My Project

Generated by Doxygen 1.8.13

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

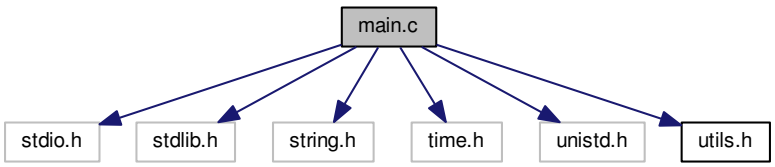
Chapter 2

File Documentation

2.1 main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "utils.h"
```

Include dependency graph for main.c:



Macros

- #define SIZE 10

Functions

- void run (int isPvCOM)
- int main ()

2.1.1 Macro Definition Documentation

2.1.1.1 SIZE

```
#define SIZE 10
```

Definition at line 17 of file main.c.

2.1.2 Function Documentation

2.1.2.1 main()

```
int main ( )
```

Definition at line 131 of file main.c.

```
131           {
132     clearScreen();
133     printBattleship();
134     printMenu1();
135 
136     char optionMenu1;
137     char optionMenu2;
138 
139     do{
140         scanf(" %c", &optionMenu1);
141         switch(optionMenu1){
142             case '1':
143                 clearScreen();
144                 printBattleship();
145                 printMenu2();
146 
147                 do{
148                     scanf(" %c", &optionMenu2);
149                     switch(optionMenu2){
150                         case '1':
151                             clearScreen();
152                             printBattleship();
153                             printf("\n\nBem-vindo à opção de jogo PvCOM!\n\n\n");
154                             run(1);
155                             clearScreen();
156                             printBattleship();
157                             printMenu2();
158                             break;
159                         case '2':
160                             clearScreen();
161                             printBattleship();
162                             printf("\n\nBem-vindo à opção de jogo PvP!\n\n\n");
163                             run(0);
164                             clearScreen();
165                             printBattleship();
166                             printMenu2();
167                             break;
168                         case '3':break;
169                         default:
170                             printf("Opção de menu inválida!\n");
171                     }
172 
173                 }while((optionMenu2)!= '2');
174 
175                 clearScreen();
176                 printBattleship();
177                 printMenu1();
178                 break;
179             case '2':
180                 clearScreen();
181                 printBattleship();
182                 printAbout();
183                 printMenu1();
184                 break;
185             case '3': break;
186             default:
187                 printf("Opção de menu inválida!\n");
188         }
189     }while((optionMenu1 = getchar())!= '3');
190 
191     return(0);
192 }
```

2.1.2.2 run()

```
void run (
            int isPvCOM )
```

Executa o jogo com as opções: 0 se deseja jogar PvP, 1 se deseja jogar PvCOM

Parameters

isPvCOM	Defini o tipo de jogo: 0 PvP ou 1 PvCOM

Definition at line 24 of file main.c.

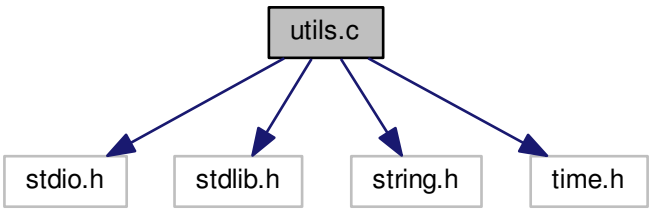
```
24                      {
25      while(1){
26          // inicio o tabuleiro para o primeiro jogador
27          char tabuleiro1[SIZE][SIZE];
28          char jogadas1[SIZE][SIZE];
29          char nomeJogador1[100];
30
31          printf("Jogador 1, informe o seu nome: ");
32          scanf("%s", nomeJogador1);
33          initMatrix(jogadas1);
34          putShips(nomeJogador1, tabuleiro1);
35
36          // inicio o tabuleiro para o segundo jogador
37          char tabuleiro2[SIZE][SIZE];
38          char jogadas2[SIZE][SIZE];
39          char nomeJogador2[100] = "COMPUTER"; // Nome default
40          initMatrix(jogadas2);
41
42          // Se o jogo for PvCOM gero um tabuleiro aleatório para o COMPUTER.
43          // Caso contrário o Jogador 2 deve informar seu nome e posicionar o seu tabuleiro.
44          if(isPvCOM){
45              putShipsCOM(tabuleiro2);
46          }else{
47              printf("Jogador 2, informe o seu nome: ");
48              scanf("%s", nomeJogador2);
49              putShips(nomeJogador2, tabuleiro1);
50          }
51
52          clearScreen();
53          printBattleship();
54
55          // Defino quem irá iniciar o jogo
56          char moeda;
57          printf("Jogador 1, escolha cara (C) ou coroa (O): ");
58          do{
59              scanf(" %c", &moeda);
60              if(moeda != 'C' && moeda != 'O'){
61                  printf("Opção inválida! Tente novamente.\n");
62              }
63          }while(moeda != 'C' && moeda != 'O');
64
65          srand(time(NULL));
66          int primeiroJogador = 2 * (rand() / ((double)RAND_MAX + 1));
67
68          char jogarNovamente;
69          clearScreen();
70          printBattleship();
71
72          printf("O jogador %s irá começar! Preparem-se! \n\n", (primeiroJogador == (moeda == 'C' ? 0 : 1)) ?
    nomeJogador1 : nomeJogador2);
73
74          sleep(2);
75
76          // Jogo
77          do{
78              displayGameInfo(nomeJogador1, jogadas1, nomeJogador2, jogadas2);
79              printf("\nAgora é a vez do %s \n", primeiroJogador == 0 ? nomeJogador1 : nomeJogador2);
80              switch(primeiroJogador){
81                  case 0:
82                      // Jogador 1
83                      giveShot(jogadas1, tabuleiro2, nomeJogador1, nomeJogador2);
84                      break;
85                  case 1:
86                      // Jogador 2
```

```
87                      if(isPvCOM){
88                          sleep(2);
89                          giveRandomShot(jogadas2, tabuleiro1, nomeJogador2, nomeJogador1);
90                      }else{
91                          giveShot(jogadas2, tabuleiro1, nomeJogador2, nomeJogador1);
92                      }
93                      break;
94              }
95
96              // Alterno a jogada
97              primeiroJogador = primeiroJogador == 0 ? 1 : 0;
98          }while(!isEndGame(primeiroJogador == 0 ? jogadas2 : jogadas1)); // Comparo o inverso pois
        mudei o jogador anteriormente
99
100         displayGameInfo(nomeJogador1, jogadas1, nomeJogador2, jogadas2);
101         printf("\n\n\n FIM DE JOGO! %s venceu!\n", primeiroJogador == 0 ? nomeJogador2 : nomeJogador1);
102
103         // Mostro os resultados
104         sleep(2);
105         clearScreen();
106         printBattleship();
107         printResultados(nomeJogador1, jogadas1, tabuleiro1);
108         printf("\n");
109         printResultados(nomeJogador2, jogadas2, tabuleiro2);
110
111         // Novo jogo?
112         printf("\n\n\n\n\n");
113         printf("Pressione S para jogar novamente ou N para sair: ");
114
115         do{
116             scanf(" %c", &jogarNovamente);
117             if(jogarNovamente != 'S' && jogarNovamente != 'N'){
118                 printf("Opção inválida! Tente novamente.\n");
119             }
120         }while(jogarNovamente != 'S' && jogarNovamente != 'N');
121
122         if(jogarNovamente == 'S'){
123             clearScreen();
124             printBattleship();
125         }else{
126             break;
127         }
128     }
129 }
```

2.2 utils.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
```

Include dependency graph for utils.c:



Functions

- void clearScreen ()

- void initMatrix (char tabuleiro[ ][10])
- void printAbout ()
- void printAlign (int space, char ∗string)
- void printBattleship ()
- void printMenu1 ()
- void printMenu2 ()
- const char ∗ directionMessage (char direction)
- const char ∗ shipName (char ship)
- const int getShipSize (char ship)
- int checkThreshold (int line, int col)
- int checkCoordenate (char shipType, char line, int col, char direction, char tabuleiro[ ][10])
- void readCoordenate (char ∗line, int ∗col)
- void checkShot (char jogadas[ ][10], char tabuleiroAdversario[ ][10], char ∗nomeJogador, char ∗nome↩ Adversario, int line, int col)
- void displayGameInfo (char ∗nomeJogador1, char jogadas1[ ][10], char ∗nomeJogador2, char jogadas2[ ][10])
- void displayGamePlayer (char ∗nomeJogador, char tabuleiro[ ][10])
- int isEndGame (char jogadas[ ][10])
- void giveShot (char jogadas[ ][10], char tabuleiroAdversario[ ][10], char ∗nomeJogador, char ∗nome↩ Adversario)
- void giveRandomShot (char jogadas[ ][10], char tabuleiroAdversario[ ][10], char ∗nomeJogador, char ∗nomeAdversario)
- void putShip (char shipType, char line, int col, char direction, char tabuleiro[ ][10])
- void putShips (char ∗nomeJogador, char tabuleiro[ ][10])
- void putShipsCOM (char tabuleiro[ ][10])
- void printResultados (char ∗nomeJogador, char jogadas[ ][10], char tabuleiro[ ][10])

