

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  char board1[10][10]; //defines the two boards for the two players as global boards, that way they can be
called any time without having to pass them through every function
4  char board2[10][10];
5  char board3[10][10];
6  char board4[10][10];
7  #define EMPTY 10 //defines all the states that the values of the arrays can be
8  #define MISS 16
9  #define HIT 17
10 #define FULL 18
11 #define STRIKE 19
12
13 void menu () { //menu for starting the game, viewing rules, and quitting the application
14 int n;
15 printf("Battleship by Captain Nick Rodgers\n\n\n");
16 printf("Welcome! please select an option to continue\n");
17 printf("1: Start the game\n2: View rules \n3: Quit (this might delete system31) \n");
18 scanf ("%d",&n);
19 switch (n){
20     case 1:
21         reset();
22         break;
23     case 2:
24         system("cls");
25         rules();
26         break;
27     case 3:
28         exit(0); //exits the application
29         break;
30 }
31
32 }
33
34 void rules (void) { //prints the rules, when the user presses any key it returns to the menu
35 printf("Rules of Battleship:\n");
36 printf("1: Enter an X and Y coordinate, as well as a direction and a ship to place your ships\n");
37 printf("2: Players go in turns, don't screencheat like a little bitch\n");
38 printf("3: Guess coordinates of enemy ships by entering an X and Y coordinate into our highly advanced
nuclear launch platform during your turn.\n Sink all ships to win!\n");
39 printf("(developers note: winner gets bragging rights and loser has to buy winner a beer)\n");
40 printf("\t\t\tPress any key to return to menu\n");
41 getch(); //scans for any key before proceeding to the next command
42 system("cls"); //very handy operator, clears the screen making it hard to cheat and look at the other
player's screen
43 menu();
44 }
45
46 void printboardplayer1 () { //prints the board for player 1, all comments for this apply to
printboardplayer2 and the printupperboards as well
47 system("cls"); //clears the screen to ensure no cheating and to keep things tidy.
48 printf("player 1 turn\n");
49 printupperboard1();
50 int i,j,n=0,m=0; //variables needed to traverse array as well as print the numbers along the sides of the
board.
51 printf("  0 1 2 3 4 5 6 7 8 9\n"); // prints the top row of numbers: could be done with a loop but since
battleship is a set size, this is simpler
52 printf("%d ",m);
53 for (i=0;i<10; i++){
54     for (j=0; j<10; j++){ //traverses through the 2D array
55         n=n+1; //increases counter that decides when row ends, when it reaches 10, a new row is created
56         switch (board1[i][j]) { //simple switch case, to decide which symbol gets printed where in the
board, enumeration makes this easy to read
57             case EMPTY:
58                 printf ("~ "); //water tile
59                 break;

```

```

60     case MISS:
61         printf ("O "); //missed shot
62         break;
63     case FULL:
64         printf ("I "); //location of part of a ship
65         break;
66     case STRIKE:
67         printf ("X "); //where a ship has been hit
68         break;
69     default:
70         break;
71
72     }
73 }
74 m=m+1; //prints the number at the beginning of the row
75 if (n==10){ //starts a new row
76     printf("\n");
77     n=0; //resets row counter
78     if(m<10)
79         printf("%d ",m);
80 }
81 }
82 }
83
84 void printboardplayer2 (){ //prints the board for player 2
85     system("cls");
86     printf("player 2 turn\n");
87     printupperboard2();
88     int i,j,n=0,m=0;
89     printf("  0 1 2 3 4 5 6 7 8 9\n");
90     printf("%d ",m);
91     for (i=0;i<10; i++){
92         for (j=0; j<10 ;j++){
93             n=n+1;
94             switch (board2[i][j]){
95                 case EMPTY:
96                     printf ("~ ");
97                     break;
98                 case MISS:
99                     printf ("O ");
100                    break;
101                 case FULL:
102                     printf ("I ");
103                     break;
104                 case STRIKE:
105                     printf ("X ");
106                     break;
107                 default:
108                     break;
109
110             }
111         }
112         m=m+1;
113         if (n==10){
114             printf("\n");
115             n=0;
116             if(m<10)
117                 printf("%d ",m);
118         }
119     }
120 }
121
122 void printupperboard1 (){ //prints upper board for player 1
123     int i,j,n=0,m=0;
124     printf("  0 1 2 3 4 5 6 7 8 9\n");
125     printf("%d ",m);

```

