BỘ TÀI CHÍNH

TRƯỜNG ĐẠI HỌC TÀI CHÍNH – MARKETING

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN MÔN HỌC
LẬP TRÌNH C#.NET CƠ BẢN

Tên đề tài:

LẬP TRÌNH GAME MINESWEEPER

Giảng viên hướng dẫn: **ThS. Nguyễn Thanh Trường**

Mã lớp HP: **2421101169301**

Nhóm sinh viên thực hiện:

Nguyễn Trần Đoan Thi – MSSV: 2221004306

Nguyễn Thị Trà Giang – MSSV: 2221004162

TP. HCM, THÁNG 06 NĂM 2024

BỘ TÀI CHÍNH

TRƯỜNG ĐẠI HỌC TÀI CHÍNH – MARKETING

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN MÔN HỌC
LẬP TRÌNH C#.NET CƠ BẢN

Tên đề tài:

LẬP TRÌNH GAME MINESWEEPER

Giảng viên hướng dẫn: **ThS. Nguyễn Thanh Trường**

Mã lớp HP: **2421101169301**

Nhóm sinh viên thực hiện:

Nguyễn Trần Đoan Thi – MSSV: 2221004306

Nguyễn Thị Trà Giang – MSSV: 2221004162

TP. HCM, THÁNG 06 NĂM 2024

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin gửi lời cảm ơn đến toàn thể quý thầy cô khoa Công nghệ thông tin trường Đại học Tài chính – Marketing đã dành nhiều thời gian và công sức của mình để giảng dạy và truyền đạt những kiến thức hữu ích cho chúng em. Thầy cô đã tạo cho chúng em những điều kiện thuận lợi nhất để chúng em có thể hoàn thành đồ án này. Đặc biệt, chúng em xin dành lời cảm ơn chân thành đến thầy Nguyễn Thanh Trường – người thầy dạy dỗ tận tình và luôn giải đáp các thắc mắc, ý kiến, giúp đỡ chúng em trong quá trình thực hiện đồ án kết thúc môn học.

Mặc dù đã có nhiều cố gắng nhưng do hạn chế về thời gian thực hiện, kinh nghiệm nên không tránh khỏi những sai sót trong bài đồ án. Chúng em xin chân thành cảm ơn những ý kiến đóng góp và sự hỗ trợ nhiệt tình từ giảng viên hướng dẫn đã giúp chúng em ngày một hoàn thiện bản báo cáo này. Chúng em rất mong có thể nhận được những ý kiến đóng góp, nhận xét để có thể bổ sung, hoàn thiện kiến thức bản thân.

Cuối cùng, chúng em kính chúc **thầy Nguyễn Thanh Trường** – giảng viên khoa Công Nghệ Thông Tin trường ĐH Tài chính – Marketing lời chúc sức khỏe và luôn thành công trong sự nghiệp của mình!

NHẬN XÉT VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN 1

. .....

. .....

. .....

. .....

. .....

. .....

. .....

..... .

. .....

.....

.....

. .....

. .....

. .....

. .....

Điểm số:.....

Điểm chữ: .....

Tp.Hồ Chí Minh, ngày..... tháng.... năm 2024

Giảng viên

NHẬN XÉT VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN 2

. .....

. .....

. .....

. .....

. .....

. .....

. .....

..... .

. .....

.....

.....

. .....

. .....

. .....

. .....

Điểm số:.....

Điểm chữ: .....

Tp.Hồ Chí Minh, ngày..... tháng.... năm 2024

Giảng viên

DANH MỤC TỪ VIẾT TẮT

DANH MỤC THUẬT NGỮ ANH – VIỆT

DANH MỤC CÁC HÌNH ẢNH

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1. Lý do hình thành đề tài

Trong thời đại công nghệ 4.0 ngày nay, đặc biệt với sự xuất hiện của ChatGPT - một AI có thể giải đáp mọi câu hỏi của con người, chúng em, những sinh viên ngành công nghệ thông tin, càng nhận thức rõ ràng rằng bản thân cần phải trau dồi thêm nhiều kiến thức để có thể tạo ra những sản phẩm tuyệt vời hơn nữa.

Trước khi học môn Lập trình C#.NET, chúng em đã học qua các môn như Cơ sở lập trình và Cấu trúc dữ liệu và giải thuật bằng ngôn ngữ C++. Với sự giảng dạy tận tình của các giảng viên và sự thú vị của lập trình, chúng em luôn mong muốn vận dụng những kiến thức đã học để tạo ra sản phẩm của riêng mình. Khi bắt đầu môn học Lập trình C#.NET và được thầy định hướng về đồ án cuối kì, trong đó có thể lập trình game, chúng em đã quyết định chọn làm game để ứng dụng những kiến thức lập trình đã có.

Sau khi thảo luận về các trò chơi có thể làm và tính ứng dụng của nhiều kiến thức về ngôn ngữ C#, chúng em đã quyết định chọn đề tài "**Lập trình Game Minesweeper**" cho đồ án của mình.

1.2. Giới thiệu về Game Minesweeper

Dò mìn hay gỡ bom (tiếng Anh: *Minesweeper*) là một trò chơi giải đố trên máy tính dành cho một người chơi. Trò chơi bao gồm một "bãi mìn" là những ô vuông có thể chứa "mìn", và người chơi cần phải dựa vào những con số thể hiện số mìn xung quanh để mở hết tất cả những ô vuông trống mà không kích nổ quả mìn nào. Trò chơi được xây dựng như một chương trình giải trí cài đặt trên hệ điều hành Microsoft Windows. Ngoài Windows, trò chơi còn có trong Linux (như Kmines (giao diện KDE) và gnomine (giao diện GROME). Nhờ quy luật đơn giản, nhiều biến thể của trò chơi cũng được phát hành như *Dò mìn 3D, Minesweeper X, và Crossmines*.