2.2.1 Function Documentation

2.2.1.1 checkCoordenate()

```
int checkCoordenate (
            char shipType,
            char line,
            int col,
            char direction,
            char tabuleiro[][10] )
```

Verifica se a coordenada juntamente com a direção são válidas para inserção de uma embarcação no tabuleiro

Parameters

shipType	Tipo de embarcação
line	Linha da matriz
col	Coluna da matriz
direction	Direção do posicionamento da embarcação
tabuleiro	Tabuleiro do jogo

Returns

1 se a coordenada é válida, 0 caso contrário

Definition at line 168 of file utils.c.

```
168                                                                                                 {
169     int intLine = (int)line-65;
170     /* Verifico:
171      * - se é uma coordenada dentro da matriz
172      * - se a letra informada para a direção é válida
173      * - se a direção e a coordenada ficam dentro da matriz
174     */
175     if(!checkThreshold(intLine, col)
176         || !(direction == 'W' || direction == 'S' || direction == 'A' || direction == 'D')
177         || (direction == 'W' && (intLine - getShipSize(shipType) < 0))
178         || (direction == 'S' && (intLine + getShipSize(shipType) > 10))
179         || (direction == 'A' && (col - getShipSize(shipType) < 0))
180         || (direction == 'D' && (col + getShipSize(shipType) > 10))){
181         return 0;
182     }
183 
184     // Verifico se as posições em que a embarcação será colocada estão livres
185     switch(direction){
186         case 'W':
187             //"Para Cima"
188             for(int i = intLine; i > (intLine - getShipSize(shipType)); i--){
189                 if(tabuleiro[i][col] != '.'){
190                     return 0;
191                 }
192             }
193             break;
194         case 'S':
195             //"Para Baixo"
196             for(int i = intLine; i < (intLine + getShipSize(shipType)); i++){
197                 if(tabuleiro[i][col] != '.'){
198                     return 0;
199                 }
200             }
201             break;
202         case 'D':
203             //"Para Direita"
204             for(int i = col; i < (col + getShipSize(shipType)); i++){
205                 if(tabuleiro[intLine][i] != '.'){
206                     return 0;
207                 }
208             }
209             break;
210         case 'A':
211             //"Para Esquerda"
212             for(int i = col; i > (col - getShipSize(shipType)); i--){
213                 if(tabuleiro[intLine][i] != '.'){
214                     return 0;
215                 }
216             }
217             break;
218     }
219 
220     return 1;
221 }
```

2.2.1.2 checkShot()

```
void checkShot (
            char jogadas[][10],
            char tabuleiroAdversario[][10],
            char * nomeJogador,
            char * nomeAdversario,
            int line,
            int col )
```

Verifica o resultado de uma jogada (disparo) da batalha naval

Parameters

jogadas	Jogadas realizadas pelo jogador até o momento.
tabuleiroAdversario	Tabuleiro do adversário
nomeJogador	Nome do jogador
nomeAdversario	Nome do adversário

Definition at line 254 of file utils.c.

```
254
                        {
255     clearScreen();
256     printBattleship();
257     printf("\n\n");
258
259     printf("Resultado do disparo realizado por: %s\n\n", nomeJogador);
260
261     printf("**********************************************************************\n");
262     // Verifico onde o disparo acertou e atribuo o indicador
263     if(tabuleiroAdversario[line][col] == '.'){
264         jogadas[line][col] = 'O';
265         printf("* Água! \n");
266     }else{
267         printf("* Fogo! \n");
268         jogadas[line][col] = 'X';
269
270         char shipType = tabuleiroAdversario[line][col]; // Guardo o tipo de embarcação atingida
271         int contHits = 0; // Quantas vezes essa embarcação foi atingida
272
273         // Verifico se minha embarcação afundou
274         for(int i = 0; i < 10; i ++){
275             for(int j = 0; j < 10; j++){
276                 // Se há um disparo para minha embarcação, acumulo o número de vezes que ela foi acertada
277                 if(jogadas[i][j] == 'X' && tabuleiroAdversario[i][j] == shipType){
278                     contHits++;
279                 }
280             }
281         }
282
283         // Se acertei o número de vezes igual ao tamanho dela, significa que afundei a embarcação
284         if(contHits == getShipSize(shipType)){
285             printf("* %s: Afundou %s %s!\n", nomeAdversario, shipType == 'N' ? "minha" : "meu",
    shipName(shipType));
286
287             // Mostro no tabuleiro minha embarcação
288             for(int i = 0; i < 10; i ++){
289                 for(int j = 0; j < 10; j++){
290                     // Procuro as posições da embarcação afundada e coloco a sigla dela para ser mostrada
291                     if(tabuleiroAdversario[i][j] == shipType){
292                         jogadas[i][j] = shipType;
293                     }
294                 }
295             }
296         }
297     }
298     printf("**********************************************************************\n");
299     printf("\n\n\n");
300 }
```

2.2.1.3 checkThreshold()

```
int checkThreshold (
            int line,
            int col )
```

Verifica se a coordenada é válidas para inserção de uma embarcação no tabuleiro

Parameters

line	Linha da matriz
col	Coluna da matriz

Returns

1 se a coordenada é válida, 0 caso contrário

Definition at line 154 of file utils.c.

```
154                                         {
155     return !(line < 0 || col < 0 || line > 10 || col > 10);
156 }
```

2.2.1.4 clearScreen()

```
void clearScreen ( )
```

Limpa a tela do console

Definition at line 18 of file utils.c.

```
18                      {
19   system("@cls||clear");
20 }
```

2.2.1.5 directionMessage()

```
const char* directionMessage (
            char direction )
```

Retorna a mensagem correspondende à cada direção

Definition at line 98 of file utils.c.

```
98                                                  {
99     switch(direction){
100         case 'W':
101             return "Para Cima";
102         case 'S':
103             return "Para Baixo";
104         case 'D':
105             return "Para Direita";
106         case 'A':
107             return "Para Esquerda";
108     }
109 }
```

2.2.1.6 displayGameInfo()

```
void displayGameInfo (
             char * nomeJogador1,
             char jogadas1[][10],
             char * nomeJogador2,
             char jogadas2[][10] )
```

Mostra as informações dos tabuleiros lado a lado de dois jogadores com os seus respectivos nomes. Exemplo: Player | Computer 0 1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 A |P| | | | | | | | | | A | | | | | | | | | | B |P| | | | | | | | | | B | | | | | | | | | | C |P| | | | | | | | | | C | | | | | |O| | | | D |P| | | | | | | | | | D | | |x| | | | | | | E |P| | | | | | | | | | E | | | | | | | | | | F | | | | | | | | | | F | | | | |O| | | | | G | | | | | | | | | | G | | | | | | | | | | H | | | | | | | | | | H | |O| | | | |O| | | | I | | | | | | | | | | I | | | | | | | | | | J | | | | | | | | | | J | | | | | | | | | |

Parameters

nomeJogador1	Nome do jogador 1
jogadas1	Tabuleiro de jogadas do jogador 1
nomeJogador2	Nome do jogador 2
jogadas2	Tabuleiro de jogadas do jogador 2

Definition at line 323 of file utils.c.

```
323                                                                              {
324     char separador[] = "     |    ";
325
326     // Imprimo o cabeçalho
327     printAlign(41, nomeJogador1);
328     printf("%s", separador);
329     printAlign(44, nomeJogador2);
330     printf("\n");
331
332     for(int cont = 0; cont < 2; cont++){
333         printf(" ");
334         for (int i = 0; i < 10; i++) {
335             printf("   %i", i);
336         }
337         if(!cont){
338             printf("%s", separador);
339         }
340     }
341     printf("\n");
342
343     // Imprimo as linhas
344     for (int i = 0; i < 10; i++) {
345         // Imprime a letra
346         printf("%c | ", 65+i);
347
348         // Imprime os valores do player1
349         for (int j = 0; j < 10; j++) {
350             printf("%c | ", jogadas1[i][j]);
351         }
352
353         // Imprime a letra
354         printf("   |    %c | ", 65+i);
355         // Imprime os valores do player 2
356         for (int j = 0; j < 10; j++) {
357             printf("%c | ", jogadas2[i][j]);
358         }
359         printf("\n");
360     }
361 }
```

2.2.1.7 displayGamePlayer()

```
void displayGamePlayer (
            char * nomeJogador,
            char tabuleiro[ ][10] )
```

Mostra as informações do tabuleiro do jogador com os seu respectivo nome. Exemplo: Player 0 1 2 3 4 5 6 7 8 9 A |P| | | | | | | | | | B |P| | | | | | | | | | C |P| | | | | | | | | | D |P| | | | | | | | | | E |P| | | | | | | | | | F | | | | | | | | | | | G | | | | |N|N|N| | | | H | | | | | | | | | | | I | | | | | | | | | | | J | | | | | | | | | | |

Parameters

nomeJogador	Nome do jogador
tabuleiro	Tabuleiro do jogador

Definition at line 382 of file utils.c.

```
382                                                                 {
383     char separador[] = "     |   ";
384
385     // Imprimo o cabeçalho
386     printAlign(41, nomeJogador);
387     printf("\n");
388
389     printf(" ");
390     for (int i = 0; i < 10; i++) {
391         printf("   %i", i);
392     }
393     printf("\n");
394
395     // Imprimo as linhas
396     for (int i = 0; i < 10; i++) {
397         // Imprime a letra
398         printf("%c | ", 65+i);
399
400         // Imprime os valores do jogador
401         for (int j = 0; j < 10; j++) {
402             printf("%c | ", tabuleiro[i][j]);
403         }
404         printf("\n");
405     }
406 }
```

