Đại học Khoa Học Tự Nhiên

Môn học: Kỹ thuật lập trình

Báo cáo đồ án Matching Game

Nguyễn Thuận Phát - 21127665
Đinh Thiện Minh - 21127353

Giảng viên:
Bùi Huy Thông

1 Thông tin

Báo cáo đồ án lập trình Matching Game
Thông tin nhóm:
Nguyễn Thuận Phát - 21127665
Đinh Thiện Minh - 21127353

Mục lục

2 Tutorial

2.1 Game State

Người chơi khi vào menu game có thể chọn 4 hành động chính là easy mode, difficult mode , bảng xếp hạng và thoát game .Hình 1. Để chọn các chức năng thì người chơi sẽ dùng các phím WASD để tùy chọn lên xuống các chế độ

Hướng dẫn cách chơi: Để chơi thì người chơi sẽ sử dụng các phím WASD để di chuyển.

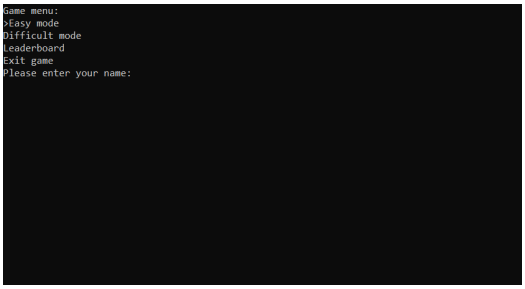
- Để khóa chọn 1 ô thì người chơi sẽ cần nhấn phím Enter và chọn thêm 1 ô để tạo thành 1 cặp hoàn chỉnh. Nếu 2 ô tạo thành 1 cặp thì 2 ô ấy sẽ bị xóa đi trong chế độ easy.Trong chế độ difficult thì các ô gần đó sẽ chạy thế chỗ chúng
- Để hoàn thành trò chơi thì người chơi phải nối hết các cặp khả dụng. Nếu không còn cặp khả dụng nào thì người chơi đã thắng. Ngược lại người chơi sẽ thua cuộc. Game over.

2.2 Game Utility

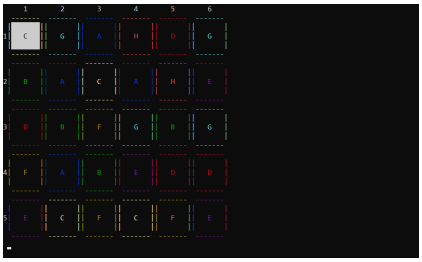
Có 2 chế độ chính: easy và dificult mode

Hình. 1: Menu Game

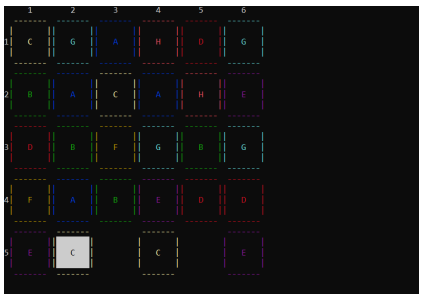
Di chuyển con trỏ để lựa chọn chế độ.Hình 2.



Hình. 2: Chọn chế độ



(a) Giao diện



(b) Khi chọn 2 ô

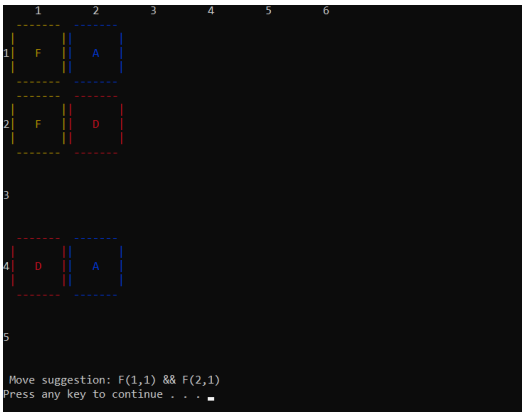
Hình. 3: Easy Mode

Giao diện chế độ easy mode (3a). Hiệu ứng khi chọn xong 1 cặp (3b)

Leader Board. Sau khi chọn mode thì trò chơi sẽ yêu cầu bạn nhập tên để lưu lại điểm số dựa trên thời gian hoàn thành màn chơi.Hình 4