Minesweeper là một trò chơi đơn giản nhưng đầy thử thách, có lịch sử lâu đời và sự phát triển đáng kể qua các phiên bản. Trò chơi lần đầu tiên xuất hiện dưới tên gọi "Cube" vào đầu những năm 1960. Tuy nhiên, Minesweeper chỉ thực sự trở nên phổ biến khi Microsoft phát hành trò chơi này trong bộ phần mềm Microsoft Entertainment Pack năm 1990. Kể từ đó, Minesweeper đã trở thành một phần không thể thiếu trong các

phiên bản hệ điều hành Windows. Trò chơi không chỉ là một công cụ giải trí mà còn giúp người chơi rèn luyện khả năng tư duy logic và kỹ năng giải quyết vấn đề.

1.3. Mục tiêu và nội dung nghiên cứu

1.3.1. Mục tiêu nghiên cứu

- Nắm chắc được các phần kiến thức lập trình liên quan đến lập trình trò chơi Minesweeper (Dò mìn).
- Nắm chắc các kiến thức về lập trình WinForms trong C#.NET.
- Nắm bắt được quy trình làm game hoàn chỉnh.
- Biết được cách phân tích giải thuật hợp lý cho code.

1.3.2. Nội dung nghiên cứu

- Tìm hiểu về ngôn ngữ lập trình C#.NET, WinForms và những thuộc tính cần dùng trong WinForms để giúp nhóm hoàn thành đề tài.
- Các sơ đồ đặc tả phù hợp để diễn giải luồng chạy của chương trình.

1.4. Đối tượng và phạm vi đề tài

1.4.1. Đối tượng

Minesweeper là một trò chơi phù hợp cho nhiều lứa tuổi và đối tượng khác nhau, tùy thuộc vào sở thích và khả năng của từng người.

- Người lớn: Nhiều người lớn chơi Minesweeper để giải trí, rèn luyện khả năng tư duy logic và sự kiên nhẫn.
- Trẻ em: Trẻ em cũng có thể chơi Minesweeper, đặc biệt là khi trò chơi được điều chỉnh độ khó để phù hợp với khả năng của chúng. Trò chơi có thể giúp trẻ phát triển khả năng giải quyết vấn đề và tư duy chiến lược.
- Những người yêu thích trò chơi trí tuệ: Minesweeper là một trò chơi đòi hỏi sự tư duy và chiến lược, do đó rất hấp dẫn đối với những ai thích thử thách trí tuệ.

- Người chơi mới làm quen với máy tính: Minesweeper là một trò chơi đơn giản, dễ hiểu, do đó là lựa chọn tốt cho những người mới bắt đầu làm quen với máy tính.

1.4.2. Phạm vi

Trò chơi dành cho 1 người chơi với một bảng chơi là một lưới ô vuông, có thể có nhiều kích thước khác nhau, từ lưới nhỏ 9 x 9 cho đến lưới lớn 30 x 16. Minesweeper chủ yếu giúp rèn luyện kỹ năng tư duy logic, phản xạ và kiên nhẫn của người chơi.

1.5. Phương pháp nghiên cứu đề tài

- Phương pháp nghiên cứu tài liệu: Xây dựng hệ thống dựa trên những tài liệu có sẵn như thông tin về lập trình game, cách thức và hoạt động,… xác định đầy đủ thông tin mà dữ liệu đầu vào của hệ thống cần cho đề tài.
- Phương pháp phân tích hệ thống: Vận dụng phương pháp này để xác định mục tiêu và mục đích rõ ràng của đề tài, tạo ra các hệ thống và thủ tục để xây dựng được hệ thống game tối ưu và hiệu quả nhất.
- Phương pháp thực nghiệm: Để thiết lập được hệ thống game Minsweeper thì phải trải qua quá trình thực nghiệm, vì vậy cần các công cụ hỗ trợ như Microsoft Visual Studio, Draw.io,…

1.6. Dự kiến kết quả đạt được

- Hoàn thành tốt các mục tiêu đã đề ra.
- Xây dựng hoàn chỉnh giao diện game Minesweeper.
- Kiểm thử thành công các chức năng của game.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Đặt vấn đề

2.1.1. Mô tả bài toán

Mục đích của trò chơi MineSweeper là người chơi phải mở được tất cả các ô không chứa mìn trên bảng chơi mà không kích nổ bất cứ một quả mìn nào. Trong MineSweeper, người chơi điều khiển con trỏ chuột để mở các ô trên bảng chơi, tránh mìn và sử dụng các manh mối số để xác định vị trí các ô chứa mìn.

Để có thể chơi trò chơi này, người chơi cần laptop có cài đặt Microsoft Visual Studio và source trò chơi này. Khi vừa khởi tạo, giao diện chỉ hiển thị thanh menu cho người chơi chọn cấp độ và các tùy chọn khác.

Trước khi bắt đầu trò chơi, người chơi sẽ được cung cấp tùy chọn để chọn cấp độ khó của trò chơi. Cấp độ khó sẽ quy định kích thước của bảng chơi và số lượng mìn được đặt trên bảng:

- *Dễ (Easy):* 9x9 ô vuông với 10 mìn.
- *Trung bình (Medium):* 16x16 ô vuông với 40 mìn.
- *Khó (Hard):* 30x16 ô vuông với 99 mìn.

Sau khi hoàn thành bước chọn cấp độ khó, màn hình sẽ hiển thị giao diện trò chơi với bảng chơi theo kích thước đã chọn và người chơi có thể bắt đầu.