2.2.1.8 getShipSize()

```
const int getShipSize (
              char ship )
```

Retorna o tamanho da embarcação de acordo com a sigla

Definition at line 132 of file utils.c.

```
132                                 {
133     switch(ship){
134         case 'P':
135             return 5;
136         case 'N':
137             return 4;
138         case 'C':
139             return 3;
140         case 'S':
141             return 3;
142         case 'D':
143             return 2;
144     }
145 }
```

2.2.1.9 giveRandomShot()

```
void giveRandomShot (
              char jogadas[][10],
              char tabuleiroAdversario[][10],
              char * nomeJogador,
              char * nomeAdversario )
```

A máquina realiza uma jogada (disparo) da batalha naval

Parameters

jogadas	Jogadas realizadas pelo jogador até o momento.
tabuleiroAdversario	Tabuleiro do adversário
nomeJogador	Nome do jogador
nomeAdversario	Nome do adversário

Definition at line 492 of file utils.c.

```
492
          {
493     int col, line;
494
495     do{
496         srand(time(NULL));
497         line = rand() % 10;
498         col = rand() % 10;
499
500         if(jogadas[line][col] == '.'){
501             break;
502         }
503     }while(1);
504
505     checkShot(jogadas, tabuleiroAdversario, nomeJogador, nomeAdversario, line, col);
506 }
```

2.2.1.10 giveShot()

```
void giveShot (
              char jogadas[][10],
              char tabuleiroAdversario[][10],
              char * nomeJogador,
              char * nomeAdversario )
```

Realiza uma jogada (disparo) da batalha naval por um jogador

Parameters

jogadas	Jogadas realizadas pelo jogador até o momento.
tabuleiroAdversario	Tabuleiro do adversário
nomeJogador	Nome do jogador
nomeAdversario	Nome do adversário

Definition at line 464 of file utils.c.

```
464                                                                                                 {
465     int col, intLine;
466     char line;
467
468     // O programa solicita um disparo válido
469     do{
470         readCoordenate(&line, &col);
471
472         intLine = (int)line-65;
473
474         if(jogadas[intLine][col] == '.'){
475             break;
476         }else{
```

```
477             printf("Essa coordenada já foi informada anteriormente. Tente novamente.\n");
478         }
479     }while(1);
480 
481     checkShot(jogadas, tabuleiroAdversario, nomeJogador, nomeAdversario, intLine, col);
482 }
```

2.2.1.11 initMatrix()

```
void initMatrix (
             char tabuleiro[][10] )
```

Inicializo cada posição do tabuleiro do jogador com '.'

Parameters

tabuleiro	Tabuleiro do jogador

Definition at line 27 of file utils.c.

```
27                                      {
28     for (int i = 0; i < 10; i++) {
29       for (int j = 0; j < 10; j++) {
30         tabuleiro[i][j] = '.';
31       }
32     }
33 }
```

2.2.1.12 isEndGame()

```
int isEndGame (
             char jogadas[][10] )
```

Verifica se o jogo já acabou

Parameters

jogadas	Tabuleiro de jogadas de algum jogador

Returns

Retorna 1 se todas as embarcações foram afundadas, 0 caso contrário

Definition at line 414 of file utils.c.

```
414                                 {
415     int embarcacoesAfundadas = 0;
416 
417     // Verifico se todas as embarcações foram afundadas
```

```
418     for(int i = 0; i < 5; i++){
419
420         char shipType;
421
422         switch(i){
423             case 0:
424                 shipType = 'P';
425                 break;
426             case 1:
427                 shipType = 'N';
428                 break;
429             case 2:
430                 shipType = 'C';
431                 break;
432             case 3:
433                 shipType = 'S';
434                 break;
435             case 4:
436                 shipType = 'D';
437                 break;
438         }
439
440         // Se o tabuleiro contém a letra da embarcação significa que essa embarcação foi afundada
441         for(int i = 0; i < 10; i++){
442             for(int j = 0; j < 10; j++){
443                 if(jogadas[i][j] == shipType){
444                     embarcacoesAfundadas++;
445                     i = j = 10;
446                     break;
447                 }
448             }
449         }
450     }
451
452
453     return embarcacoesAfundadas == 5 ? 1 : 0;
454 }
```

2.2.1.13 printAbout()

```
void printAbout ( )
```

Mostra as informações sobre o jogo

Definition at line 38 of file utils.c.

```
38                 {
39     printf("*****************************************************************\n");
40     printf("*Bem-vindo ao jogo de batalha naval! - Exemplo trabalho UFPR *\n");
41     printf("*****************************************************************\n");
42     printf("\n\n\n");
43     printf("Esse exemplo foi desenvolvido por Jackson Antonio do Prado Lima.\n\n\n");
44 }
```

2.2.1.14 printAlign()

```
void printAlign (
            int space,
            char * string )
```

Realiza o print de uma string de modo centralizado

Parameters

space	Espaço máximo em que a string deverá ser centralizada
string	String a ser mostrada

Definition at line 52 of file utils.c.

```
52                                      {
53      int length = strlen(string);
54
55     printf ("%*s%*c"
56             ,((space - length) >> 1) + length // string length + padding spaces
57             , string
58             , ((space - length) >> 1) + ((space - length) & 1) // tailing spaces
59             , ' '
60     );
61 }
```

2.2.1.15 printBattleship()

```
void printBattleship ( )
```

Mostra o nome do jogo de modo estilizado

Definition at line 66 of file utils.c.

```
66                        {
67     printf ("XXXXX   XXXX  XXXXXX XXXXXX XX     XXXXXX  XXXXX XX  XX XX XXXX\n");
68     printf ("XX  XX XX  XX   XX     XX   XX     XX     XX     XX  XX XX XX  XX\n");
69     printf ("XXXXX  XX  XX   XX     XX   XX     XXXX    XXXX  XXXXXX XX XXXX\n");
70     printf ("XX  XX XXXXXX   XX     XX   XX     XX         XX XX  XX XX XX\n");
71     printf ("XXXXX  XX  XX   XX     XX   XXXXXX XXXXXX XXXXX  XX  XX XX XX\n");
72     printf ("\n\n");
73 }
```

2.2.1.16 printMenu1()

```
void printMenu1 ( )
```

Mostra as opções de menu iniciais

Definition at line 78 of file utils.c.

```
78                  {
79     printf ("Escolha uma opção:\n");
80     printf ("1) Jogar\n");
81     printf ("2) Sobre o jogo\n");
82     printf ("3) Sair\n\n");
83 }
```

2.2.1.17 printMenu2()

```
void printMenu2 ( )
```

Mostra as opções de menu secundárias presentes ao clicar no menu inicial 1) Jogar

Definition at line 88 of file utils.c.

```
88                 {
89     printf ("Escolha uma opção:\n");
90     printf ("1) PvCOM\n");
91     printf ("2) PvP\n");
92     printf ("3) Voltar\n\n");
93 }
```

2.2.1.18 printResultados()

```
void printResultados (
            char * nomeJogador,
            char jogadas[][10],
            char tabuleiro[][10] )
```

Mostra o resultado do jogo de um jogador. O resultado apresenta: 1) Número total de disparos realizados; 2) Número de disparos na água; 3) Número de disparos em embarcações, por tipo de embarcação (na ordem inversa apresentada na tabela de embarcações, ou seja, começando pelo Destruidor), e omitindo os tipos para os quais não foi atingida qualquer embarcação; 4) Número de embarcações afundadas, por tipo de embarcação (na ordem inversa apresentada na tabela de embarcações, ou seja, começando pelo Destruidor), e omitindo os tipos para os quais não foi afundada qualquer embarcação.

