ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KĨ THUẬT MÁY TÍNH

KIẾN TRÚC MÁY TÍNH (CO2007)

BÀI TẬP LỚN

BATTLESHIP

GIẢNG VIÊN: PGS. TS. PHẠM QUỐC CƯỜNG
SINH VIÊN: VÕ HOÀNG HUY - 2211298

TP. HỒ CHÍ MINH, T12 - 2023

Mục lục

1 Giới thiệu

Trong bài tập lớn này yêu cầu sinh viên hiện thực trò chơi Battleship bằng ngôn ngữ MIPS assembly. Vì hạn chế của ngôn ngữ, nên yêu cầu của đề tài là hiện thực lại trong bản đồ 7x7. Yêu cầu đề bài:

- Mỗi tàu trong bản đồ được định dạng bằng tọa độ của điểm đầu tàu và cuối tàu

- Cần để cho mỗi người chơi đặt tàu trước khi chiến đấu. Giá trị của mỗi ô trong bản đồ sẽ là 0 hoặc 1 (1 là ô đó đã có 1 tàu chiếm đóng, còn 0 thì chưa có tàu nào đặt vào). Mỗi người chơi sẽ có 3 thuyền 2x1, 2 thuyền 3x1 và 1 thuyền 4x1. Mỗi thuyền không được xếp chồng lên nhau.

- Sau khi 2 người chơi sắp xếp thuyền thành công, mỗi người sẽ lần lượt chọn ô để tấn công bản đồ của đối phương. Nếu đánh trúng sẽ đưa ra thông báo "HIT!"và đánh dấu ô đó thành giá trị 0. Mỗi người chơi cần phải tự nhớ những ô đã đánh. Trò chơi kết thúc khi bản đồ của một trong hai người chơi chỉ toàn là giá trị 0.

2 Hiện thực

2.1 Tổ chức dữ liệu

- Đầu tiên, ở phần `.data` sẽ khởi tạo sẵn những thông báo về trò chơi hoặc lỗi và các biến dùng trong quá trình hiện thực trò chơi.

- Mỗi người chơi sẽ có một bản đồ để đặt thuyền cũng như trong suốt quá trình chiến đấu. Trong chương trình này, mỗi người chơi sẽ có một mảng 1 chiều kiểu `.byte` dùng để quản lý bản đồ của mình. Do bản đồ được hiển thị như mảng 2 chiều vì thế để quản lý trong mảng 1 chiều cần phải làm phẳng lại mảng 2 chiều. Khi cần truy xuất đến ô có tọa độ (i,j) thì chúng ta có công thức sau đây:

$$\forall i, j (0 \leq i, j \leq 6; i, j \in N) \text{ array2D[i][j]=array1D[7xi+j]}$$

- Ví dụ: Phần tử trong mảng một chiều ứng với ô `[5][6]` sẽ có được truy xuất tại ô : 5x7 + 6 = 41.

2.2 Hiện thực trò chơi

2.2.1 Đặt thuyền

- Mỗi người cần phải đặt hết thuyền của mình lên bản đồ trước khi chiến đấu. Người chơi thứ nhất sẽ đặt thuyền trước sau đó sẽ đến người chơi thứ hai.

- Khi đặt một thuyền vào bản đồ, mỗi người chơi sẽ làm như sau:

Bước 1: Chọn kích thước thuyền cần nhập (2,3 hoặc 4):

+ Chương trình sẽ đưa ra thông báo để người chơi nhập kích thước.

```
Enter the ship size to place (2 3 4)
<Kích thước cần được nhập vào>
```

+ Khi người chơi nhập vào kích thước thuyền không hợp lệ (kích thước thuyền không phải 2,3 hoặc 4).

Ví dụ: Kích thước nhập vào là 1, chương trình sẽ đưa ra thông báo kích thước nhập vào không hợp lệ và yêu cầu người chơi nhập lại.

Size of ship must be 2,3 or 4. Please enter again!

