ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

BÁO CÁO BÀI TẬP LỚN
MÔN KIẾN TRÚC MÁY TÍNH

GVHD: PGS.TS Phạm Quốc Cường
Lớp: TN01

Họ và tên: Võ Phương Minh Nhật
MSSV: 2212413

TPHCM, 12-2023

Mục lục

1 Khai báo dữ liệu

Từ dòng 1 đến dòng 46

fileName: tên file text ghi lại cách sắp xếp các con thuyền và các lần đoán của cả 2 người chơi (lưu ý file text này phải cùng thư mục với file code asm).

board1/board2: mảng 200 byte 1 chiều (nhưng được dùng như mảng 2 chiều kích thước 8x25) biểu diễn hình vuông 7x7 (có tọa độ) và các con thuyền của người chơi 1/2.

header: mảng byte lưu chuỗi " 1 2 3 4 5 6 7\n".

input: 2 byte dùng để lưu các input của người chơi khi set up các con thuyền (1 byte cho kí tự input và 1 byte cho kí tự kết thúc chuỗi \0).

guess: 2 byte dùng để lưu các input của người chơi khi đoán các ô (1 byte cho kí tự input và 1 byte cho kí tự kết thúc chuỗi \0).

Còn lại là các biến asciiz lưu các câu dùng để in ra màn hình trong chương trình (mọi tương tác input/output với chương trình đều thông qua hộp thoại chứ không phải terminal).

2 Hàm main

Từ dòng 49 đến dòng 100

Cách hiện thực:

In luật chơi ra màn hình.

In thông báo đã đến phần set up các con thuyền.

Set up mảng board1 bằng cách lưu địa chỉ của board1 vào thanh ghi $s0 và gọi hàm setupBoard.

In "Player 1's turn".

Set up các con thuyền của player 1 bằng cách lưu địa chỉ của board1 vào thanh ghi $s0 và gọi hàm setupShips.

Set up mảng board2 bằng cách lưu địa chỉ của board2 vào thanh ghi $s0 và gọi hàm setupBoard.

In "Player 2's turn".

Set up các con thuyền của player 2 bằng cách lưu địa chỉ của board1 vào thanh ghi $s0 và gọi hàm setupShips.

Gọi hàm writeBoards để ghi bảng 7x7 với các con thuyền của 2 người chơi vào file text.

Thanh ghi $s2 lưu số ô mà player 1 đã đoán trúng. Thanh ghi $s3 lưu số ô mà player 2 đã đoán trúng. Ban đầu lưu 0 vào $s2 và $s3.

In thông báo đã đến phần đoán các ô vuông.

In "Player 1's turn".

Để player 1 đoán ô vuông bằng cách lưu địa chỉ board2 vào $s0, lưu 1 vào $s1 và gọi hàm guessShip.

In "Player 2's turn".

Để player 2 đoán ô vuông bằng cách lưu địa chỉ board1 vào $s0, lưu 2 vào $s1 và gọi hàm guessShip.

Nếu game chưa kết thúc thì nhảy về lại nhãn play để tiếp tục việc đoán các ô vuông của 2 người chơi cho đến khi game kết thúc.

3 Hàm setupBoard

Từ dòng 349 đến dòng 389

Tham số: địa chỉ của board1/board2 lưu ở thanh ghi $s0 (ban đầu board1/board2 chỉ gồm các kí tự khoảng trắng).

Kết quả: set up mảng board1/board2 như hình bên dưới.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0						1			2			3			4			5			6			7	\n
1	A			0			0			0			0			0			0			0			\n
2	B			0			0			0			0			0			0			0			\n
3	C			0			0			0			0			0			0			0			\n
4	D			0			0			0			0			0			0			0			\n
5	E			0			0			0			0			0			0			0			\n
6	F			0			0			0			0			0			0			0			\n
7	G			0			0			0			0			0			0			0			\0

Ý tưởng chung:

Cột 0 gồm tên các hàng (đánh chữ A-G từ trên xuống), hàng 0 gồm tên các cột (đánh số 1-7 từ trái qua phải), các kí tự \n để cách dòng, các kí tự trên ô vuông 7x7 là 0 (do ban đầu chưa có thuyền) và kí tự cuối cùng là kí tự kết thúc chuỗi \0. Hàng đầu tiên và các hàng còn lại bị lệch với nhau để khi in ra trên hộp thoại không bị lệch.

Cách hiện thực:

Ta giả sử arr[8][25] là mảng 2 chiều tương ứng với mảng board[200]. Khi đó $arr[i][j] = board[i * 25 + j]$. Tức là ta dùng board như một mảng 2 chiều dù nó là mảng 1 chiều trong bộ nhớ. Một lưu ý thêm là do đây là mảng kí tự nên ta phải dùng mã ASCII để biểu diễn các kí tự, ví dụ '0' là 48, 'A' là 65, '\n' là 10, '\0' là 0.

Dưới đây là thuật toán của hàm được viết bằng ngôn ngữ C++ để dễ theo dõi.