❖ ***Luật chơi:***

Trò chơi bắt đầu với một bảng chơi được sắp xếp ngẫu nhiên với các ô vuông chưa được mở. Mục tiêu của người chơi là mở tất cả các ô không chứa mìn. Khi người chơi nhấn chuột vào một ô:

- *Nếu ô không chứa mìn:* Ô sẽ mở ra và hiển thị một số từ 1 đến 8 cho biết có bao nhiêu ô chứa mìn xung quanh ô đó. Nếu ô mở ra là ô trống, tất cả các ô xung quanh nó sẽ tự động được mở ra cho đến khi gặp các ô chứa số từ 1 đến 8.
- *Nếu ô chứa mìn:* Trò chơi sẽ kết thúc và hiển thị thông báo người chơi đã thua.

Nếu những ô lân cận của một ô đã có đủ số mìn mà vẫn còn các ô trống khác thì những ô đó không có mìn.

Người chơi có thể sử dụng các manh mối số để xác định vị trí các ô chứa mìn và đánh dấu các ô nghi ngờ chứa mìn bằng cách nhấn chuột phải để đặt cờ. Nếu người chơi chắc chắn ô nào chứa mìn, họ có thể đặt cờ vào ô đó để tránh mở nhầm.

Trò chơi tiếp tục cho đến khi tất cả các ô không chứa mìn được mở. Nếu người chơi thành công mở hết các ô không chứa mìn mà không làm nổ mìn nào, trò chơi sẽ kết thúc với chiến thắng cho người chơi. Hệ thống sẽ hiển thị thông báo chúc mừng người chơi đã chiến thắng.

Trò chơi MineSweeper yêu cầu người chơi tập trung, phân tích và sử dụng logic để xác định vị trí mìn, tạo ra trải nghiệm thú vị và đầy thử thách.

2.1.1 Quy trình thuật toán

2.1.1.1 Khởi tạo thanh Menu và các nút chọn cấp độ trò chơi

Khi người chơi vừa khởi chạy chương trình, giao diện trò chơi hiển thị thanh Menu, ô hiển thị số mìn và ô số điểm.

Người chơi chọn cấp độ trò chơi cùng với số mìn tương ứng với từng kích thước. Cụ thể các mức chơi:

- *Dễ (Beginner):* 9x9 ô vuông với 10 mìn.
- *Trung bình (Intermediate):* 16x16 ô vuông với 40 mìn.
- *Khó (Expert):* 16x30 ô vuông với 99 mìn. Trong game Minesweeper, ở cấp độ này, kích thước là 30x16 ô, nhưng để dễ nhìn hơn nên chúng em thay đổi lại kích thước là 16x30 ô.

Sau khi ghi nhận lựa chọn của người chơi, chương trình sẽ chuẩn bị các thành phần khác để chuẩn bị cho người chơi.

2.1.1.2 Đặt vị trí các quả mìn

Chương trình sau khi ghi nhận lựa chọn của người chơi thì chuẩn bị đặt các quả mìn vào mảng 2 chiều có kích thước tương ứng với kích thước của cấp độ mà người chơi lựa chọn.

Chương trình sẽ random vào ô bất kì và gán giá trị bằng 10 vào ô đó (mặc định ô có giá trị 10 sẽ là nơi mìn được đặt). Số ô cần gán chính là số quả mìn tương ứng với cấp độ mà người chơi chọn.

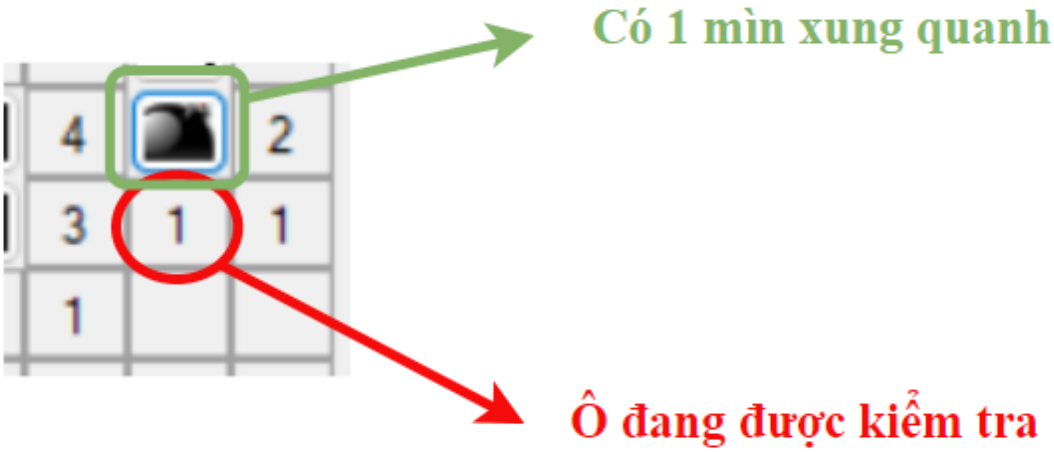
Trong lúc gán giá trị cho các ô sẽ chứa mìn, chương trình sẽ thực hiện kiểm tra xem ô được random đó đã được gán giá trị hay chưa, nếu chưa thì mới gán giá trị vào ô đó.

2.1.1.3 Tạo giá trị cho các ô không chứa mìn

Các ô không chứa mìn sẽ nhận các giá trị từ 1 – 8 hoặc không chứa giá trị nào (ô trống). Số ô không chứa mìn bằng tổng số ô trừ đi số ô chứa mìn.

Chương trình sẽ lần lượt kiểm tra từng ô trong mảng:

- Nếu ô đó chứa mìn thì bỏ qua.
- Nếu ô đó không chứa mìn thì đếm 8 ô xung quanh nó có bao nhiêu ô có mìn rồi gán giá trị cho nó là số ô có mìn đếm được. Nếu xung quanh ô đó không chứa mìn thì là ô trống, không cần phải gán giá trị.