+ Khi người chơi nhập vào kích thước của thuyền muốn đặt nhưng đã hết số lượng thuyền loại đó, thì chương trình sẽ đưa ra thông báo để người chơi chọn lại kích thước thuyền khác.

Ví dụ: Mỗi người chơi sẽ có 3 thuyền kích thước 2x1, ở những lượt chọn trước, người chơi đã chọn hết thuyền có kích thước này nhưng lại không nhớ thì chương trình sẽ đưa ra thông báo.

You have owned full this size of ship. Please choose another size

Bước 2: Người chơi nhập vào tọa độ muốn đặt thuyền vào

+ Ý tưởng: khai báo một chuỗi `InCoor` trên phần `.data` để người chơi nhập vào tọa độ theo đúng định dạng : `<rbow cbow rstern cstern>`. Sau khi có được chuỗi nhập vào, dùng lệnh la lên một thanh ghi để xử lý tính hợp lệ của dữ liệu nhập vào.

+ Sau khi chọn kích thước thuyền thành công, người chơi sẽ được đưa ra thông báo để nhập vào tọa độ muốn đặt thuyền.

Enter your ship coordinates in the following form (rowBow colBow rowStern colStern)

< Tọa độ của thuyền>
Tọa độ của thuyền được nhập vào dưới dạng chuỗi với đúng 7 kí tự
Tọa độ hợp lệ: 0 0 0 1
Tọa độ không hợp lệ: 0 1 0 7, 0 a 0 b, ...

+ Sau khi người chơi nhập tọa độ của thuyền vào, chương trình sẽ kiểm tra tính hợp lệ của tọa độ. Khi không hợp lệ, chương trình sẽ đưa ra thông báo cho người chơi và yêu cầu người nhập lại từ bước 1. Những trường hợp tọa độ nhập vào không hợp lệ và các thông báo của chương trình:

1. Tọa độ nhập vào chứa các kí tự không phải là những số nguyên từ 0 đến 6 hoặc không đủ kích thước yêu cầu.

Ví dụ: `0 a 0 b, 0 1 0 p, 0 1 0 8,...`

Ý tưởng: Dùng một vòng lặp để kiểm tra các kí tự, mỗi kí tự được theo bởi đúng một dấu cách (trừ kí tự cuối) nếu kí tự không phải là số nguyên từ 0 đến 6 hoặc không theo bởi đúng 1 dấu cách.

Your input is not valid. Please enter again!

2. Tọa độ thuyền nhập vào không nằm cùng một hàng hoặc một cột vì thuyền chỉ có thể đặt dọc hoặc ngang.

Ví dụ: `0 1 2 3, ....`

Ý tưởng: Dùng lệnh lb để load những giá trị tại vị trí 0 2 4 6 lên 4 thanh ghi tạm, sau đó kiểm tra: nếu giá trị tại vị trí 0 và 4 bằng nhau thì thuyền được đặt ngang, hoặc giá trị tại vị trí 2 và 6 bằng nhau thì thuyền được đặt dọc. Nếu không thỏa một trong hai trường hợp trên thì thuyền được xéo hoặc một hình thức không hợp lệ.

You can only place the ship in the same row or in the same column. Please enter again!

3. Tọa độ nhập vào không đúng với kích thước của thuyền vừa chọn.
Ví dụ: Người chơi chọn kích thước là 2 nhưng lại nhập là `0 0 0 2,...`

Ý tưởng: Sau khi kiểm tra thuyền được đặt ngang hoặc dọc, sẽ kiểm tra kích thước đã khớp với kích thước đã chọn ở trước. Nếu thuyền được đặt ngang, tiến hành lấy giá trị tại vị trí 6 và 2 trừ đi nhau, nếu âm thì lấy trị tuyệt đối của chúng, sau đó cộng lên 1 và kiểm tra với kích thước ban đầu. Tương tự với trường hợp thuyền đặt dọc.

The length of your ship does not fit with the coordinate. Please enter again!