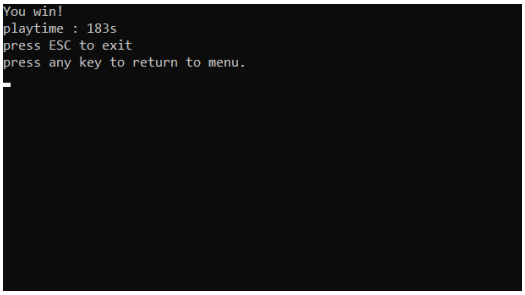
Hình. 4: Leader Board

Suggestion features. Nhấn phím h để nhận gợi ý 1 cặp bao gồm chữ và tọa độ vị trí . Ví dụ: F(1,1); F(2,1) . Hình 5

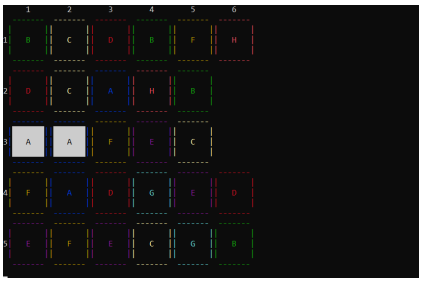


Hình. 5: Suggestion Feature

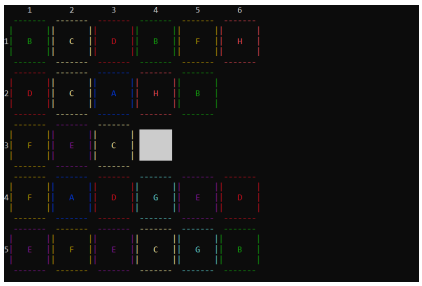
Difficult Mode. Sau khi chọn thành công 1 cặp thì các cặp kế bên sẽ trượt vào chỗ của chúng.Hình 7a và hình 7b. Khi hoàn thành màn hình sẽ hiện tin nhắn và thời gian chơi hình 6



Hình. 6: Winning Scene



(a) Chọn 1 cặp



(b) Sau khi chọn

Hình. 7: Difficult Mode

3 Functional

3.1 File Structural

+Board.cpp: gồm những hàm liên quan đến biểu diễn bảng(tạo giá trị cho bảng, in bảng ra màn hình, đổi màu, in menu...)

+endGameCheck.cpp: gồm những hàm kiểm tra xem trò chơi đã kết thúc chưa(còn ô hay không, còn cặp nối được hay không)

main.cpp: file chính chứa các thuật toán vận hành game

matching.cpp: file cpp chứa các hàm boolean kiểm tra đường nối 2 ô có hợp lệ hay không, hàm di chuyển , hàm reset giá trị các biến

advance.cpp: file cpp chứa các hàm để làm thực hiện các yêu cầu nâng cao

boardLinkedList.cpp: file cpp chứa các hàm liên quan tới bảng nhưng làm bằng linked list

Các file .h là header của các file cpp cùng tên

3.2 Function

Hàm Board.cpp
-hàm repBoard:
+chức năng: in bảng các ô ra console, biểu diễn visual effect, mỗi ô có màu riêng
+cách hoạt động: chạy vòng lặp for để duyệt từng dòng của mảng char 2 chiều, trong vòng lặp lại có thêm các vòng lặp for khác duyệt từng cột của dòng để in ra từng phần của ô, nếu phần tử có giá trị là đấu cách ‘ ‘ thì không in ra gì cả(quy ước phần tử nào của mảng có giá trị là ‘ ‘ ngĩa là ô trống). Vòng lặp for lớn chạy được 1 lần thì in ra được 1 dòng của bảng. Ô nào đang được người chơi chọn sẽ đổi chữ đen và nền trắng
-hàm genBoard:
+chức năng: tạo giá trị ngẫu nhiên cho mỗi phần tử của mảng char 2 chiều nhập vào

+cách hoạt đông: sẽ có 1 mảng a gồm 7 phần tử từ a[0] đến a[6] mỗi phần tử lưu số lần xuất hiện của 1 chữ cái.Có 1 biến có giá trị ngẫu nhiên rd từ 0 đến 7 bởi hàm rand().Chạy vòng lặp để duyệt từng phần tử của mảng 2 chiều.Kiểm tra xem a[rd] có khác 0 hay không, nếu bằng không tức là chữ cái đó đã xuất hiện đủ số lần quy định , phải chạy vòng while để random lại giá trị cho rd đến khi nào a[rd] != 0. Nếu a[rd] đã khác 0 thì giá trị của phần tử đó của mảng sẽ bằng 'A' + rd (như vậy mảng sẽ bao gồm các chữ cái từ a đến h) rồi sao đó giảm a[rd] đi 1 đơn vị rồi tiếp tục random lại rd cho ô tiếp theo.