Parameters

nomeJogador	Nome do jogador
jogadas	Jogadas realizadas pelo jogador
tabuleiro	Tabuleiro do jogador

Definition at line 741 of file utils.c.

```
741                                                                                 {
742     int totalDisparos = 0, totalFogo = 0, totalAgua = 0, totalAfundado = 0;
743     char embarcacoesAfundadas[5] = {'.','.','.','.','.'};
744     int embarcacoesAtingidas[5] = {0,0,0,0,0};
745 
746     for(int i = 0; i < 10; i++){
747         for(int j = 0; j < 10; j++){
748             // Obrigatoriamente conto um disparo, depois é verificado se a coordenada foi atingid
749             totalDisparos++;
750             switch(jogadas[i][j]){
751                 case '.':
752                     // Disparo não realizado
753                     totalDisparos--;
754                     break;
755                 case 'X':
756                     // Disparo que acertou alguma embarcação
757                     totalFogo++;
758                     // Verifico em qual embarcação atingiu e aumento o número de vezes em que ela foi
       atingida
759                     switch(tabuleiro[i][j]){
760                         case 'P':
```

```
761                             embarcacoesAtingidas[0] += 1;
762                             break;
763                         case 'N':
764                             embarcacoesAtingidas[1] += 1;
765                             break;
766                         case 'C':
767                             embarcacoesAtingidas[2] += 1;
768                             break;
769                         case 'S':
770                             embarcacoesAtingidas[3] += 1;
771                             break;
772                         case 'D':
773                             embarcacoesAtingidas[4] += 1;
774                             break;
775                     }
776                     break;
777                 case 'O':
778                     // Disparo na água
779                     totalAgua++;
780                     break;
781                 // Embarcações afundadas
782                 case 'P':
783                     if(embarcacoesAfundadas[0] == '.'){
784                         embarcacoesAfundadas[0] = 'P';
785                         totalAfundado++;
786                     }
787                     break;
788                 case 'N':
789                     if(embarcacoesAfundadas[1] == '.'){
790                         embarcacoesAfundadas[1] = 'N';
791                         totalAfundado++;
792                     }
793                     break;
794                 case 'C':
795                     if(embarcacoesAfundadas[2] == '.'){
796                         embarcacoesAfundadas[2] = 'C';
797                         totalAfundado++;
798                     }
799                     break;
800                 case 'S':
801                     if(embarcacoesAfundadas[3] == '.'){
802                         embarcacoesAfundadas[3] = 'S';
803                         totalAfundado++;
804                     }
805                     break;
806                 case 'D':
807                     if(embarcacoesAfundadas[4] == '.'){
808                         embarcacoesAfundadas[4] = 'D';
809                         totalAfundado++;
810                     }
811                     break;
812             }
813 
814         }
815     }
816 
817     printf("%s, seu resultado foi: \n", nomeJogador);
818     printf("Número total de disparos realizados: %d\n", totalDisparos);
819     printf("Número total de disparos na água: %d\n", totalAgua);
820 
821 
822     if(totalAfundado != 5){
823         int valor = 0;
824 
825         for(int i = 0; i < 5; valor+=embarcacoesAtingidas[i], i++);
826 
827         printf("\nNúmero de disparos em embarcações: %d\n", valor);
828 
829         if((totalDisparos - totalAgua) > 0){
830             for(int i = 4; i >= 0; i--){
831                     char shipType;
832 
833                     switch(i){
834                         case 0:
835                             shipType = 'P';
836                             break;
837                         case 1:
838                             shipType = 'N';
839                             break;
840                         case 2:
841                             shipType = 'C';
842                             break;
843                         case 3:
844                             shipType = 'S';
845                             break;
846                         case 4:
847                             shipType = 'D';
```

```
848                             break;
849                     }
850 
851                     if(embarcacoesAtingidas[i]!= 0){
852                         printf("\t%s, atingida %d %s\n", shipName(shipType), embarcacoesAtingidas[i
    ], embarcacoesAtingidas[i] == 1 ? "vez" : "vezes");
853                     }
854             }
855         }
856     }
857 
858     printf("\nNúmero de embarcações afundadas: %d\n", totalAfundado);
859 
860     if(totalAfundado > 0){
861         printf("Embarcações afundadas: \n");
862         for(int i = 4; i >= 0; i--){
863                 if(embarcacoesAfundadas[i]!= '.'){
864                     printf("\t%s\n", shipName(embarcacoesAfundadas[i]));
865                 }
866         }
867     }
868 }
```

2.2.1.19 putShip()

```
void putShip (
              char shipType,
              char line,
              int col,
              char direction,
              char tabuleiro[][10] )
```

Coloca uma embarcação no tabulerio

Parameters

shipType	Tipo de embarcação
line	Linha da matriz
col	Coluna da matriz
direction	Direção do posicionamento da embarcação
tabuleiro	Tabuleiro do jogo

Definition at line 517 of file utils.c.

```
517                                                                                                 {
518     int intLine = (int)line-65;
519     switch(direction){
520         case 'W':
521             //"Para Cima"
522             for(int i = intLine; i > (intLine - getShipSize(shipType)); i--){
523                 tabuleiro[i][col] = shipType;
524             }
525             break;
526         case 'S':
527             //"Para Baixo"
528             for(int i = intLine; i < (intLine + getShipSize(shipType)); i++){
529                 tabuleiro[i][col] = shipType;
530             }
531             break;
532         case 'D':
533             //"Para Direita"
534             for(int i = col; i < (col + getShipSize(shipType)); i++){
535                 tabuleiro[intLine][i] = shipType;
536             }
537             break;
```

```
538         case 'A':
539             //"Para Esquerda"
540             for(int i = col; i > (col - getShipSize(shipType)); i--){
541                 tabuleiro[intLine][i] = shipType;
542             }
543             break;
544     }
545 }
```

2.2.1.20 putShips()

```
void putShips (
            char * nomeJogador,
            char tabuleiro[][10] )
```

Coloca as embarcações de um jogador no tabulerio.

Parameters

nomeJogador	Nome do jogador
tabuleiro	Tabuleiro do jogador

Definition at line 553 of file utils.c.

```
553                                                     {
554     printf("\n %s, agora posicione as suas embarcações no tabuleiro.\n\n", nomeJogador);
555 
556     char chooseSelection = 'S';
557 
558     do{
559         initMatrix(tabuleiro);
560 
561         for(int i = 0; i < 5; i++){
562             char shipType;
563 
564             switch(i){
565                 case 0:
566                     shipType = 'P';
567                     break;
568                 case 1:
569                     shipType = 'N';
570                     break;
571                 case 2:
572                     shipType = 'C';
573                     break;
574                 case 3:
575                     shipType = 'S';
576                     break;
577                 case 4:
578                     shipType = 'D';
579                     break;
580             }
581 
582             do{
583                 // Valores default (só pra iniciar com algo)
584                 char line = 'A';
585                 int col = 0;
586                 char direction = 'W';
587 
588                 printf("Você está posicionando a embarcação \"%s\".\n", shipName(shipType));
589                 readCoordenate(&line, &col);
590                 printf("Informe a direção (W,S,A,D): ");
591                 scanf(" %c", &direction);
592 
593                 if(checkCoordenate(shipType, line, col, direction, tabuleiro)){
594                     char confirmation = 'S';
595                     printf("\nPressione S para confirmar a coordenada %c-%i com direção \"%s\" para a
    embarcação \"%s\".\n", line, col, directionMessage(direction),
```

```
    shipName(shipType));
596
597                     do{
598                         scanf(" %c", &confirmation);
599                         if(confirmation != 'S' && confirmation != 'N'){
600                             printf("Opção inválida! Tente novamente.\n");
601                         }
602                     }while(confirmation != 'S' && confirmation != 'N');
603
604                     if(confirmation == 'S'){
605                         putShip(shipType, line, col, direction, tabuleiro);
606                         break;
607                     }
608                 }else{
609                     printf("\nCoordenada inválida! Tente novamente.\n");
610                 }
611             }while(1);
612
613             clearScreen();
614             printBattleship();
615             displayGamePlayer(nomeJogador, tabuleiro);
616         }
617         printf("\n\nEmbarcações posicionadas! Pressione S para confirmar as posições ou N para reiniciar: "
    );
618         do{
619             scanf(" %c", &chooseSelection);
620             if(chooseSelection != 'S' && chooseSelection != 'N'){
621                 printf("Opção inválida! Tente novamente.\n");
622             }
623         }while(chooseSelection != 'S' && chooseSelection != 'N');
624     }while(chooseSelection != 'S');
625 }
```

2.2.1.21 putShipsCOM()

```
void putShipsCOM (
             char tabuleiro[][10] )
```

Coloca as embarcações da "máquina" no tabulerio. Aleatoriamente é selecionado um tabuleiro para a máquina.

