length; i++) winners.push(gl[i][1]); //순위에 맞는 플레이어 나열
507     for (var i=0; i<winners.length; i++) prize.push(money);
508     if (allin.length>0) for (var i=0; i<allin.length; i++) prize[winners.indexOf(allin[i])]
509     //플레이어별 획득가능 칩 배열. 올인한 플레이어의 획득가능 칩은 사이드금액으로 저장됩니다.
510     for (var i=0; i<prize.length-1; i++){
511         if (prize[i]>=prize[i+1]) {
512             winners.splice(i+1,prize.length-i-1);
513             prize.splice(i+1,prize.length-i-1);
514             break;
515         }
516     } //상금을 받을 수 있는 플레이어들만 가려냅니다.
517     replier.reply(sh(ult));
518     replier.reply(sh(Res.trim()));
519

```



```

520     for (var i=0;i<winners.length;i++){
521         chips[player2.indexOf(winners[i])] += prize[i];
522         for (var j=i;j<prize.length-1;j++) prize[j+1] -= prize[i];
523     } //상금 분배
524     pc="현재 칩 보유량";
525     for (var k=0; k<=player2.length-1;k++) pc=pc+"\n"+player2[k]+": "+chips[k]+"개";
526     replier.reply("승리자는 "+winner+"님입니다!\n\n"+pc);
527     //결과 출력
528     if (nagagi.length!=0){
529         for (i=0;i<nagagi.length;i++){
530             player2.splice(player2.indexOf(nagagi[i]),1);
531             chips.splice(player2.indexOf(nagagi[i]),1);
532         }
533         replier.reply(nagagi+"님이 나가셨습니다.\n남은 참가자: "+player2);
534     } //나가기
535     Gstart=false;
536     coin=[]; player=[]; pot=[]; pot2=[];
537     cardlist=[]; common=[];
538     player2.unshift(player2[player2.length-1]);
539     chips.unshift(chips[chips.length-1]);
540     player2.pop();
541     chips.pop(); //자리가 바뀝니다
542     for ( var i=0; i<=player2.length-1;i++){
543         coin.push(chips[i]);
544         player.push(player2[i]);
545     }
546     for(var j=0; j<=player.length-1; j++){
547         pot.push(0);
548         pot2.push(0);
549         cardlist.push([]);
550     }
551     beting=[]; money=0; call=0; m=0; isBet=false; status=0;
552     x="";
553     isFirstgame=false;
554     nagagi=[];
555     isCheck=0;
556     allin=[]; side=[];
557     replier.reply("새로운 참가자가 저에게 랜덱으로 '/홀덤 참가'를 보내주시면 참가 처리됩니다.\n");
558 }
559
560 if (msg=="/나가기" && player2.indexOf(sender)!=-1 && nagagi.indexOf(sender)==-1){
561     if (Gstart==true){ //게임이 진행중인 경우 나가가 예약됨
562         nagagi.push(sender);
563         replier.reply(R, sender+"님의 나가가 예약되었습니다. 폴드 시 혹은 한 게임이 끝날 시");
564     } else { //게임이 진행 중이 아닌 경우 바로 나가짐
565         player.splice(player.indexOf(sender),1);
566         player2.splice(player2.indexOf(sender),1);
567         replier.reply(R, sender+"님이 나가셨습니다.\n남은 참가자: "+player2);
568     }
569 }
570
571 if (msg=="/나가기취소" && player2.indexOf(sender)!=-1 && nagagi.indexOf(sender)!=-1){
572     nagagi.splice(nagagi.indexOf(sender),1);
573     replier.reply(R, sender+"님 나가기 예약 취소");
574 } //나가기 예약이 되어 있었을 경우 취소
575
576 if (Gstart==true && player2.length==1){
577     replier.reply("참가자가 1명밖에 남지 않아 게임이 자동 종료됩니다.");
578     Api.compile("홀덤.js");
579 } //모두 나가고 1명밖에 남지 않았을때 자동 게임 종료
580
581 if (msg=="/홀덤 종료" && player2.indexOf(sender)!=-1){
582     turnoff=true;
583     onoff.push(sender);
584     replier.reply("/찬성 으로 종료 찬성이 가능합니다. 5초 이내 과반수 이상 찬성 시 게임이 종료");
585     java.lang.Thread.sleep(5000);
586     if (onoff.length>=player2.length/2){
587         replier.reply(onoff.length+"명의 찬성으로 게임이 종료됩니다.");
588         Api.compile("홀덤.js");

```

```

589     } else {
590         replier.reply("과반수가 넘지 않아 게임이 계속 진행됩니다.");
591         onoff=[];
592         turnoff=false;
593     }
594 } //게임 종료: 투표제. 5초 이내 과반수 이상 찬성 시 자동 종료. 종료 명령어를 친 사람은 자동 찬성
595
596 if (turnoff=true && onoff.indexOf(sender)===-1 && msg=="/찬성"){
597     onoff.push(sender);
598     replier.reply(sender+"님 찬성");
599 } //찬성
600 }

```

이제 7포커를 만들 준비가 되었습니다.

각종 오류 지적해주시면 정말 감사하겠습니다.

질문은 댓글로



seori님의 게시물 더보기 >

❤️ 좋아요 1 💬 댓글 4

🔗 공유 | 신고

댓글 등록순 최신순 ↻

댓글알림 ☐



카톡우앙
고생추~~

2020.02.07. 05:57 답글쓰기



seori 작성자



2020.02.07. 06:07 답글쓰기



새름
멋지셔용!! 수고하셔어요

2020.02.07. 07:17 답글쓰기



seori 작성자

고약 몇시간도 안돼서 오류발견

2020.02.07. 09:49 답글쓰기

Hibot

댓글을 남겨보세요



등록

✎ 글쓰기

답글

목록

▲ TOP

'| 일반 소스 공유 |' 게시판 글

이 게시판 새글 구독하기 ☐

문세 날짜 추가 소스요 ㄸ 3	bidforpower23	2020.02.07.
더이상 (은)는, (이)가, (을)를을 할 필요가 없습니다. 15	Violet	2020.02.07.
(공유) 포커[텍사스 홀덤] v1.8 by seori : 완성!! 4	seori	2020.02.07.
제 첫 소스에요!! 19	FFaker	2020.02.06.
jsoup를 이용한 두번째 소스 2	deadmau5	2020.02.06.

1 2 3

전체보기

이 카페 인기글

안녕하세요

0 0 0 5

안녕하세요

tomohong
♡ 1 💬 3



틱택토 (카카오링크 적용)

네이버에서 지역별 날씨 정보 크롤링하기 / 스압 알아서 주의

년

[학습] 모니카는 여기에 잘 있음

[illegible]

[카카오톡 못] 양산형 사동약습 못 소스 공유



혹시 여기서 변수를 저장하는 부분이 어딘지

1 2 3 4 5



틱택토 (Player vs Player)

타 메신저 사용시 무한반복

NaN

자동학습 봇 특정 말만 무시하기?

헤히
♡0 💬4