-hàm delete cell:

-chức năng: xóa đi giá 2 ô nhập vào(nếu match 2 ô đó thành công)

+cách hoạt động: gán giá trị 2 ô nhập vào là dấu cách ' ' (hàm repBoard không in ra những ô có giá trị là dấu cách)

-hàm repColor:

+chức năng: đổi màu console tùy theo giá trị của ô nhập vào

+cách hoạt động: sử dụng switch case đối với ô nhập vào(ví dụ nếu là A thì sẽ đổi màu console thành xanh)

matching.cpp

-hàm move:

+chức năng: di chuyển ô con trỏ tùy vào nút mà người dùng nhập là a, s, w hay d

+cách hoạt động: nhập vào hàm là sx,sy tương đương hàng và cột của ô con trỏ, switch case nút người dùng nhập, rồi tùy vào đó mà cộng hay trừ sx và sy đi 1 đơn vị(ví dụ người dùng nhấn nút a tức là sang trái thì trừ sy đi 1)

-hàm checkLine:

+chức năng: kiểm tra 1 hàng có phải là hàng trống

+cách hoạt động:các giá trị nhập vào là start(vị trí bắt đầu), end(vị trí kết thúc) và row(hàng cần kiểm tra) chay vòng lặp for cho i từ giá trị start đến giá trị end nếu a[row][i] != ' ' thì trả về false (tức là đã đụng ô không phải ô trống) còn nếu chạy hết vòng lặp mà không gặp chướng ngại thì trả về true

-hàm checkColumn:

+chức năng: kiểm tra một cột xem có phải cột trống

+cách hoạt động: tương tự hàm checkLine chỉ đổi chỗ biến i

-hàm checkI:

+chức năng: kiểm tra xem 2 ô nhập vào có nối theo đường thẳng được không

+cách hoạt động: dùng if so sánh nếu 2 ô trùng nhau thì trả về false. xác định y của ô nào nhỏ hơn và y của ô nào lớn hơn để chạy hàm checkLine

kiểm tra đường nằm ngang nối 2 ô. Đường thẳng nằm dọc làm tương tự nhưng với checkColumn

-hàm checkL:

+chức năng:kiểm tra xem 2 ô có nối hình chữ L được không

+cách hoạt động: kiểm tra xem 2 ô có cùn hàng hay cùng cột hay không nếu có thì trả về false vì như vậy không nối kiểu L được. Xác định 2 ô phân bố theo kiểu nào(ô này nằm bên trái hay bên phải, phía trên hay phía dưới ô kia, có 2 kiểu: 1 ô bên trái phía trên và 1 ô bên phải phía dưới hoặc bên phải phía trên và bên trái phía dưới).Xác định ô nằm phía trên bên trái LU(LeftUp) và phía dưới bên phải RD(RightDown) hoặc ô nằm phía trên bên phải RU(RightUp) và phía dưới bên trái LD(LeftDown) sau khi đã kiểm tra vị trí 2 ô. Sau đó kiểm tra 2 đường nối hình chữ L bằng cách dùng checkLine và checkColumn để kiểm tra các hàng ,cột tạo thành hình chữ L theo các khuôn mẫu tùy theo vị trí 2 ô. Nếu đã kiểm tra hết 2 đường nối hình chữ L mà không thỏa thì trả về false

-hàm checkZ:

+chức năng: kiểm tra xem 2 ô có nối kiểu z được không

+cách hoạt động: đầu tiên, xác định vị trí 2 ô tương tự như hàm checkL

- Kiểm tra đường nối hình chữ Z: chạy vòng lặp for cho i chạy từ giá trị cột của ô nằm bên trái cộng thêm 1 tới giá trị cột của ô nằm bên phải trừ 1 để kiểm tra các cột nằm giữa 2 ô có trống hay không bằng hàm checkColumn, với start là giá trị dòng của ô nằm trên và end là giá trị dòng của ô nằm dưới, column là i. Nếu có bất cứ cột i nào là cột trống thì chạy 2 hàm checkLine

- kiểm tra đường nằm ngang từ 2 ô đến ô cùng hàng trên cột i có trống hay không, nếu có thì nối được kiểu Z trả về true. Chạy hết vòng lặp trả về false. Kiểm tra đường nối hình chữ N: tương tự chữ Z nhưng đổi vị trí hàm checkLine và checkColumn. Đầu tiên chạy hàm checkLIne kiểm tra các hàng giữa 2 ô, nếu có hàng nào trống thì chạy 2 hàm checkColumn tiếp tục kiểm tra 2 cột nối từ ô cùng cột trên hàng đó đến 2 ô.