Parameters

tabuleiro	Tabuleiro da máquina

Definition at line 633 of file utils.c.

```
633                                     {
634     // Como é complicado posicionar aleatoriamente e cheio de regras eu vou fazer uns jogos "fixos" e
    selecionar aleatoriamente um deles
635     srand(time(NULL));
636     int game = rand() % 3;
637     char shipType, line, direction;
638     int col;
639
640     initMatrix(tabuleiro);
641
642     switch(game){
643         case 0:
644             // Posicionar o Porta-aviões
645             shipType = 'P';
646             line = 'A', col = 0, direction = 'D';
647             putShip(shipType, line, col, direction, tabuleiro);
648
649             // Posicionar o Embarcação de Guerra
650             shipType = 'N';
651             line = 'F', col = 1, direction = 'S';
652             putShip(shipType, line, col, direction, tabuleiro);
653
654             // Posicionar o Cruzador
655             shipType = 'C';
```

```
656             line = 'E', col = 6, direction = 'A';
657             putShip(shipType, line, col, direction, tabuleiro);
658 
659             // Posicionar o Submarino
660             shipType = 'S';
661             line = 'H', col = 6, direction = 'W';
662             putShip(shipType, line, col, direction, tabuleiro);
663 
664             // Posicionar o Destruidor
665             shipType = 'D';
666             line = 'D', col = 6, direction = 'D';
667             putShip(shipType, line, col, direction, tabuleiro);
668             break;
669         case 1:
670             // Posicionar o Porta-aviões
671             shipType = 'P';
672             line = 'I', col = 1, direction = 'W';
673             putShip(shipType, line, col, direction, tabuleiro);
674 
675             // Posicionar o Embarcação de Guerra
676             shipType = 'N';
677             line = 'C', col = 7, direction = 'A';
678             putShip(shipType, line, col, direction, tabuleiro);
679 
680             // Posicionar o Cruzador
681             shipType = 'C';
682             line = 'H', col = 5, direction = 'S';
683             putShip(shipType, line, col, direction, tabuleiro);
684 
685             // Posicionar o Submarino
686             shipType = 'S';
687             line = 'A', col = 9, direction = 'S';
688             putShip(shipType, line, col, direction, tabuleiro);
689 
690             // Posicionar o Destruidor
691             shipType = 'D';
692             line = 'C', col = 3, direction = 'S';
693             putShip(shipType, line, col, direction, tabuleiro);
694             break;
695         case 2:
696             // Posicionar o Porta-aviões
697             shipType = 'P';
698             line = 'A', col = 0, direction = 'D';
699             putShip(shipType, line, col, direction, tabuleiro);
700 
701             // Posicionar o Embarcação de Guerra
702             shipType = 'N';
703             line = 'C', col = 1, direction = 'D';
704             putShip(shipType, line, col, direction, tabuleiro);
705 
706             // Posicionar o Cruzador
707             shipType = 'C';
708             line = 'E', col = 3, direction = 'D';
709             putShip(shipType, line, col, direction, tabuleiro);
710 
711             // Posicionar o Submarino
712             shipType = 'S';
713             line = 'G', col = 7, direction = 'W';
714             putShip(shipType, line, col, direction, tabuleiro);
715 
716             // Posicionar o Destruidor
717             shipType = 'D';
718             line = 'J', col = 4, direction = 'D';
719             putShip(shipType, line, col, direction, tabuleiro);
720             break;
721     }
722 }
```

2.2.1.22 readCoordenate()

```
void readCoordenate (
            char * line,
            int * col )
```

Realiza a leitura de uma coordenada para o tabuleiro. A leitura continua até que uma coordenada válida seja informada.

Parameters

∗line	Ponteiro de uma variável para ser a linha da matriz
∗col	Ponteiro de uma variável para ser a coluna da matriz

Definition at line 230 of file utils.c.

```
230                                         {
231     int intLine;
232     do{
233         printf("Informe a linha (A-J): ");
234         scanf(" %c", line);
235         printf("Informe a coluna (0-9): ");
236         scanf("%i", col);
237 
238          intLine= (int)*line-65;
239         // Verifica se a coordenada é válida
240         if(!checkThreshold(intLine, *col)){
241             printf("\nCoordenada inválida! Tente novamente.\n");
242         }
243     }while(!checkThreshold(intLine, *col));
244 }
```

2.2.1.23 shipName()

```
const char* shipName (
              char ship )
```

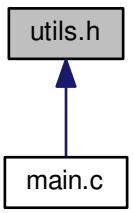
Retorna o nome da embarcação de acordo com a sigla

Definition at line 114 of file utils.c.

```
114                                 {
115     switch(ship){
116         case 'P':
117             return "Porta-aviões";
118         case 'N':
119             return "Embarcação de Guerra";
120         case 'C':
121             return "Cruzador";
122         case 'S':
123             return "Submarino";
124         case 'D':
125             return "Destruidor";
126     }
127 }
```

2.3 utils.h File Reference

This graph shows which files directly or indirectly include this file:



Functions

- void clearScreen ()
- void initMatrix (char tabuleiro[ ][10])
- void printAbout ()
- void printAlign (int space, char ∗string)
- void printBattleship ()
- void printMenu1 ()
- void printMenu2 ()
- const char ∗ directionMessage (char direction)
- const char ∗ shipName (char ship)
- const int getShipSize (char ship)
- int checkThreshold (char line, int col)
- int checkCoordenate (char shipType, char line, int col, char direction, char tabuleiro[ ][10])
- void readCoordenate (char ∗line, int ∗col)
- void checkShot (char jogadas[ ][10], char tabuleiroAdversario[ ][10], char ∗nomeJogador, char ∗nome↩ Adversario, int line, int col)
- void displayGameInfo (char ∗nomeJogador1, char jogadas1[ ][10], char ∗nomeJogador2, char jogadas2[ ][10])
- void displayGamePlayer (char ∗nomeJogador, char tabuleiro[ ][10])
- int isEndGame (char jogadas[ ][10])
- void giveShot (char jogadas[ ][10], char tabuleiroAdversario[ ][10], char ∗nomeJogador, char ∗nome↩ Adversario)
- void giveRandomShot (char jogadas[ ][10], char tabuleiroAdversario[ ][10], char ∗nomeJogador, char ∗nomeAdversario)
- void putShip (char shipType, char line, int col, char direction, char tabuleiro[ ][10])
- void putShips (char ∗nomeJogador, char tabuleiro[ ][10])
- void putShipsCOM (char tabuleiro[ ][10])
- void printResultados (char ∗nomeJogador, char jogadas[ ][10], char tabuleiro[ ][10])

2.3.1 Function Documentation

2.3.1.1 checkCoordenate()

```
int checkCoordenate (
            char shipType,
            char line,
            int col,
            char direction,
            char tabuleiro[][10] )
```

Verifica se a coordenada é válida para inserção de uma embarcação no tabuleiro

Parameters

shipType	Tipo de embarcação
line	Linha da matriz
col	Coluna da matriz
direction	Direção do posicionamento da embarcação
tabuleiro	Tabuleiro do jogo

Returns

1 se a coordenada é válido, 0 caso contrário

Verifica se a coordenada juntamente com a direção são válidas para inserção de uma embarcação no tabuleiro

Parameters

shipType	Tipo de embarcação
line	Linha da matriz
col	Coluna da matriz
direction	Direção do posicionamento da embarcação
tabuleiro	Tabuleiro do jogo

Returns

1 se a coordenada é válida, 0 caso contrário

Definition at line 168 of file utils.c.

```
168                                                                                                 {
169     int intLine = (int)line-65;
170     /* Verifico:
171      * - se é uma coordenada dentro da matriz
172      * - se a letra informada para a direção é válida
173      * - se a direção e a coordenada ficam dentro da matriz
174     */
175     if(!checkThreshold(intLine, col)
176         || !(direction == 'W' || direction == 'S' || direction == 'A' || direction == 'D')
177         || (direction == 'W' && (intLine - getShipSize(shipType) < 0))
178         || (direction == 'S' && (intLine + getShipSize(shipType) > 10))
179         || (direction == 'A' && (col - getShipSize(shipType) < 0))
180         || (direction == 'D' && (col + getShipSize(shipType) > 10))){
181         return 0;
182     }
183 
184     // Verifico se as posições em que a embarcação será colocada estão livres
185     switch(direction){
186         case 'W':
187             //"Para Cima"
188             for(int i = intLine; i > (intLine - getShipSize(shipType)); i--){
189                 if(tabuleiro[i][col] != '.'){
190                     return 0;
191                 }
192             }
193             break;
194         case 'S':
195             //"Para Baixo"
196             for(int i = intLine; i < (intLine + getShipSize(shipType)); i++){
197                 if(tabuleiro[i][col] != '.'){
198                     return 0;
199                 }
200             }
201             break;
202         case 'D':
203             //"Para Direita"
204             for(int i = col; i < (col + getShipSize(shipType)); i++){
205                 if(tabuleiro[intLine][i] != '.'){
206                     return 0;
207                 }
208             }
209             break;
210         case 'A':
211             //"Para Esquerda"
212             for(int i = col; i > (col - getShipSize(shipType)); i--){
213                 if(tabuleiro[intLine][i] != '.'){
214                     return 0;
215                 }
216             }
217             break;
218     }
219 
220     return 1;
221 }
```

2.3.1.2 checkShot()

```
void checkShot (
            char jogadas[][10],
            char tabuleiroAdversario[][10],
            char * nomeJogador,
            char * nomeAdversario,
            int line,
            int col )
```

Verifica o resultado de uma jogada (disparo) da batalha naval

Parameters

jogadas	Jogadas realizadas pelo jogador até o momento.
tabuleiroAdversario	Tabuleiro do adversário
nomeJogador	Nome do jogador
nomeAdversario	Nome do adversário

Definition at line 254 of file utils.c.

```
254
                    {
255     clearScreen();
256     printBattleship();
257     printf("\n\n");
258
259     printf("Resultado do disparo realizado por: %s\n\n", nomeJogador);
260
261     printf("*****************************************************************\n");
262     // Verifico onde o disparo acertou e atribuo o indicador
263     if(tabuleiroAdversario[line][col] == '.'){
264         jogadas[line][col] = 'O';
265         printf("* Água! \n");
266     }else{
267         printf("* Fogo! \n");
268         jogadas[line][col] = 'X';
269
270         char shipType = tabuleiroAdversario[line][col]; // Guardo o tipo de embarcação atingida
271         int contHits = 0; // Quantas vezes essa embarcação foi atingida
272
273         // Verifico se minha embarcação afundou
274         for(int i = 0; i < 10; i ++){
275             for(int j = 0; j < 10; j++){
276                 // Se há um disparo para minha embarcação, acumulo o número de vezes que ela foi acertada
277                 if(jogadas[i][j] == 'X' && tabuleiroAdversario[i][j] == shipType){
278                     contHits++;
279                 }
280             }
281         }
282
283         // Se acertei o número de vezes igual ao tamanho dela, significa que afundei a embarcação
284         if(contHits == getShipSize(shipType)){
285             printf("* %s: Afundou %s %s!\n", nomeAdversario, shipType == 'N' ? "minha" : "meu",
    shipName(shipType));
286
287             // Mostro no tabuleiro minha embarcação
288             for(int i = 0; i < 10; i ++){
289                 for(int j = 0; j < 10; j++){
290                     // Procuro as posições da embarcação afundada e coloco a sigla dela para ser mostrada
291                     if(tabuleiroAdversario[i][j] == shipType){
292                         jogadas[i][j] = shipType;
293                     }
294                 }
295             }
296         }
297     }
298     printf("*****************************************************************\n");
299     printf("\n\n\n");
300 }
```