4. Tọa độ thuyền nhập vào đã được chọn của thuyền khác trước đó.
Ví dụ: `0 0 0 1`, tuy nhiên ô `0 0` đã bị thuyền khác chọn từ trước

Ý tưởng: Để kiểm tra thuyền có bị đặt chồng lên nhau thì trước hết phải xử lý các tọa độ đầu vào. Lần lượt dùng lệnh lb để load những kí tự tại vị trí 0, 2, 4, 6 lên các thanh ghi tạm để lấy tọa độ, sau đó trừ đi 48 (mã ASCII của '0'). Sau đó, dùng một vòng lặp để kiểm tra thuyền chồng lên nhau, với mỗi một lần lặp sẽ lấy lần lượt giá trị i,j trong khoảng từ bow đến stern, dùng công thức đã trình bày ở trên để truy xuất đến ô trong bản đồ, nếu ô đó đã được đánh dấu là 1 thì đã bị trùng còn là 0 thì tiếp tục vòng lặp.

The coordinate has been occupied by another ship. Please enter again!

+ Khi tọa độ nhập vào không hợp lệ, người chơi sẽ quay lại bước 1.

+ Khi tọa độ nhập vào không vướng vào những trường hợp không hợp lệ ở trên. Tiến hành đánh dấu những ô trong khoảng từ bow đến stern bằng một vòng lặp, ý tưởng như phần kiểm tra thuyền nhau nhưng phải đánh dấu những ô đấy thành giá trị 1. Sau đó sẽ hiển thị bản đồ lên cho người chơi.