```

126 for (i=0;i<10; i++){
127     for (j=0; j<10 ;j++){
128         n=n+1;
129         switch (board3[i][j]){
130             case EMPTY:
131                 printf ("~ ");
132                 break;
133             case MISS:
134                 printf ("O ");
135                 break;
136             case STRIKE:
137                 printf ("X ");
138                 break;
139             default:
140                 break;
141         }
142     }
143 }
144 m=m+1;
145 if (n==10){
146     printf("\n");
147     n=0;
148     if(m<10)
149         printf("%d ",m);
150 }
151 }
152 printf("_____\\n");
153 }
154
155 void printupperboard2 () { //prints upper board for player 1
156     int i,j,n=0,m=0;
157     printf("  0 1 2 3 4 5 6 7 8 9\\n");
158     printf("%d ",m);
159     for (i=0;i<10; i++){
160         for (j=0; j<10 ;j++){
161             n=n+1;
162             switch (board4[i][j]){
163                 case EMPTY:
164                     printf ("~ ");
165                     break;
166                 case MISS:
167                     printf ("O ");
168                     break;
169                 case STRIKE:
170                     printf ("X ");
171                     break;
172                 default:
173                     break;
174             }
175         }
176     }
177     m=m+1;
178     if (n==10){
179         printf("\n");
180         n=0;
181         if(m<10)
182             printf("%d ",m);
183     }
184 }
185 printf("_____\\n");
186 }
187
188 void player1taketurn(){ //allows player 1 to make their turn and prints their board, all comments apply to
player2taketurn as well
189     printboardplayer1(); //prints the players upper and lower board
190     int i,j; //variables for coordinates of attack

```

```

191 printf("player 1, please enter coordinates to attack\n");
192 scanf("%d,%d",&i,&j); //scans the the user entered coordinates
193 switch (board2[i][j]){ //switch case to decide what to set the new value of the opposite board to at the
chosen coordinate
194 case EMPTY:
195     board2[i][j]=MISS;
196     board3[i][j]=MISS;
197     printf("missed!\n"); //if the coordinate on the opposite board is empty, a 0 will be printed there
instead of the ~ water tile
198     break;
199 case FULL:
200     board2[i][j]=STRIKE;
201     board3[i][j]=STRIKE;
202     printf("hit!\n"); //if there is a ship at this coordinate in board 2, an X will be printed in place of
the I ship to signify that it has been hit
203     break;
204 }
205 printf("press any key to finish turn\n"); //waits for the player to be ready to end turn, to ensure no
cheating
206 getch();
207 system("cls");
208 checkvictory(); //checks to see if either player has won so far
209 printf("player 2 turn\n");
210 player2taketurn(); //allows other player to go
211 }
212
213 void player2taketurn(){ //allows player 2 to make their turn and prints their board
214 printboardplayer2();
215     int i,j;
216 printf("player 2, please enter coordinates to attack\n");
217 scanf("%d,%d",&i,&j);
218 switch (board1[i][j]){
219 case EMPTY:
220     board1[i][j]=MISS;
221     board4[i][j]=MISS;
222     printf("missed!\n");
223     break;
224 case FULL:
225     board1[i][j]=STRIKE;
226     board4[i][j]=STRIKE;
227     printf("hit!\n");
228     break;
229 }
230 printf("press any key to finish turn\n");
231 getch();
232 system("cls");
233 checkvictory();
234 player1taketurn(); //goes back to player 1, so that the pair can take turns until checkvictory finds a
winner
235 }
236
237 void reset () { //initializes board to start a new game
238     int i,j; //variables to traverse both arrays
239     for (i=0;i<10; i++){
240         for (j=0; j<10 ;j++){
241             board1[i][j]=EMPTY; //resets both boards
242             board2[i][j]=EMPTY;
243             board3[i][j]=EMPTY;
244             board4[i][j]=EMPTY;
245         }
246     }
247     system("cls");
248     printf ("board reset, ready to begin a new game\n");
249     setshipsplayer1();
250 }
251

```