2.3.1.3 checkThreshold()

```
int checkThreshold (
            char line,
            int col )
```

Verifica se a coordenada é válidas para inserção de uma embarcação no tabuleiro

Parameters

line	Linha da matriz
col	Coluna da matriz

Returns

1 se a coordenada é válida, 0 caso contrário

2.3.1.4 clearScreen()

```
void clearScreen ( )
```

Limpa a tela do console

Definition at line 18 of file utils.c.

```
18                  {
19   system("@cls||clear");
20 }
```

2.3.1.5 directionMessage()

```
const char* directionMessage (
            char direction )
```

Retorna a mensagem correspondende à cada direção

Definition at line 98 of file utils.c.

```
98                                          {
99     switch(direction){
100         case 'W':
101             return "Para Cima";
102         case 'S':
103             return "Para Baixo";
104         case 'D':
105             return "Para Direita";
106         case 'A':
107             return "Para Esquerda";
108     }
109 }
```

2.3.1.6 displayGameInfo()

```
void displayGameInfo (
            char * nomeJogador1,
            char jogadas1[][10],
            char * nomeJogador2,
            char jogadas2[][10] )
```

Mostra as informações de duas matrizes lado a lado com o nome dos jogadores. Exemplo: Player | Computer 0 1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 A |P| | | | | | | | | | A | | | | | | | | | | B |P| | | | | | | | | | B | | | | | | | | | | C |P| | | | | | | | | | C | | | | | | |O| | | | D |P| | | | | | | | | | D | | |x| | | | | | | E |P| | | | | | | | | | E | | | | | | | | | | F | | | | | | | | | | | F | | | | |O| | | | | G | | | | | | | | | | | G | | | | | | | | | | H | | | | | | | | | | | H | |O| | | | |O| | | | I | | | | | | | | | | | | I | | | | | | | | | | | J | | | | | | | | | | | | J | | | | | | | | | | |

Parameters

nomeJogador1	Nome do jogador 1
jogadas1	Tabuleiro de jogadas do jogador 1
nomeJogador2	Nome do jogador 2
jogadas2	Tabuleiro de jogadas do jogador 2

Mostra as informações dos tabuleiros lado a lado de dois jogadores com os seus respectivos nomes. Exemplo: Player | Computer 0 1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 A |P| | | | | | | | | | A | | | | | | | | | | B |P| | | | | | | | | | B | | | | | | | | | | C |P| | | | | | | | | | C | | | | | | |O| | | | D |P| | | | | | | | | | D | | |x| | | | | | | E |P| | | | | | | | | | E | | | | | | | | | | F | | | | | | | | | | | F | | | | |O| | | | | G | | | | | | | | | | | G | | | | | | | | | | H | | | | | | | | | | | H | |O| | | | |O| | | | I | | | | | | | | | | | | I | | | | | | | | | | | J | | | | | | | | | | | | J | | | | | | | | | | |

Parameters

nomeJogador1	Nome do jogador 1
jogadas1	Tabuleiro de jogadas do jogador 1
nomeJogador2	Nome do jogador 2
jogadas2	Tabuleiro de jogadas do jogador 2

Definition at line 323 of file utils.c.

```
323                                                                                                 {
324     char separador[] = "     |   ";
325
326     // Imprimo o cabeçalho
327     printAlign(41, nomeJogador1);
328     printf("%s", separador);
329     printAlign(44, nomeJogador2);
330     printf("\n");
331
332     for(int cont = 0; cont < 2; cont++){
333         printf(" ");
334         for (int i = 0; i < 10; i++) {
335             printf("   %i", i);
336         }
337         if(!cont){
338             printf("%s", separador);
339         }
340     }
341     printf("\n");
342
343     // Imprimo as linhas
344     for (int i = 0; i < 10; i++) {
345         // Imprime a letra
346         printf("%c | ", 65+i);
347
```

```
348         // Imprime os valores do player1
349         for (int j = 0; j < 10; j++) {
350             printf("%c | ", jogadas1[i][j]);
351         }
352 
353         // Imprime a letra
354         printf("  |   %c | ", 65+i);
355         // Imprime os valores do player 2
356         for (int j = 0; j < 10; j++) {
357             printf("%c | ", jogadas2[i][j]);
358         }
359         printf("\n");
360     }
361 }
```

2.3.1.7 displayGamePlayer()

```
void displayGamePlayer (
              char * nomeJogador,
              char tabuleiro[][10] )
```

Mostra as informações do tabuleiro do jogador com os seu respectivo nome. Exemplo: Player 0 1 2 3 4 5 6 7 8 9 A |P| | | | | | | | | | B |P| | | | | | | | | | C |P| | | | | | | | | | D |P| | | | | | | | | | E |P| | | | | | | | | | F | | | | | | | | | | | G | | | | |N|N|N| | | | H | | | | | | | | | | | I | | | | | | | | | | | J | | | | | | | | | | |

Parameters

nomeJogador	Nome do jogador
tabuleiro	Tabuleiro do jogador

Definition at line 382 of file utils.c.

```
382                                                                 {
383     char separador[] = "     |   ";
384 
385     // Imprimo o cabeçalho
386     printAlign(41, nomeJogador);
387     printf("\n");
388 
389     printf(" ");
390     for (int i = 0; i < 10; i++) {
391         printf("   %i", i);
392     }
393     printf("\n");
394 
395     // Imprimo as linhas
396     for (int i = 0; i < 10; i++) {
397         // Imprime a letra
398         printf("%c | ", 65+i);
399 
400         // Imprime os valores do jogador
401         for (int j = 0; j < 10; j++) {
402             printf("%c | ", tabuleiro[i][j]);
403         }
404         printf("\n");
405     }
406 }
```

2.3.1.8 getShipSize()

```
const int getShipSize (
            char ship )
```

Retorna o tamanho da embarcação de acordo com a sigla

Definition at line 132 of file utils.c.

```
132                                 {
133     switch(ship){
134         case 'P':
135             return 5;
136         case 'N':
137             return 4;
138         case 'C':
139             return 3;
140         case 'S':
141             return 3;
142         case 'D':
143             return 2;
144     }
145 }
```

2.3.1.9 giveRandomShot()

```
void giveRandomShot (
            char jogadas[][10],
            char tabuleiroAdversario[][10],
            char * nomeJogador,
            char * nomeAdversario )
```

A máquina realiza uma jogada (disparo) da batalha naval

Parameters

jogadas	Jogadas realizadas pelo jogador até o momento.
tabuleiroAdversario	Tabuleiro do adversário
nomeJogador	Nome do jogador
nomeAdversario	Nome do adversário

Definition at line 492 of file utils.c.

```
492
          {
493     int col, line;
494
495     do{
496         srand(time(NULL));
497         line = rand() % 10;
498         col = rand() % 10;
499
500         if(jogadas[line][col] == '.'){
501             break;
502         }
503     }while(1);
504
505     checkShot(jogadas, tabuleiroAdversario, nomeJogador, nomeAdversario, line, col);
506 }
```

2.3.1.10 giveShot()

```
void giveShot (
            char jogadas[ ][10],
            char tabuleiroAdversario[ ][10],
            char * nomeJogador,
            char * nomeAdversario )
```

Realiza uma jogada (disparo) da batalha naval por um jogador

Parameters

jogadas	Jogadas realizadas pelo jogador até o momento.
tabuleiroAdversario	Tabuleiro do adversário
nomeJogador	Nome do jogador
nomeAdversario	Nome do adversário

Definition at line 464 of file utils.c.

```
464                                                                                                      {
465     int col, intLine;
466     char line;
467 
468     // O programa solicita um disparo válido
469     do{
470         readCoordenate(&line, &col);
471 
472         intLine = (int)line-65;
473 
474         if(jogadas[intLine][col] == '.'){
475             break;
476         }else{
477             printf("Essa coordenada já foi informada anteriormente. Tente novamente.\n");
478         }
479     }while(1);
480 
481     checkShot(jogadas, tabuleiroAdversario, nomeJogador, nomeAdversario, intLine, col);
482 }
```

2.3.1.11 initMatrix()

```
void initMatrix (
            char tabuleiro[ ][10] )
```

Inicializo cada posição do tabuleiro do jogador com '.'

