

```
1 using System;
2 using System.Collections.Generic;
3
4 namespace Battleships
5 {
6     class Location
7     {
8         private int x;
9         private int y;
10
11         private Dictionary<char, int> guessRow = new Dictionary<char, int>()
12         {
13             {'A', 0},
14             {'B', 1},
15             {'C', 2},
16             {'D', 3},
17             {'E', 4},
18             {'F', 5},
19             {'G', 6},
20             {'H', 7},
21             {'I', 8},
22             {'J', 9}
23         };
24
25         public Location(int x, int y)
26         {
27             this.x = x;
28             this.y = y;
29         }
30
31         public Location(String guessLocation)
32         {
33             try
34             {
35                 this.y = guessRow[guessLocation[0]];
36                 this.x = (int)char.GetNumericValue(guessLocation[1]);
37             }
38             catch(System.Exception)
39             {
40                 throw;
41             }
42         }
43
44         public int getX()
45         {
46             return x;
47         }
48
49         public void setX(int x)
50         {
51             this.x = x;
52         }
53
54         public int getY()
55         {
56             return y;
```

```

57     }
58
59     public void setY(int y)
60     {
61         this.y = y;
62     }
63 }
64
65 abstract class Board
66 {
67     private Object[,] board = new Object[10, 10];
68
69     public Object getCell(Location location)
70     {
71         return board[location.getX(), location.getY()];
72     }
73
74     public void setCell(Location location, Object cellContents)
75     {
76         board[location.getX(), location.getY()] = cellContents;
77     }
78
79     protected void initialise(Object initial)
80     {
81         for (int x = 0; x < 10; x++)
82         {
83             for (int y = 0; y < 10; y++)
84             {
85                 setCell(new Location(x, y), initial);
86             }
87         }
88     }
89
90     public void showBoard()
91     {
92         Console.WriteLine("+-----+");
93         Console.WriteLine("|   | A | B | C | D | E | F | G | H | I | J   |");
94         Console.WriteLine("+-----+");
95
96         for (int x = 0; x < 10; x++)
97         {
98             Console.Write("| " + x + " | ");
99
100             for (int y = 0; y < 10; y++)
101             {
102                 Location temp = new Location(x, y);
103                 Console.Write(displayCell(temp) + " | ");
104             }
105
106             Console.WriteLine("\n+-----+");
107         }
108     }

```

```
109
110     protected abstract char displayCell(Location location);
111 }
112
113 class Ship
114 {
115     protected int length;
116     protected char shipSymbol;
117
118     public int getLength()
119     {
120         return length;
121     }
122
123     public char getShipSymbol()
124     {
125         return shipSymbol;
126     }
127 }
128
129 class CarrierShip : Ship
130 {
131     public CarrierShip()
132     {
133         length = 5;
134         shipSymbol = 'A';
135     }
136 }
137
138 class BattleShip : Ship
139 {
140     public BattleShip()
141     {
142         length = 4;
143         shipSymbol = 'B';
144     }
145 }
146
147 class CruiserShip : Ship
148 {
149     public CruiserShip()
150     {
151         length = 3;
152         shipSymbol = 'C';
153     }
154 }
155
156 class SubmarineShip : Ship
157 {
158     public SubmarineShip()
159     {
160         length = 3;
161         shipSymbol = 'S';
162     }
163 }
164
```

```

165 class DestroyerShip : Ship
166 {
167     public DestroyerShip()
168     {
169         length = 2;
170         shipSymbol = 'D';
171     }
172 }
173
174 class ShipBoard : Board
175 {
176     private int totalShips;
177
178     public ShipBoard()
179     {
180         initialise(null);
181
182         Random random = new Random();
183
184         totalShips = 0;
185
186         List<Ship> ships = new List<Ship>()
187         {
188             new DestroyerShip(),
189             new SubmarineShip(),
190             new CruiserShip(),
191             new BattleShip(),
192             new CarrierShip()
193         };
194
195         int locationX;
196         int locationY;
197         int direction;
198         List<Location> shipLocations;
199
200         foreach (Ship ship in ships)
201         {
202             bool shipPlaced = false;
203
204             while (!shipPlaced)
205             {
206                 locationX = random.Next(0, 9);
207                 locationY = random.Next(0, 9);
208                 direction = random.Next(0, 3);
209                 shipLocations = new List<Location>();
210
211                 switch (direction)
212                 {
213                     case 1:
214                         if (locationX - ship.getLength() > 0)
215                         {
216                             for (int x = locationX; x > locationX -
217                                 ship.getLength(); x --)
218                                 addShip(new Location(x, locationY),
219                                     shipLocations);

```

```
219         }
220
221         shipPlaced = placeShips(shipLocations, ship);
222     }
223     break;
224     case 2:
225         if (locationY + ship.getLength() < 10)
226         {
227             for (int y = locationY; y < locationY + ship.getLength(); y++)
228             {