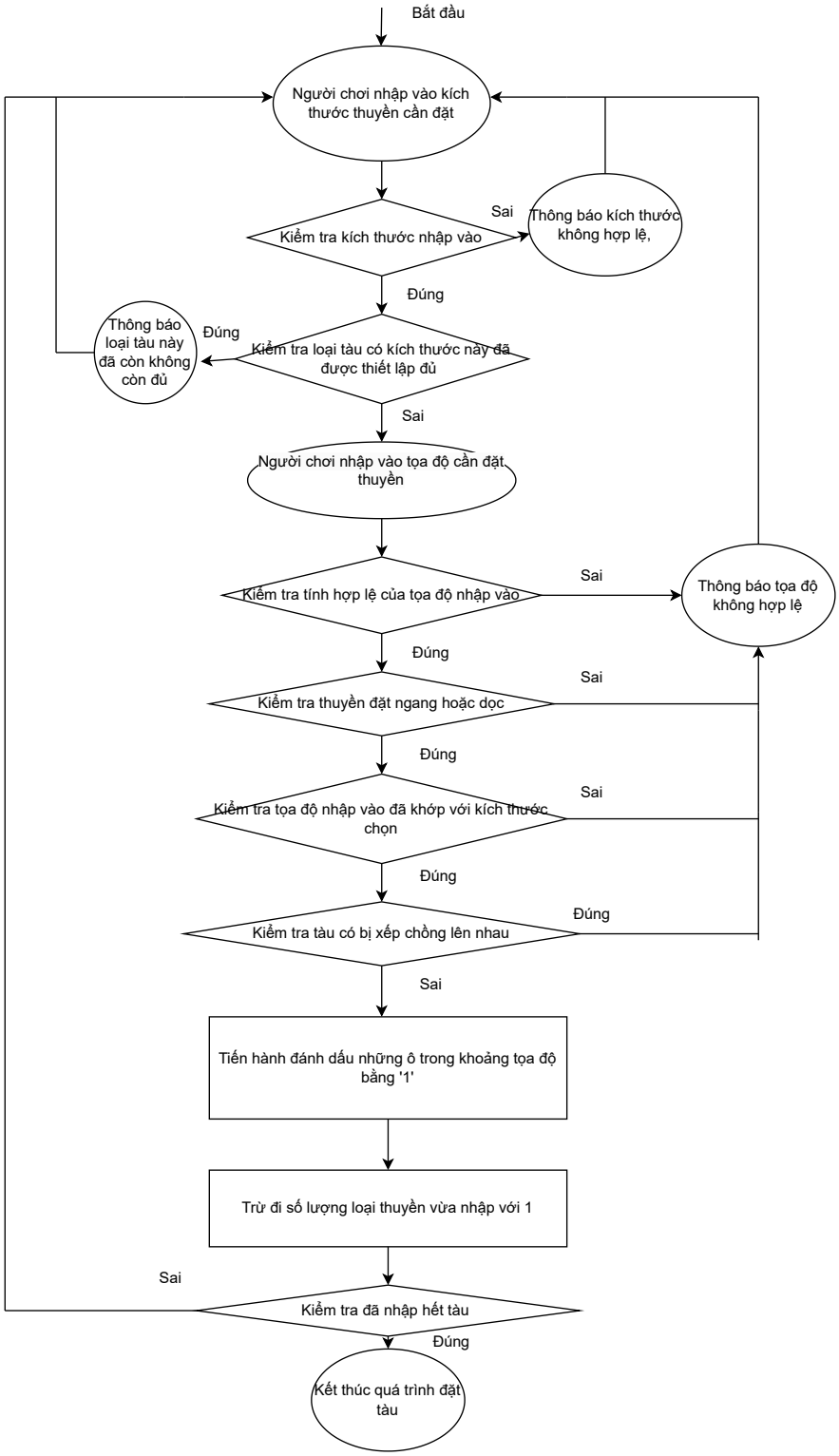
Size = 2, InCoor = 0 0 0 1

0 0 0 0 0 0 0 –>	1 1 0 0 0 0 0
0 0 0 0 0 0 0 –>	0 0 0 0 0 0 0
0 0 0 0 0 0 0 –>	0 0 0 0 0 0 0
0 0 0 0 0 0 0 –>	0 0 0 0 0 0 0
0 0 0 0 0 0 0 –>	0 0 0 0 0 0 0
0 0 0 0 0 0 0 –>	0 0 0 0 0 0 0
0 0 0 0 0 0 0 –>	0 0 0 0 0 0 0

+ Tiếp tục quay lại bước 1, để người chơi hoàn thành đặt các thuyền lên bản đồ. Khi đặt xong tất cả thuyền người chơi sẽ được thông báo đặt thuyền thành công.

Successfully place all the ships

+ Khi cả hai người chơi đặt xong tất cả thuyền của mình thì chương trình sẽ thông báo đến phần chiến đấu.



Hình 1: *Lưu đồ giai đoạn đặt thuyền*

2.2.2 Chiến đấu

- Khi bước vào giai đoạn chiến đấu, người chơi thứ nhất sẽ tấn công trước sau đó hai người chơi thay phiên nhau tấn công. Mỗi người chơi sẽ lần lượt chọn ô cần đánh. Trong giai đoạn chiến đấu, mỗi người chơi sẽ có cho mình một bản đồ chiến đấu với khởi tạo ban đầu chỉ toàn là giá trị 0, nếu đánh trúng trên bản đồ đặt thuyền của đối phương thì sẽ được đánh dấu là 1 còn đối phương sẽ bị đánh dấu trong bản đồ đặt thuyền là 0. Bản đồ đặt thuyền sẽ được giấu đi và không được xem bởi đối thủ

- Mỗi người chơi sẽ sở hữu 6 thuyền với tổng cộng 16 (4x1 + 2x3 + 3x2) ô có giá trị là 1 trên bản đồ đặt thuyền của mình.

- Ô người chơi muốn tấn công cần được nhập vào dưới dạng chuỗi có đúng 3 kí tự, được lưu trữ trong biến `targetcell` trong phần `.data` và theo đúng định dạng sau `<rtarget ctarget>`. Những trường hợp chuỗi nhập vào không hợp lệ thì chương trình sẽ đưa ra thông báo và yêu cầu người chơi nhập lại.