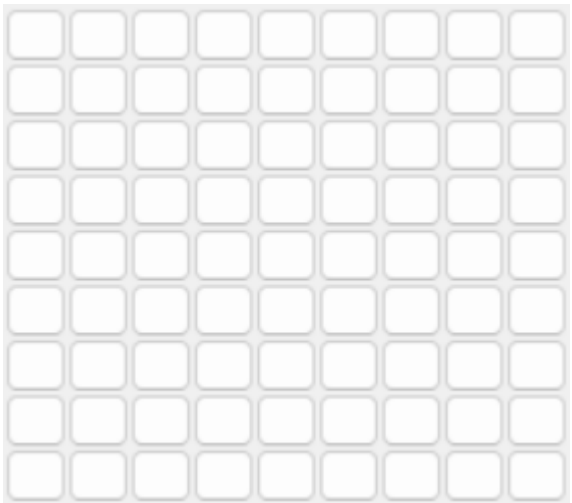
Hình 2. 1 Xung quanh ô được kiểm tra có 1 mìn

Hình 2. 2 Không có mìn xung quanh ô được kiểm tra

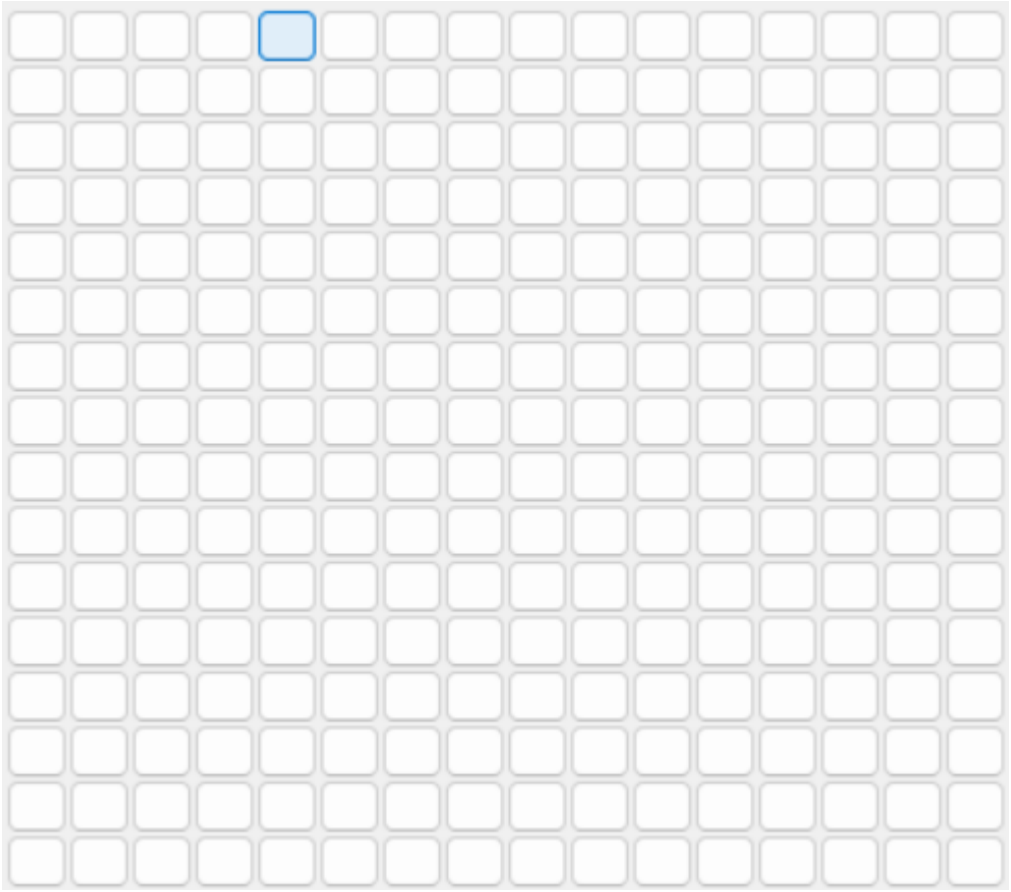
2.1.1.4 Khởi tạo các ô của bảng trò chơi

Chương trình lặp qua từng vị trí để tạo các ô và thiết đặt các thuộc tính cho các ô. Sau đó gán các ô vào mảng ô.

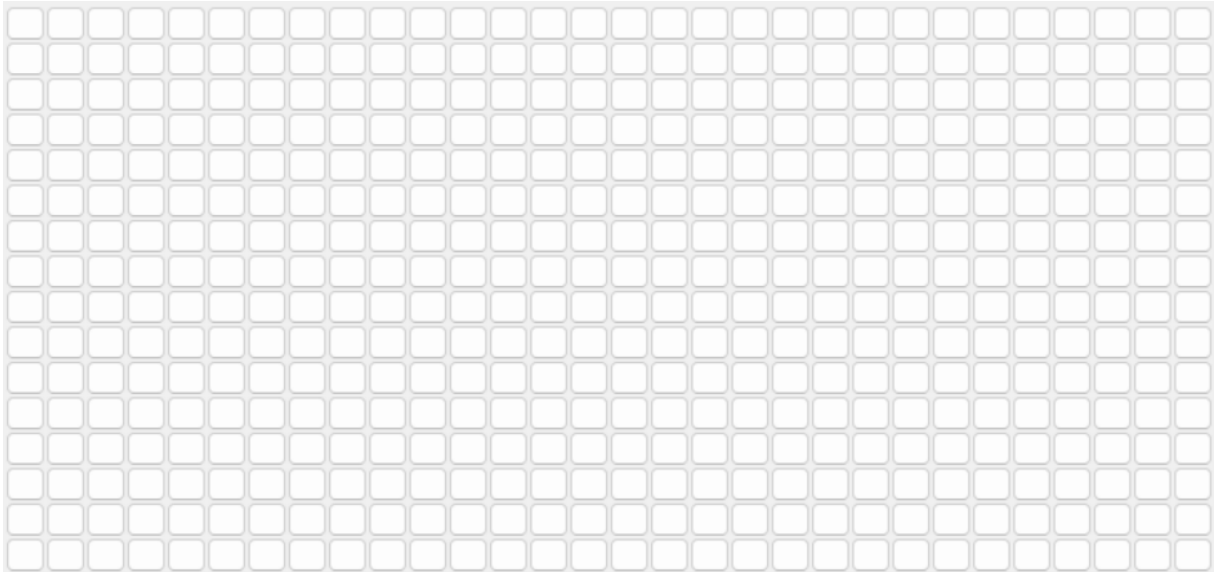
Lý do chương trình tạo các ô sau khi tạo giá trị của ô đó là vì trong WinForms các điều khiển được tạo ra sau sẽ nằm chồng lên điều khiển đã được tạo ra trước đó. Như vậy các ô sẽ che đi giá trị của nó và người chơi sẽ không thể thấy được giá trị của nó cho đến khi mở ô đó ra.