229                 addShip(new Location(locationX, y), shipLocations);
230             }
231
232             shipPlaced = placeShips(shipLocations, ship);
233         }
234         break;
235     case 3:
236         if (locationX + ship.getLength() < 10)
237         {
238             for (int x = locationX; x < locationX + ship.getLength(); x++)
239             {
240                 addShip(new Location(x, locationY), shipLocations);
241             }
242
243             shipPlaced = placeShips(shipLocations, ship);
244         }
245         break;
246     case 4:
247         if (locationY - ship.getLength() > 0)
248         {
249             for (int y = locationY; y > locationY - ship.getLength(); y--)
250             {
251                 addShip(new Location(locationX, y), shipLocations);
252             }
253
254             shipPlaced = placeShips(shipLocations, ship);
255         }
256         break;
257     }
258 }
259 }
260 }
261
262 private bool placeShips(List<Location> locations, Ship ship)
263 {
264     if (locations.Count == ship.getLength())
265     {
266         foreach (Location location in locations)
267         {
268             setCell(location, ship);
```

```
269         totalShips++;
270     }
271     return true;
272 }
273
274     return false;
275 }
276
277 private void addShip(Location location, List<Location> shipLocations)
278 {
279     if (getCell(location) == null)
280     {
281         shipLocations.Add(location);
282     }
283 }
284
285 public int getTotalShips()
286 {
287     return totalShips;
288 }
289
290 public Boolean hasHit(Location guessLocation)
291 {
292     return (getCell(guessLocation) != null);
293 }
294
295 protected override char displayCell(Location location)
296 {
297     Object ship = getCell(location);
298     return (ship == null ? '-' : ((Ship)ship).getShipSymbol());
299 }
300 }
301
302 class Guess
303 {
304     private char guess;
305
306     public Guess(char guess)
307     {
308         this.guess = guess;
309     }
310
311     public char getGuess()
312     {
313         return guess;
314     }
315
316     public void setGuess(char guess)
317     {
318         this.guess = guess;
319     }
320 }
321
322 class GuessBoard : Board
323 {
324     private int totalMisses;
```

```
325     private int totalHits;
326
327     public GuessBoard()
328     {
329         initialise(null);
330         totalHits = 0;
331         totalMisses = 0;
332     }
333
334     protected override char displayCell(Location location)
335     {
336         Object guess = getCell(location);
337         return (guess == null ? '-' : ((Guess)guess).getGuess());
338     }
339
340     public void update(Location guessLocation, bool hasHit)
341     {
342         char guess;
343
344         if (hasHit)
345         {
346             totalHits++;
347             guess = 'x';
348         }
349         else
350         {
351             totalMisses++;
352             guess = '0';
353         }
354
355         setCell(guessLocation, new Guess(guess));
356     }
357
358     public int getTotalHits()
359     {
360         return totalHits;
361     }
362
363     public int getTotalMisses()
364     {
365         return totalMisses;
366     }
367 }
368
369 class Game
370 {
371     private ShipBoard shipBoard;
372     private GuessBoard guessBoard;
373
374     public void play()
375     {
376         shipBoard = new ShipBoard();
377
378         guessBoard = new GuessBoard();
379
380         while(!hasWon() && !hasLost())
```

```
381         {
382             makeGuess();
383         }
384     }
385
386     private Boolean hasLost()
387     {
388         if (guessBoard.getTotalMisses() > 20)
389         {
390             Console.WriteLine("You've lost :(");
391             shipBoard.showBoard();
392             Console.ReadLine();
393             return true;
394         }
395
396         return false;
397     }
398
399     private Boolean hasWon()
400     {
401         if (guessBoard.getTotalHits() == shipBoard.getTotalShips())
402         {
403             Console.WriteLine("You've won :)");
404             Console.ReadLine();
405             return true;
406         }
407
408         return false;
409     }
410
411     private void makeGuess()
412     {
413         Console.Clear();
414         guessBoard.showBoard();
415
416         bool guessValid = true;
417         Location guessLocation;
418
419         do
420         {
421             Console.WriteLine(guessValid ? "Enter a location (e.g. A1):" : "Please enter a valid location: ");
422             String guess = Console.ReadLine().ToUpper();
423
424             if (guess == "")
425             {
426                 System.Environment.Exit(1);
427             }
428
429             if (guess.Length != 2)
430             {
431                 guessValid = false;
432             }
433             else
434             {
435                 try
```



```
436         {
437             guessLocation = new Location(guess);
438             guessBoard.update(guessLocation, shipBoard.hasHit
439                             (guessLocation));
440             guessValid = true;
441         }
442         catch
443         {
444             guessValid = false;
445         }
446     } while (guessValid == false);
447 }
448 }
449
450 class Program
451 {
452     static void Main(string[] args)
453     {
454         new Game().play();
455     }
456 }
457 }
458
```