- Dùng lệnh la để load giá trị của chuỗi nhập vào `targetcell` lên một thanh ghi. Sau đó load lần lượt những giá trị tại vị trí 0, 1, 2 của chuỗi bằng lệnh lb lên 3 thanh ghi tạm. Kiểm tra tính hợp lệ của chuỗi nhập vào bằng cách: giá trị tại vị trí 0 và 2 là số nguyên có giá trị từ 0 đến 6 và kí tự tại vị trí 1 phải là dấu cách. Nếu không thỏa điều kiện trên thì đưa ra thông báo cho người chơi nhập lại.

> The range of coordinate's target cell is from 0 to 6. Please enter again!

- Nếu ô tấn công đã hợp lệ thì tiến hành kiểm tra xem người chơi có đánh trúng hoặc không. Ý tưởng: lấy các giá trị tại vị trí 0 và 2, dùng công thức đã nói ở trên để truy xuất đến ô trong bản đồ đặt thuyền của mỗi người chơi, nếu ô đấy bằng 1 tức là đánh trúng và đánh dấu lại bằng 0, còn ngược lại đã đánh hụt.

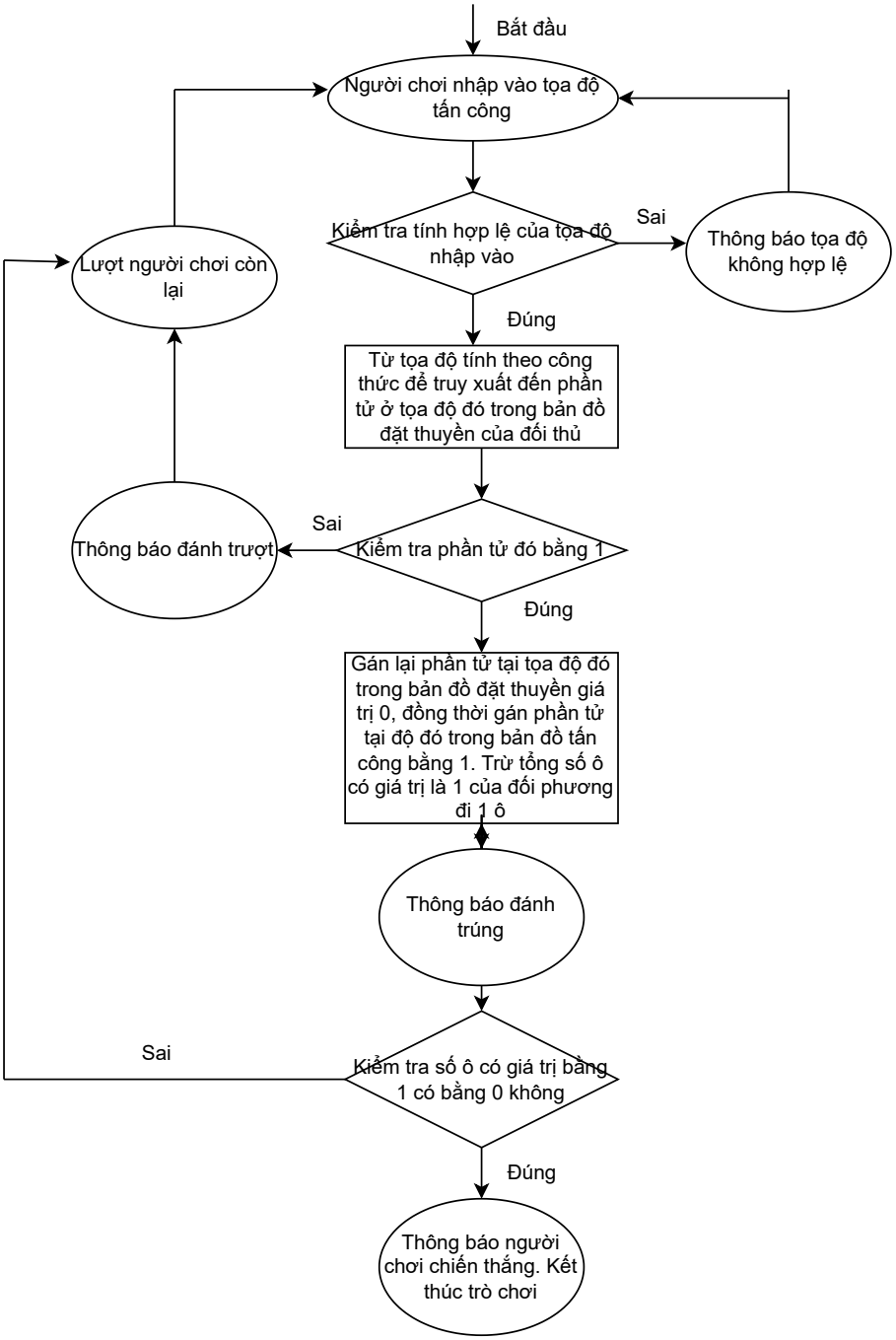
> Người thứ nhất tấn công Bản đồ đặt thuyền của người thứ hai (Hidden)
> Targetcell : 0 2
> 0 0 0 0 0 0 0 1 1 1 1 1 1 0
> 0 0 0 0 0 0 0 1 1 1 1 1 1 0
> 0 0 0 0 0 0 0 1 1 1 1 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> Thông báo trò chơi
> HIT!
> 0 0 1 0 0 0 0 1 1 0 1 1 1 0
> 0 0 0 0 0 0 0 1 1 1 1 1 1 0
> 0 0 0 0 0 0 0 1 1 1 1 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> 0 0 0 0 0 0 0 0 0 0 0 0 0 0