Hình 2. 3 Bảng trò chơi ở cấp độ Easy 9x9



Hình 2. 4 Bảng trò chơi ở cấp độ Medium 16x16

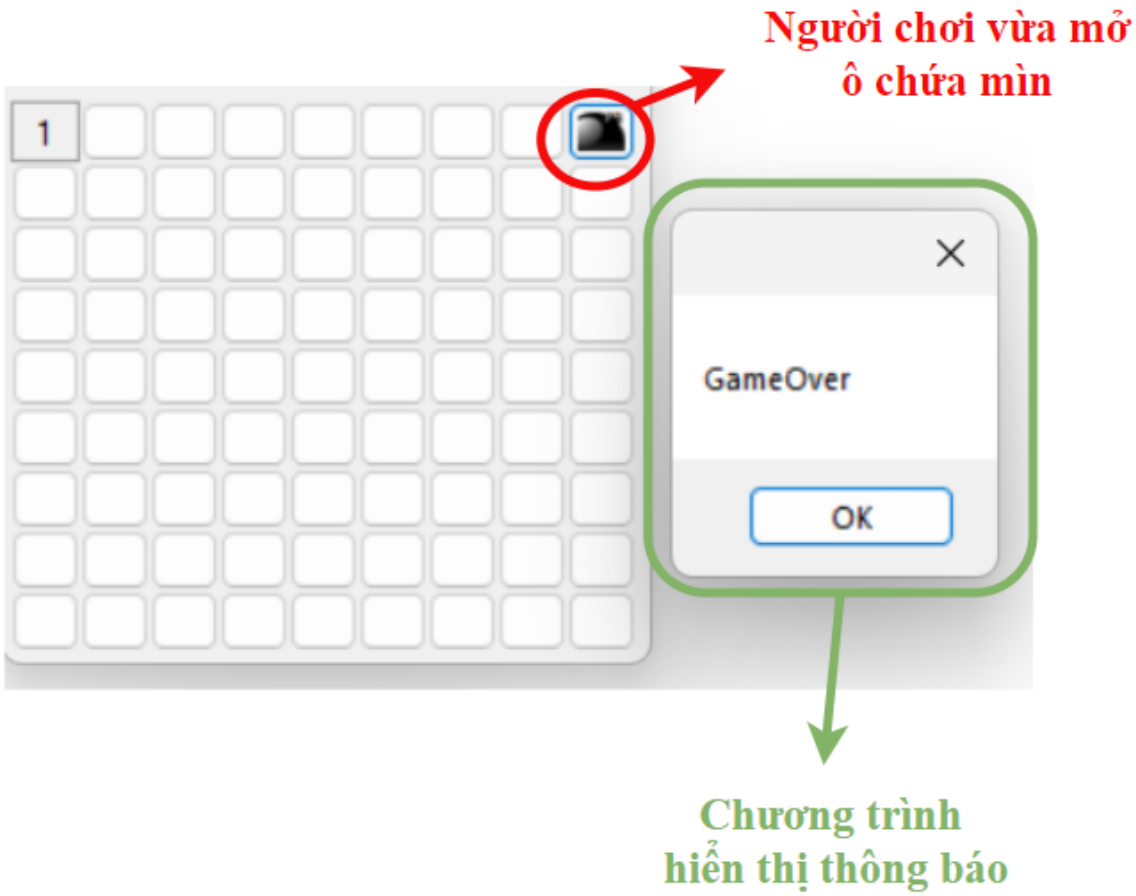


Hình 2. 5 Bảng trò chơi ở cấp độ Hard 16x30

2.1.1.5 Bước đi của người chơi và hiển thị ô người chơi chọn

Người chơi lần lượt chọn một ô để mở. Chương trình sẽ kiểm tra và mở ô mà người chơi chọn:

- Nếu ô đó là ô chứa mìn thì kết thúc trò chơi và hiện thông báo "GameOver".