Thuật toán:

```
for (int i = 1; i < 8; i++){
    board[i * 3 + 2] = i + 48; // cac so 1,2,...,7 o hang tren cung
    board[i * 25] = i + 64;  // cac chu A,B,...,G o cot ngoai cung
    board[i * 25 - 1] = 10;  // cac ki tu \n o cuoi moi hang
}

for (int i = 1; i < 8; i++){
    for (int j = 3; j < 24; j += 3){
        board[i * 25 + j] = 48;  // cac so 0 tren hinh vuong 7x7
    }
}

board[199] = 0;  // ki tu \0 o cuoi
```

4 Hàm setupShips

Từ dòng 391 đến dòng 722

Tham số: địa chỉ của board1/board2 lưu ở thanh ghi $s0.

Ý tưởng chung: Hàm sẽ nhận input từ player để set up các con thuyền. Nếu input không hợp lệ thì sẽ thông báo để người chơi nhập lại. Người chơi có thể dừng game bất cứ lúc nào.

Cách hiện thực:

In ra bảng 7x7 với các con thuyền đã đặt để người chơi quan sát.

Nếu đã đặt 6 thuyền thì kết thúc hàm.

Hỏi người chơi nhập kích thước của con thuyền (2, 3 hoặc 4). Nếu input không hợp lệ hoặc dư thuyền (đã có 3 thuyền 2x1 hoặc 2 thuyền 3x1 hoặc 1 thuyền 4x1) thì yêu cầu nhập lại cho đến khi đúng.

Hỏi người chơi nhập hàng của đầu con thuyền (chữ A-G). Nếu input không hợp lệ thì yêu cầu nhập lại cho đến khi đúng.

Hỏi người chơi nhập cột của đầu con thuyền (số 1-7). Nếu input không hợp lệ thì yêu cầu nhập lại cho đến khi đúng.

Hỏi người chơi nhập hướng của con thuyền (U, D, L hoặc R). Nếu input không hợp lệ thì yêu cầu nhập lại cho đến khi đúng.

Sau khi đã có kích thước, tọa độ hàng cột, hướng của thuyền thì kiểm tra xem thuyền có bị tràn ra ngoài hình vuông 7x7 hay trùng ô với các thuyền đã đặt hay không. Nếu có thì hỏi lại từ đầu (tức là từ việc hỏi kích thước). Nếu không thì ta thêm thuyền vào.

Với việc thêm thuyền vào, ta chia 4 nhãn up, down, left, right tương ứng với hướng của con thuyền để dễ xử lý. Sau khi thêm thuyền xong thì tăng số thuyền tổng và số thuyền với kích thước tương ứng lên 1.

Quay lại đầu hàm để thực thi tiếp cho đến khi kết thúc.

5 Hàm writeBoards

Từ dòng 102 đến dòng 160

Ý tưởng chung: Ghi 2 bảng 7x7 các số 0/1 biểu diễn vị trí các con thuyền mà 2 người chơi đã đặt vào file text.

Cách hiện thực:

Với mỗi bảng, ở hàng đầu tiên (hàng các số 1,2,...,7) thì ta dùng biến header(trong phần Khai báo dữ liệu) để ghi vào file, 7 hàng còn lại thì ta dùng biến board1/board2 tương ứng với player1/player2. Điều này là do hàng 1 và 7 hàng còn lại trong board1/board2 bị lệch nhau (xem phần hàm setupBoard).

Ghi vào file "Player 1's board" và "Player 2's board" tương ứng với bảng 1 và 2.

6 Hàm guessShip

Từ dòng 177 đến dòng 345

Tham số: Nếu là lượt đoán của người chơi 1 thì $s0 lưu địa chỉ của board2, $s1 lưu 1. Còn nếu của người chơi 2 thì $s0 lưu địa chỉ của board1, $s1 lưu 2.

Ý tưởng chung: Hàm sẽ nhận input là tọa độ mà người chơi đoán và thông báo là đoán đúng hay sai. Nếu input không hợp lệ thì thông báo để người chơi nhập lại. Người chơi đầu tiên đoán hết tất cả các ô của đối phương là người thắng. Ngoài ra người chơi có thể dừng game bất cứ lúc nào.

Cách hiện thực:

Hỏi người chơi nhập hàng của ô mà họ đoán (chữ A-G). Nếu input không hợp lệ thì yêu cầu nhập lại cho đến khi đúng.

Hỏi người chơi nhập cột của ô mà họ đoán (số 1-7). Nếu input không hợp lệ thì yêu cầu nhập lại cho đến khi đúng.

Ta ghi ô mà người chơi đoán vào file text và kiểm tra ô mà họ đoán. Nếu ô đó chứa số 0 thì thông báo "MISS!" và kết thúc hàm.

Nếu ô đó chứa số 1 thì thông báo "HIT!", thay số 0 vào ô đó và tăng số ô mà người đó đã bắn trúng thêm 1 ($s2,$s3 lưu số ô đã bắn trúng của player 1, player 2). Nếu người này bắn trúng 16 ô thì thông báo đây là người chiến thắng, ghi kết quả game vào file text và kết thúc game.

7 Tài liệu tham khảo

[1]: https://github.com/pauldevnull/Battleship/blob/master/Battleship.asm