Parameters

tabuleiro	Tabuleiro do jogador

Definition at line 27 of file utils.c.

```
27                                        {
28     for (int i = 0; i < 10; i++) {
29       for (int j = 0; j < 10; j++) {
```

```
30          tabuleiro[i][j] = '.';
31        }
32      }
33 }
```

2.3.1.12 isEndGame()

```
int isEndGame (
            char jogadas[][10] )
```

Verifica se o jogo já acabou

Parameters

jogadas	Tabuleiro de jogadas de algum jogador

Returns

Retorna 1 se todas as embarcações foram afundadas, 0 caso contrário

Definition at line 414 of file utils.c.

```
414                                 {
415     int embarcacoesAfundadas = 0;
416 
417     // Verifico se todas as embarcações foram afundadas
418     for(int i = 0; i < 5; i++){
419 
420         char shipType;
421 
422         switch(i){
423             case 0:
424                 shipType = 'P';
425                 break;
426             case 1:
427                 shipType = 'N';
428                 break;
429             case 2:
430                 shipType = 'C';
431                 break;
432             case 3:
433                 shipType = 'S';
434                 break;
435             case 4:
436                 shipType = 'D';
437                 break;
438         }
439 
440         // Se o tabuleiro contém a letra da embarcação significa que essa embarcação foi afundada
441         for(int i = 0; i < 10; i++){
442             for(int j = 0; j < 10; j++){
443                 if(jogadas[i][j] == shipType){
444                     embarcacoesAfundadas++;
445                     i = j = 10;
446                     break;
447                 }
448             }
449         }
450     }
451 
452 
453     return embarcacoesAfundadas == 5 ? 1 : 0;
454 }
```

2.3.1.13 printAbout()

```
void printAbout ( )
```

Mostra as informações sobre o jogo

Definition at line 38 of file utils.c.

```
38                 {
39     printf("*************************************************************\n");
40     printf("*Bem-vindo ao jogo de batalha naval! - Exemplo trabalho UFPR *\n");
41     printf("*************************************************************\n");
42     printf("\n\n\n");
43     printf("Esse exemplo foi desenvolvido por Jackson Antonio do Prado Lima.\n\n\n");
44 }
```

2.3.1.14 printAlign()

```
void printAlign (
            int space,
            char * string )
```

Realiza o print de uma string de modo centralizado

Parameters

space	Espaço máximo em que a string deverá ser centralizada
string	String a ser mostrada

Definition at line 52 of file utils.c.

```
52                                      {
53      int length = strlen(string);
54
55     printf ("%*s%*c"
56             ,((space - length) >> 1) + length // string length + padding spaces
57             , string
58             , ((space - length) >> 1) + ((space - length) & 1) // tailing spaces
59             , ' '
60     );
61 }
```

2.3.1.15 printBattleship()

```
void printBattleship ( )
```

Mostra o nome do jogo de modo estilizado

Definition at line 66 of file utils.c.

```
66                      {
67     printf ("XXXXX   XXXX  XXXXXX XXXXXX XX     XXXXXX  XXXXX XX  XX XX XXXX\n");
68     printf ("XX  XX XX  XX   XX     XX   XX     XX     XX     XX  XX XX XX  XX\n");
69     printf ("XXXXX  XX  XX   XX     XX   XX     XXXX    XXXX  XXXXXX XX XXXX\n");
70     printf ("XX  XX XXXXXX   XX     XX   XX     XX         XX XX  XX XX XX\n");
71     printf ("XXXXX  XX  XX   XX     XX   XXXXXX XXXXXX XXXXX  XX  XX XX XX\n");
72     printf ("\n\n");
73 }
```

2.3.1.16 printMenu1()

```
void printMenu1 ( )
```

Mostra as opções de menu iniciais

Definition at line 78 of file utils.c.

```
78                 {
79     printf ("Escolha uma opção:\n");
80     printf ("1) Jogar\n");
81     printf ("2) Sobre o jogo\n");
82     printf ("3) Sair\n\n");
83 }
```

2.3.1.17 printMenu2()

```
void printMenu2 ( )
```

Mostra as opções de menu secundárias presentes ao clicar no menu inicial 1) Jogar

Definition at line 88 of file utils.c.

```
88                 {
89     printf ("Escolha uma opção:\n");
90     printf ("1) PvCOM\n");
91     printf ("2) PvP\n");
92     printf ("3) Voltar\n\n");
93 }
```

2.3.1.18 printResultados()

```
void printResultados (
             char * nomeJogador,
             char jogadas[][10],
             char tabuleiro[][10] )
```

Mostra o resultado do jogo de um jogador. O resultado apresenta: 1) Número total de disparos realizados; 2) Número de disparos na água; 3) Número de disparos em embarcações, por tipo de embarcação (na ordem inversa apresentada na tabela de embarcações, ou seja, começando pelo Destruidor), e omitindo os tipos para os quais não foi atingida qualquer embarcação; 4) Número de embarcações afundadas, por tipo de embarcação (na ordem inversa apresentada na tabela de embarcações, ou seja, começando pelo Destruidor), e omitindo os tipos para os quais não foi afundada qualquer embarcação.

Parameters

nomeJogador	Nome do jogador
jogadas	Jogadas realizadas pelo jogador
tabuleiro	Tabuleiro do jogador

Definition at line 741 of file utils.c.

```
741                                                                                 {
742     int totalDisparos = 0, totalFogo = 0, totalAgua = 0, totalAfundado = 0;
743     char embarcacoesAfundadas[5] = {'.','.','.','.','.'};
744     int embarcacoesAtingidas[5] = {0,0,0,0,0};
745 
746     for(int i = 0; i < 10; i++){
747         for(int j = 0; j < 10; j++){
748             // Obrigatoriamente conto um disparo, depois é verificado se a coordenada foi atingid
749             totalDisparos++;
750             switch(jogadas[i][j]){
751                 case '.':
752                     // Disparo não realizado
753                     totalDisparos--;
754                     break;
755                 case 'X':
756                     // Disparo que acertou alguma embarcação
757                     totalFogo++;
758                     // Verifico em qual embarcação atingiu e aumento o número de vezes em que ela foi
    atingida
759                     switch(tabuleiro[i][j]){
760                         case 'P':
761                             embarcacoesAtingidas[0] += 1;
762                             break;
763                         case 'N':
764                             embarcacoesAtingidas[1] += 1;
765                             break;
766                         case 'C':
767                             embarcacoesAtingidas[2] += 1;
768                             break;
769                         case 'S':
770                             embarcacoesAtingidas[3] += 1;
771                             break;
772                         case 'D':
773                             embarcacoesAtingidas[4] += 1;
774                             break;
775                     }
776                     break;
777                 case 'O':
778                     // Disparo na água
779                     totalAgua++;
780                     break;
781                 // Embarcações afundadas
782                 case 'P':
783                     if(embarcacoesAfundadas[0] == '.'){
784                         embarcacoesAfundadas[0] = 'P';
785                         totalAfundado++;
786                     }
787                     break;
788                 case 'N':
789                     if(embarcacoesAfundadas[1] == '.'){
790                         embarcacoesAfundadas[1] = 'N';
791                         totalAfundado++;
792                     }
793                     break;
794                 case 'C':
795                     if(embarcacoesAfundadas[2] == '.'){
796                         embarcacoesAfundadas[2] = 'C';
797                         totalAfundado++;
798                     }
799                     break;
800                 case 'S':
801                     if(embarcacoesAfundadas[3] == '.'){
802                         embarcacoesAfundadas[3] = 'S';
803                         totalAfundado++;
804                     }
805                     break;
806                 case 'D':
807                     if(embarcacoesAfundadas[4] == '.'){
808                         embarcacoesAfundadas[4] = 'D';
809                         totalAfundado++;
810                     }
811                     break;
```

```
812             }
813
814         }
815     }
816
817     printf("%s, seu resultado foi: \n", nomeJogador);
818     printf("Número total de disparos realizados: %d\n", totalDisparos);
819     printf("Número total de disparos na água: %d\n", totalAgua);
820
821
822     if(totalAfundado != 5){
823         int valor = 0;
824
825         for(int i = 0; i < 5; valor+=embarcacoesAtingidas[i], i++);
826
827         printf("\nNúmero de disparos em embarcações: %d\n", valor);
828
829         if((totalDisparos - totalAgua) > 0){
830             for(int i = 4; i >= 0; i--){
831                     char shipType;
832
833                     switch(i){
834                         case 0:
835                             shipType = 'P';
836                             break;
837                         case 1:
838                             shipType = 'N';
839                             break;
840                         case 2:
841                             shipType = 'C';
842                             break;
843                         case 3:
844                             shipType = 'S';
845                             break;
846                         case 4:
847                             shipType = 'D';
848                             break;
849                     }
850
851                     if(embarcacoesAtingidas[i]!= 0){
852                         printf("\t%s, atingida %d %s\n", shipName(shipType), embarcacoesAtingidas[i
    ], embarcacoesAtingidas[i] == 1 ? "vez" : "vezes");
853                     }
854             }
855         }
856     }
857
858     printf("\nNúmero de embarcações afundadas: %d\n", totalAfundado);
859
860     if(totalAfundado > 0){
861         printf("Embarcações afundadas: \n");
862         for(int i = 4; i >= 0; i--){
863                 if(embarcacoesAfundadas[i]!= '.'){
864                     printf("\t%s\n", shipName(embarcacoesAfundadas[i]));
865                 }
866         }
867     }
868 }
```