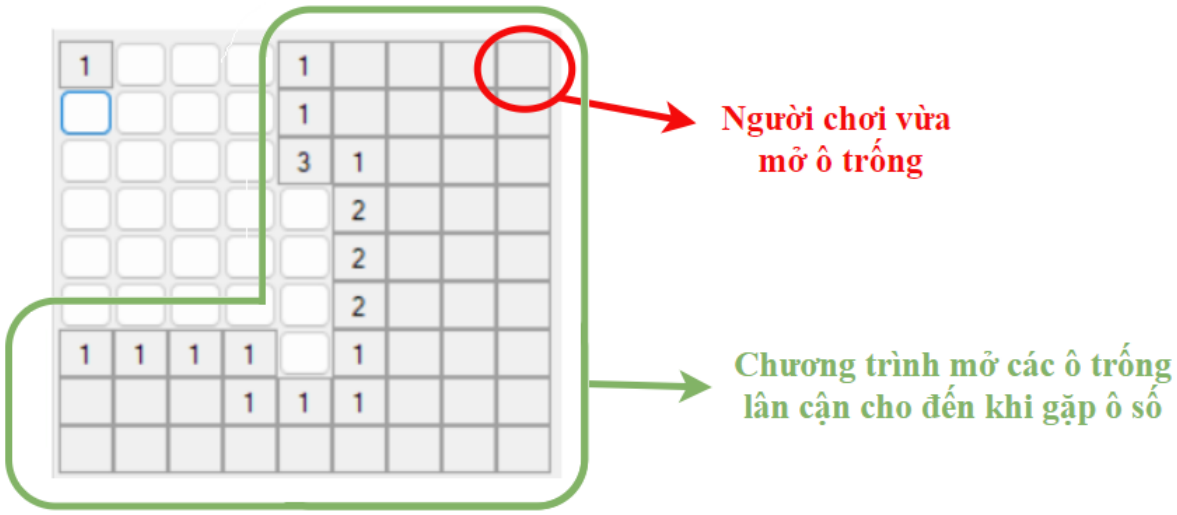
Hình 2. 6 Người chơi chọn trúng ô chứa mìn

- Nếu ô đó không phải ô chứa mìn thì hiển thị giá trị của ô đó (tương ứng với số mìn xung quanh ô đó) và người chơi vẫn tiếp tục chơi.

Hình 2. 7 Người chơi mở trúng ô số

2.1.1.6 Hiển thị các ô lân cận ô trống

Khi người chơi mở trúng ô trống, chương trình sẽ hiển thị các ô trống lân cận ô đó cho đến khi gặp ô số.



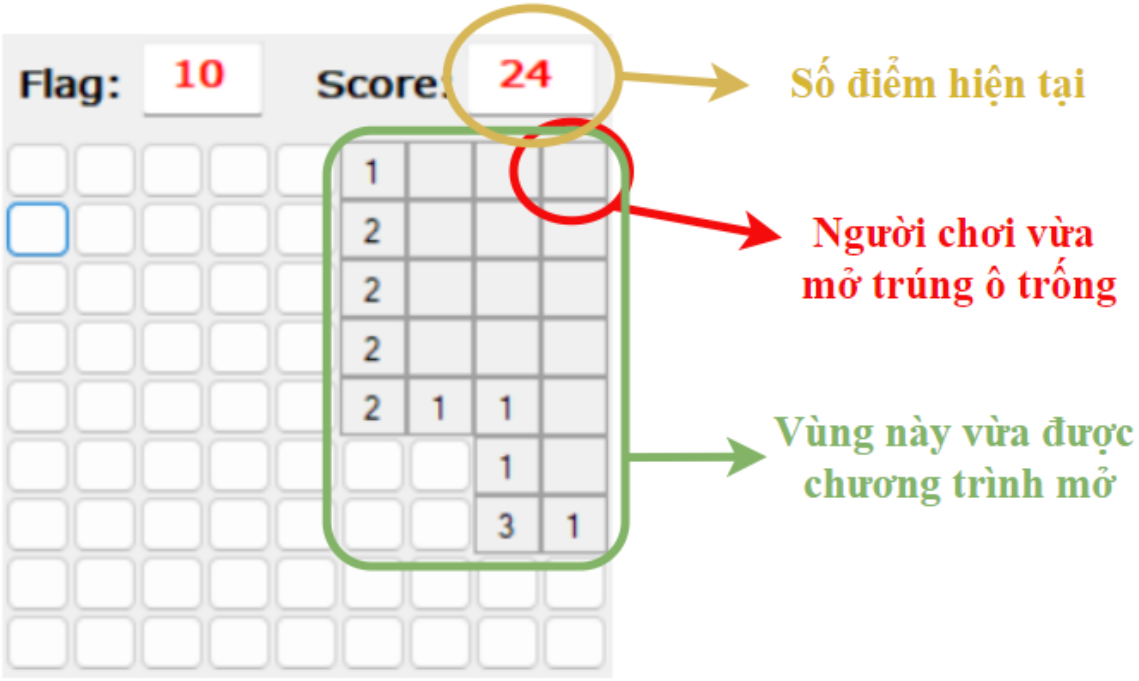
Hình 2. 8 Người chơi mở trúng ô trống

2.1.1.7 Tính điểm

Khi vừa bắt đầu trò chơi, số điểm bằng 0. Tổng số điểm tối đa người chơi có thể đạt được bằng tổng số ô trừ đi số mìn.

Chương trình sẽ cộng số điểm bằng số ô được mở mỗi khi người chơi click chuột và mở trúng ô trống. Nếu người chơi mở ô số thì cộng 1 điểm.

Điểm số sẽ được hiển thị ở ô Score phía bên trên bảng trò chơi.