-hàm checkU:

+chức năng: kiểm tra xem 2 ô có nối hình chữ U được hay không

+cách hoạt động: xác định 2 ô có cùng hàng , cùng cột hay khác hàng khác cột. Nếu cùng hàng thì chỉ kiểm tra hình chữ U và chữ U ngược. Nếu cùng

cột thì chỉ kiểm tra chữ U ngã về trái và chữ U ngã về phải. Nếu khác hàng khác cột thì kiểm tra tất cả trường hợp chữ U. Mảng 2 chiều lúc tạo thì có đến 7 hàng 8 cột nhưng chỉ có 5 hàng 6 cột bên trong là có nội dung còn viền bên ngoài đều là dấu cách ' ' chính là để thuận tiện cho hàm checkU.

- Chữ U: chạy vòng lặp for cho i chạy từ chỉ số hàng của ô có chỉ số hàng lớn hơn(nằm dưới) đến hàng cuối cùng của mảng, dùng hàm checkLine để duyệt qua từng hàng, hàng nào trống thì tiếp tục chạy 2 hàm checkColumn kiểm tra đường nối từ 2 ô đến ô cùng cột trên hàng đó.

- Chữ U ngược: tương tự chữ U nhưng kiểm tra các hàng từ hàng đầu của mảng đến hàng của ô có chỉ số hàng thấp hơn(nằm trên), hàng nào trống thì tiếp tục chạy 2 hàm checkColumn kiểm tra đường nối từ 2 ô đến ô cùng cột trên hàng đó.

- Chữ U ngã về trái: chạy vòng lặp for cho i chạy từ chỉ số cột của ô có chỉ số cột lớn hơn(nằm bên phải) đến cột cuối cùng của mảng, dùng hàm checkColmun để duyệt qua từng cột, cột nào trống thì tiếp tục chạy 2 hàm checkLine kiểm tra đường nối từ 2 ô đến ô cùng hàng trên cột đó.

- Chữ U ngã về phải:tương tự chữ U ngã về trái nhưng kiểm tra các cột từ cột đầu của mảng đến cột của ô có chỉ số cột thấp hơn(nằm bên trái), cột nào trống thì tiếp tục chạy 2 hàm checkLine kiểm tra đường nối từ 2 ô đến ô cùng hàng trên cột đó.

endGameCheck.cpp

-hàm checkPairsleft:

+chức năng: kiểm tra xem bảng còn ô nào không

+cách hoạt động: dòng vòng for duyệt từ đầu đến cuối mảng. Nếu có ô nào có giá trị khác dấu cách ' ' thì tức là còn ô, trả về true, ngược lại chạy hết vòng lặp thì tức là các phầm tử mảng đều là dấu cách hết trả về false.

-hàm checkValidPairs:

+chức năng: kiểm tra xem còn ô nào trong bảng còn có thể nối với nhau được không

+cách hoạt động: chạy vòng lặp for duyệt từng phần tử trong mảng. Trong vòng lặp đó thì lại chạy thêm 1 vòng lặp for để duyệt từ phần tử kế phần tử

đó đến phân tử cuối cùng mà cùng hàng phần tử đó dùng if để kiểm tra xem 2 phần tử có nối với nhau bằng bất cứ kiểu nào được hay không, nếu có thì trả về true. Sau đó tiếp tục dùng vòng lặp for để chạy từng phần tử trong hàng kế tiếp đến hàng cuối cùng rồi lại dùng if để kiểm tra xem 2 phần tử có nối với nhau bằng bất cứ kiểu nào được hay không, nếu có thì trả về true. Chạy hết vòng lặp lớn mà vẫn không có phần tử nào match được thì trả về false

advance.cpp

-hàm moveSuggestion:

+chức năng: in ra console chỉ số hàng, cột của 2 ô có thể nối được với nhau
+cách hoạt động: như hàm checkValidPairs nhưng thay vì trả về true/false thì in chỉ số cột, dòng của 2 ô ra, nếu không có ô nào nối được thì không in gì hết.