```

252 void setshipsplayer1(){ //gets coordinates, direction, and ship type from player 1 to set their ships, all
comments apply to setshipsplayer2
253 system("cls");
254 printboardplayer1(); //prints board for player reference
255 int i,j,count,loop=4; //variables needed to traverse the array, and decide when the player cannot place any
more ships
256 int c=1,b=1,d=1,p=1,n,direction; //a count for each ship to ensure that only one of every ship is printed
to the board
257 printf("player 1, please place your ships\n");
258 printf("ships available: 4 long CARRIER, 3 long BATTLESHIP, 2 long DESTROYER, 1 long PATROL\n");
259 printf("please choose a ship (length), a coordinate set, and a direction (1 for horizontal, 2 for vertical)
to place your ship\n");
260 while (loop!=0){ //loop variable gets smaller with each ship placed, allowing the player to place only 4
ships
261 printf("%d CARRIERS remaining, %d BATTLESHIPS remaining, %d DESTROYERS remaining, %d PATROLS remaining\n",c
,b,d,p); //tells the player what ships they can still place
262 scanf("%d,%d,%d",&n,&i,&j,&direction); //scans the ship to be placed, coordinate of head of ship, and
direction to place the ship
263 if(n==1&&p==0){ //this set of if/else statements makes sure that the player cannot put down two of the same
ship, by cross referencing the length they entered and the number of ships of that length remaining
264     printf("no more PATROLS, try again\n");
265     n=0;
266 }
267 else if(n==2&&d==0){
268     printf("no more DESTROYERS, try again\n");
269     n=0;
270 }
271 else if (n==3&&b==0){
272     printf("no more BATTLESHIPS, try again\n");
273     n=0;
274 }
275 else if (n==4&&c==0){
276     printf("no more CARRIERS, try again\n");
277     n=0;
278 }
279
280 if (direction==1){ //prints the ships according to direction that the player has chosen, 1 is horizontal, 2
is vertical
281     for (count=0;count<n;count++)
282         board1[i][j+count]=FULL;
283 }
284 else if (direction==2){
285     for (count=0;count<n;count++)
286         board1[i+count][j]=FULL;
287 }
288 if(n==1){ //sets the individual ship counts to zero once a ship has been placed, to ensure that only one of
each ship can be placed
289     p=0;
290     loop=loop-1;
291 }
292 else if(n==2){
293     d=0;
294     loop=loop-1;
295 }
296 else if (n==3){
297     b=0;
298     loop=loop-1;
299 }
300 else if (n==4){
301     c=0;
302     loop=loop-1;
303 }
304 printboardplayer1(); //prints the board to show the current places of the ships
305 }
306 printf("press any key to let player 2 have a turn\n");
307 getch();

```

```

308 system("cls"); //clears screen to make ready for other player to enter their ships
309 setshipsplayer2();
310 }
311
312 void setshipsplayer2(){ //gets coordinates, direction, and ship type from player 2 to set their ships
313 system("cls");
314 printboardplayer2();
315 int i,j,count,loop=4;
316 int c=1,b=1,d=1,p=1,n,direction;
317 printf("player 2, please place your ships\n");
318 printf("ships available: 4 long CARRIER, 3 long BATTLESHIP, 2 long DESTROYER, 1 long PATROL\n");
319 printf("please choose a ship (length), a coordinate set, and a direction (1 for horizontal, 2 for vertical)
to place your ship\n");
320 while (loop!=0){
321 printf("%d CARRIERS remaining, %d BATTLESHIPS remaining, %d DESTROYERS remaining, %d PATROLS remaining\n",c
,b,d,p);
322 scanf("%d,%d,%d,%d",&n,&i,&j,&direction);
323 if(n==1&&p==0){
324     printf("no more PATROLS, try again\n");
325     n=0;
326 }
327 else if(n==2&&d==0){
328     printf("no more DESTROYERS, try again\n");
329     n=0;
330 }
331 else if (n==3&&b==0){
332     printf("no more BATTLESHIPS, try again\n");
333     n=0;
334 }
335 else if (n==4&&c==0){
336     printf("no more CARRIERS, try again\n");
337     n=0;
338 }
339
340 if (direction==1){
341     for (count=0;count<n;count++)
342         board2[i][j+count]=FULL;
343 }
344 else if (direction==2){
345     for (count=0;count<n;count++)
346         board2[i+count][j]=FULL;
347 }
348 if(n==1){
349     p=0;
350     loop=loop-1;
351 }
352 else if(n==2){
353     d=0;
354     loop=loop-1;
355 }
356 else if (n==3){
357     b=0;
358     loop=loop-1;
359 }
360 else if (n==4){
361     c=0;
362     loop=loop-1;
363 }
364 printboardplayer2();
365 }
366 printf("press any key to let player 1 have a turn\n");
367 getch();
368 system("cls");
369 player1taketurn();
370 }
371

```

```

372 void checkvictory (){ //checks to see if a player has won yet and if so, declares the winner
373 int i,j; //variables to traverse both arrays
374 int p1=0, p2=0; //number of hits player 1 and 2 have
375 for (i=0;i<10;i++){
376     for (j=0;j<10;j++){
377         if (board3[i][j]==STRIKE) //goes through board 1 and counts up the total hits
378             p1=p1+1;
379         else if (board4[i][j]==STRIKE) //goes through board 2 and counts up the total hits
380             p2=p2+1;
381         else
382             p1=p1;
383     }
384 }
385 }
386 if (p1==10){ // checks to see if player 1 has 10 hits, if so player 1 loses and player 2 wins
387     printf ("\n\n\n\n\n+++++PLAYER 2 VICTORY!+++++\n\n\n\n\n");
388     menu();
389 }
390 else if (p2==10){ // checks to see if player 1 has 10 hits, if so player 1 loses and player 1 wins
391     printf ("\n\n\n\n\n=====PLAYER 1 VICTORY!=====\n\n\n\n\n");
392     menu();
393 }
394 else
395     return;
396
397 }
398
399 int main (){ //main program, calls the menu as soon as program starts
400     menu();
401 }
402
403

```