2.3.1.19 putShip()

```
void putShip (
              char shipType,
              char line,
              int col,
              char direction,
              char tabuleiro[][10] )
```

Coloca uma embarcação no tabulerio

Parameters

shipType	Tipo de embarcação

Parameters

line	Linha da matriz
col	Coluna da matriz
direction	Direção do posicionamento da embarcação
tabuleiro	Tabuleiro do jogo

Definition at line 517 of file utils.c.

```
517                                                                                                 {
518     int intLine = (int)line-65;
519     switch(direction){
520         case 'W':
521             //"Para Cima"
522             for(int i = intLine; i > (intLine - getShipSize(shipType)); i--){
523                 tabuleiro[i][col] = shipType;
524             }
525             break;
526         case 'S':
527             //"Para Baixo"
528             for(int i = intLine; i < (intLine + getShipSize(shipType)); i++){
529                 tabuleiro[i][col] = shipType;
530             }
531             break;
532         case 'D':
533             //"Para Direita"
534             for(int i = col; i < (col + getShipSize(shipType)); i++){
535                 tabuleiro[intLine][i] = shipType;
536             }
537             break;
538         case 'A':
539             //"Para Esquerda"
540             for(int i = col; i > (col - getShipSize(shipType)); i--){
541                 tabuleiro[intLine][i] = shipType;
542             }
543             break;
544     }
545 }
```

2.3.1.20 putShips()

```
void putShips (
             char * nomeJogador,
             char tabuleiro[][10] )
```

Coloca as embarcações de um jogador no tabulerio.

Parameters

nomeJogador	Nome do jogador
tabuleiro	Tabuleiro do jogador

Definition at line 553 of file utils.c.

```
553                                                     {
554     printf("\n %s, agora posicione as suas embarcações no tabuleiro.\n\n", nomeJogador);
555
556     char chooseSelection = 'S';
557
558     do{
```

```
559         initMatrix(tabuleiro);
560 
561         for(int i = 0; i < 5; i++){
562             char shipType;
563 
564             switch(i){
565                 case 0:
566                     shipType = 'P';
567                     break;
568                 case 1:
569                     shipType = 'N';
570                     break;
571                 case 2:
572                     shipType = 'C';
573                     break;
574                 case 3:
575                     shipType = 'S';
576                     break;
577                 case 4:
578                     shipType = 'D';
579                     break;
580             }
581 
582             do{
583                 // Valores default (só pra iniciar com algo)
584                 char line = 'A';
585                 int col = 0;
586                 char direction = 'W';
587 
588                 printf("Você está posicionando a embarcação \"%s\".\n", shipName(shipType));
589                 readCoordenate(&line, &col);
590                 printf("Informe a direção (W,S,A,D): ");
591                 scanf(" %c", &direction);
592 
593                 if(checkCoordenate(shipType, line, col, direction, tabuleiro)){
594                     char confirmation = 'S';
595                     printf("\nPressione S para confirmar a coordenada %c-%i com direção \"%s\" para a
     embarcação \"%s\".\n", line, col, directionMessage(direction),
    shipName(shipType));
596 
597                     do{
598                         scanf(" %c", &confirmation);
599                         if(confirmation != 'S' && confirmation != 'N'){
600                             printf("Opção inválida! Tente novamente.\n");
601                         }
602                     }while(confirmation != 'S' && confirmation != 'N');
603 
604                     if(confirmation == 'S'){
605                         putShip(shipType, line, col, direction, tabuleiro);
606                         break;
607                     }
608                 }else{
609                     printf("\nCoordenada inválida! Tente novamente.\n");
610                 }
611             }while(1);
612 
613             clearScreen();
614             printBattleship();
615             displayGamePlayer(nomeJogador, tabuleiro);
616         }
617         printf("\n\nEmbarcações posicionadas! Pressione S para confirmar as posições ou N para reiniciar: "
    );
618         do{
619             scanf(" %c", &chooseSelection);
620             if(chooseSelection != 'S' && chooseSelection != 'N'){
621                 printf("Opção inválida! Tente novamente.\n");
622             }
623         }while(chooseSelection != 'S' && chooseSelection != 'N');
624     }while(chooseSelection != 'S');
625 }
```

2.3.1.21 putShipsCOM()

```
void putShipsCOM (
            char tabuleiro[][10] )
```

Coloca as embarcações da "máquina" no tabulerio. Aleatoriamente é selecionado um tabuleiro para a máquina.

Parameters

tabuleiro	Tabuleiro da máquina

Definition at line 633 of file utils.c.

```
633                                     {
634     // Como é complicado posicionar aleatoriamente e cheio de regras eu vou fazer uns jogos "fixos" e
        selecionar aleatoriamente um deles
635     srand(time(NULL));
636     int game = rand() % 3;
637     char shipType, line, direction;
638     int col;
639 
640     initMatrix(tabuleiro);
641 
642     switch(game){
643         case 0:
644             // Posicionar o Porta-aviões
645             shipType = 'P';
646             line = 'A', col = 0, direction = 'D';
647             putShip(shipType, line, col, direction, tabuleiro);
648 
649             // Posicionar o Embarcação de Guerra
650             shipType = 'N';
651             line = 'F', col = 1, direction = 'S';
652             putShip(shipType, line, col, direction, tabuleiro);
653 
654             // Posicionar o Cruzador
655             shipType = 'C';
656             line = 'E', col = 6, direction = 'A';
657             putShip(shipType, line, col, direction, tabuleiro);
658 
659             // Posicionar o Submarino
660             shipType = 'S';
661             line = 'H', col = 6, direction = 'W';
662             putShip(shipType, line, col, direction, tabuleiro);
663 
664             // Posicionar o Destruidor
665             shipType = 'D';
666             line = 'D', col = 6, direction = 'D';
667             putShip(shipType, line, col, direction, tabuleiro);
668             break;
669         case 1:
670             // Posicionar o Porta-aviões
671             shipType = 'P';
672             line = 'I', col = 1, direction = 'W';
673             putShip(shipType, line, col, direction, tabuleiro);
674 
675             // Posicionar o Embarcação de Guerra
676             shipType = 'N';
677             line = 'C', col = 7, direction = 'A';
678             putShip(shipType, line, col, direction, tabuleiro);
679 
680             // Posicionar o Cruzador
681             shipType = 'C';
682             line = 'H', col = 5, direction = 'S';
683             putShip(shipType, line, col, direction, tabuleiro);
684 
685             // Posicionar o Submarino
686             shipType = 'S';
687             line = 'A', col = 9, direction = 'S';
688             putShip(shipType, line, col, direction, tabuleiro);
689 
690             // Posicionar o Destruidor
691             shipType = 'D';
692             line = 'C', col = 3, direction = 'S';
693             putShip(shipType, line, col, direction, tabuleiro);
694             break;
695         case 2:
696             // Posicionar o Porta-aviões
697             shipType = 'P';
698             line = 'A', col = 0, direction = 'D';
699             putShip(shipType, line, col, direction, tabuleiro);
700 
701             // Posicionar o Embarcação de Guerra
702             shipType = 'N';
703             line = 'C', col = 1, direction = 'D';
704             putShip(shipType, line, col, direction, tabuleiro);
705 
706             // Posicionar o Cruzador
707             shipType = 'C';
```

```
708             line = 'E', col = 3, direction = 'D';
709             putShip(shipType, line, col, direction, tabuleiro);
710
711             // Posicionar o Submarino
712             shipType = 'S';
713             line = 'G', col = 7, direction = 'W';
714             putShip(shipType, line, col, direction, tabuleiro);
715
716             // Posicionar o Destruidor
717             shipType = 'D';
718             line = 'J', col = 4, direction = 'D';
719             putShip(shipType, line, col, direction, tabuleiro);
720             break;
721     }
722 }
```

2.3.1.22 readCoordenate()

```
void readCoordenate (
             char * line,
             int * col )
```

Realiza a leitura de uma coordenada para o tabuleiro. A leitura continua até que uma coordenada válida seja informada.

Parameters

∗line	Ponteiro de uma variável para ser a linha da matriz
∗col	Ponteiro de uma variável para ser a coluna da matriz

Definition at line 230 of file utils.c.

```
230                                         {
231     int intLine;
232     do{
233         printf("Informe a linha (A-J): ");
234         scanf(" %c", line);
235         printf("Informe a coluna (0-9): ");
236         scanf("%i", col);
237
238          intLine= (int)*line-65;
239         // Verifica se a coordenada é válida
240         if(!checkThreshold(intLine, *col)){
241             printf("\nCoordenada inválida! Tente novamente.\n");
242         }
243     }while(!checkThreshold(intLine, *col));
244 }
```

2.3.1.23 shipName()

```
const char* shipName (
             char ship )
```

Retorna o nome da embarcação de acordo com a sigla

Definition at line 114 of file utils.c.

```
114                                 {
115     switch(ship){
116         case 'P':
117             return "Porta-aviões";
118         case 'N':
119             return "Embarcação de Guerra";
120         case 'C':
121             return "Cruzador";
122         case 'S':
123             return "Submarino";
124         case 'D':
125             return "Destruidor";
126     }
127 }
```

Index