- Sau mỗi lần một người tấn công, sẽ kiểm tra xem đã có người chơi nào chiến thắng hay

chưa.

Ý tưởng: mỗi người ban đầu sẽ có 16 ô bằng 1, khi bị đối phương đánh trúng sẽ trừ đi 1, còn nếu đánh hụt sẽ giữ nguyên. Nếu một trong hai người chơi có lượng ô số 1 trong bản đồ đặt thuyền bằng 0 thì người chơi đó sẽ thua.

```
Người thứ nhất tấn công                Bản đồ đặt thuyền của người thứ hai (Hidden)
Targetcell : 0 2
1 1 0 1 1 1 0                          0 0 1 0 0 0 0
1 1 1 1 1 1 0                          0 0 0 0 0 0 0
1 1 1 1 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
Thông báo trò chơi
HIT!
1 1 1 1 1 1 0                          0 0 0 0 0 0 0
1 1 1 1 1 1 0                          0 0 0 0 0 0 0
1 1 1 1 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
0 0 0 0 0 0 0                          0 0 0 0 0 0 0
Congrats!! Player 1 win!!!
```



Hình 2: *Lưu đồ giai đoạn chiến đấu*

2.3 Kết thúc trò chơi

- Khi xác định được một trong hai người chơi chiến thắng, người chơi có thể xem lại tọa độ đặt thuyền và quá trình tấn công của cả hai ở 2 file : `p1.txt` và `p2 .txt`.

```
p1.txt
Set up
Size: 2
Coordinate: 0 0 0 1
Size: 2
Coordinate: 0 2 0 3
Size: 2
Coordinate: 0 4 0 5
Size: 3
Coordinate: 1 0 1 2
Size: 3
Coordinate: 1 3 1 5
Size: 4
Coordinate: 2 0 2 3
Attack: 0 0
Attack: 0 1
Attack: 0 2
Attack: 0 3
Attack: 0 4
Attack: 1 0
Attack: 1 1
Attack: 1 2
Attack: 1 3
Attack: 1 4
Attack: 1 5
Attack: 0 5
Attack: 2 0
Attack: 2 1
Attack: 2 2
Attack: 2 3
```

Tài liệu tham khảo

[1] David A. Patterson & John L. Hennessy, *Computer Organization and Design: the Hardware/Software Interface.*

[2] MARS - Mips Assembly and Runtime Simulator, SYSCALL system services. Truy cập tại: https://courses.missouristate.edu/kenvollmar/mars/help/syscallhelp.html