-hàm sortFile:

+chức năng: đọc và sắp xếp thứ tự của dữ liệu trong file lưu trữ thành tích người chơi(dùng cho leaderboard) theo thứ tự tăng dần theo thời gian chơi
+cách hoạt động: khai báo mảng kiểu phayer (kiểu struct khai báo từ trước, gồm có tên người chơi và thời gian chơi), dùng vòng lặp while để đọc dữ liệu từ file, có biến count để biết số lượng phần tử. Sort file bằng interchange sort theo thời gian chơi. In lại dữ liệu vào file

- Nếu số lượng phần tử nhỏ hơn 5 hoặc bằng thì in lại hết
- Nếu số lượng phần tử lớn hơn 5 thì chỉ in lại 5 phần tử có thời gian chơi ngắn nhất

-hàm deleteBlank:

+chức năng: dồn ô về phía bên trái trong difficult mode
+cách hoạt động: chạy vòng lặp for duyệt từng phần tử. Nếu phần tử đó là khoảng trống thì chạy vòng for dồn ô sang trái (gán giá trị của phần tử này bằng giá trị của phần tử sau nó), sau đó gán giá trị của ô sau nó bằng khoảng trống. Dùng if để kiểm tra xem giá trị của ô có phải khoảng trống không, nếu có thì chạy thêm lần nữa(trường hợp 2 ô trống liền nhau). chỉ chạy vòng lặp dồn hàng trong hàm này tối đa 2 lần vì trong 1 thời điểm nếu có 1 cặp nối được với nhau thành công thì sẽ chỉ có 2 ô trống.

-Hàm printLB:

+chức năng: in leaderboard ra console
+cách hoạt động: khai báo 1 chuỗi 5 phần tử (chỉ in ra 5 người có thành

tích tốt nhất sau khi đã chạy hàm sortFile) có kiểu dữ liệu là player để lưu dữ liệu từ file rồi in ra console

3.3 Data Structures

-hàm setcolor:
+chức năng: đổi màu chữ và nền
+nhập vào theo format 0xab ,trong đó a là màu nền và b là màu chữ, cần thư viện window.h

4 Functions using Linked List

boardLinkedList.cpp:
-hàm getCell:
+chức năng: tạo ô mới với dữ liệu nhập vào
-hàm genBoardLinkedList:
+chức năng: tạo bảng theo kiểu linked list
+cách hoạt động: cũng có mảng a lưu tần suất xuất hiện của giá trị và biến rd như bên pointer . Đầu tiên cấp phát động để tạo 1 list board rỗng, sau đó tạo giá trị ngẫu nhiên cho biến rd bằng hàm rand(); rồi tạo phần tử đầu tiên của list với hàm getCell với giá trị nhập vào là 'A' + rd. Tạo 1 biến cell *p để phục vụ việc tạo list, gán p bằng board.head. Chạy vòng for cho i chạy từ 1 đến phần tử thứ boardSize - 1 (boardSize là dữ liệu người dùng nhập vào hàm, là số lượng ô của bảng, chỉ chạy tới boardSize - 1 vì trong list đã có sẵn 1 phần tử rồi). Kiểm tra a[rd] có bằng 0 hay không , nếu có thì random lại tới khi nào a[rd] khác 0, cho p->next = getCell('A' +rd); rồi gán p = p->next (để tiếp tục nối ô kế tiếp), trừ giá trị a[rd] đi 1 rồi tiếp tục random lại biến rd

5 References

Đổi màu chữ trên console Color change.
Link https://daynhauhoc.com/t/huong-dan-nhung-cach-don-gian-de-doi-mau-chu-tren-console-windows-trong-ngon-ngu-c-cho-newbie/106319?fbclid=IwAR0tMeEbvc-_rOHA6AT3YGhtAR1Glea2TrrdLJJ737XUlEt

arYN1y6kd2ho
Tính thời gian để lưu leader board Playtime
Link https://duongdinh24.com/tinh-thoi-gian-chay-chuong-trinh-trong-c-c/?fbclid=IwAR0tMeEbvc-_rOHA6AT3YGhtAR1Glea2TrrdLJJ737XUlEtarYN1y6kd2ho