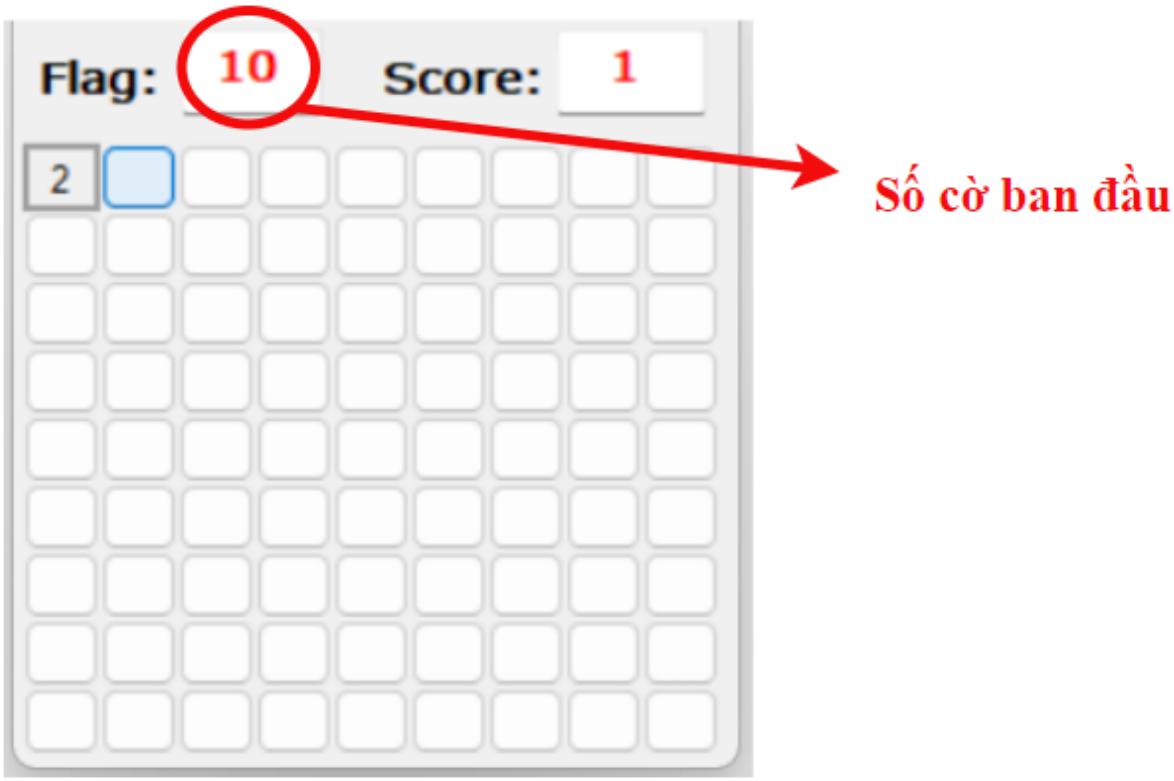
Hình 2. 9 Số điểm khi đã mở được 24 ô

2.1.1.8 Tính số lá cờ được cắm

Khi mới bắt đầu màn chơi mới, số cờ có thể cắm (bằng số mìn) được hiển thị tại ô Flag phía bên trên bảng.

Khi người chơi cắm 1 cờ vào ô chưa được mở, số lá cờ sẽ trừ đi 1. Khi người chơi bỏ cắm cờ tại ô đã cắm thì số lá cờ sẽ tăng lên 1. Người chơi không cắm được cờ vào các ô đã mở.

Nếu người chơi cắm hết số lá cờ đã cho thì sẽ không được cắm cờ nữa.



Hình 2. 10 Số cờ ban đầu khi người chơi vừa bắt đầu màn mới hiển thị tại ô Flag

Hình 2. 11 Số cờ giảm đi khi người chơi đã cắm 1 lá

Hình 2. 12 Số cờ tăng lên khi người chơi vừa bỏ cắm 1 lá

Hình 2. 13 Người chơi không thể cắm cờ khi chỉ còn 0 lá

2.1.1.9 Kiểm tra điều kiện thắng

Khi người chơi đạt điểm số tối đa thì chương trình sẽ hiện thông báo người chơi đã thắng "You win". Sau khi tắt thông báo, người chơi cũng sẽ không thể điều khiển bảng trò chơi nữa mà phải mở màn chơi mới.

Điểm số tối đa cho mỗi cấp độ trò chơi như sau:

- *Dễ (Easy):* 9 x 9 – 10 = 71 điểm
- *Trung bình (Medium):* 16 x 16 – 40 = 216 điểm
- *Khó (Hard):* 30 x 16 – 99 = 381 điểm

Hình 2. 14 Thông báo người chơi đã thắng khi đạt điểm số tối đa

Hình 2. 15 Chương trình không cho phép người chơi điều khiển bản trò chơi sau khi thắng

2.1.1.10 Kết thúc trò chơi

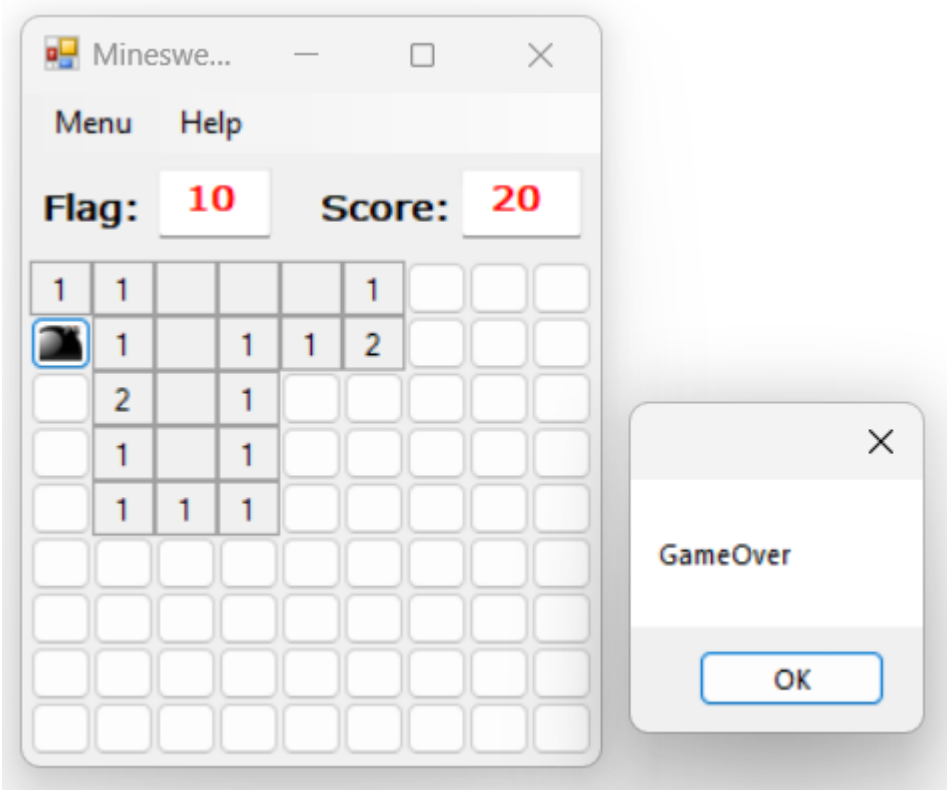
Trò chơi sẽ kết thúc trong 2 tình huống sau:

- Khi người chơi mở trúng ô có mìn. Thông báo hiển thị "GameOver".
- Khi người chơi đạt điểm số tối đa. Thông báo hiển thị "You win".

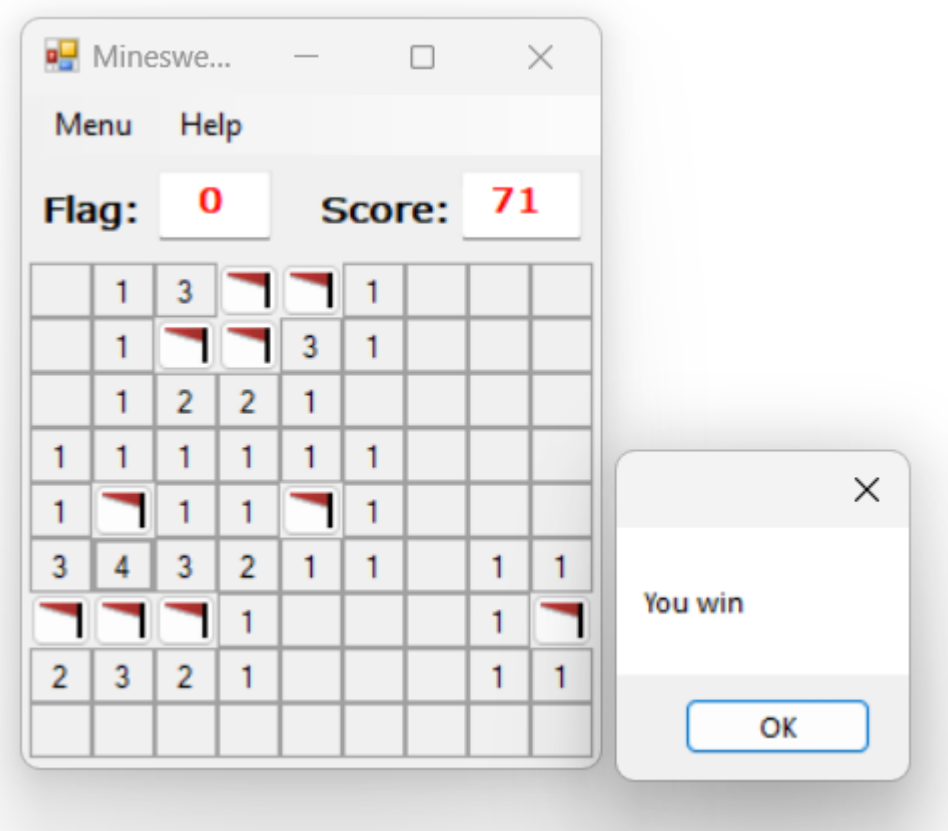
Trước khi hiển thị thông báo, chương trình sẽ tự động hiện lên tất cả vị trí chứa mìn.

Tiếp theo, chương trình sẽ hiện thông báo tương ứng với mỗi tình huống trên và sẽ không cho phép người chơi điều khiển bảng trò chơi nữa.

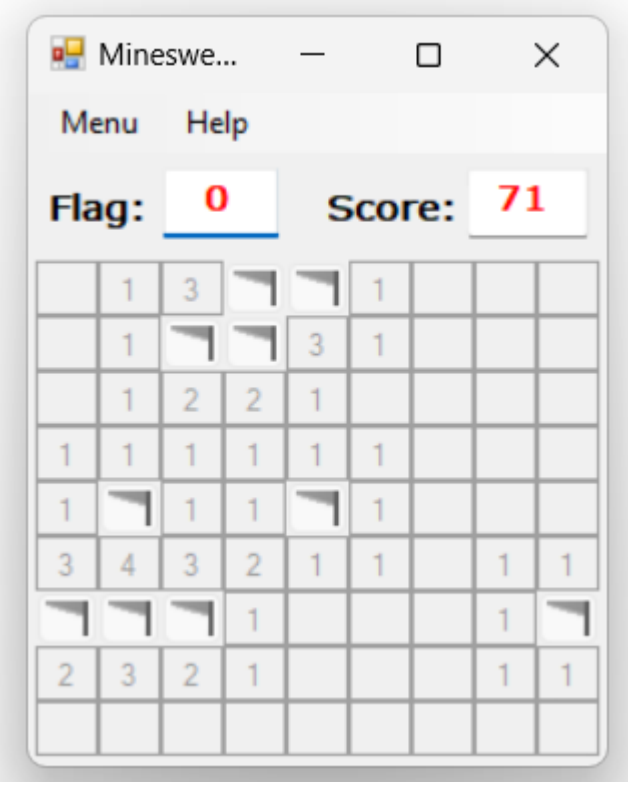
Nếu người chơi muốn bắt đầu lại màn mới thì mở Menu và chọn cấp độ trò chơi.



Hình 2. 16 Người chơi thua và kết thúc trò chơi với thông báo "GameOver"

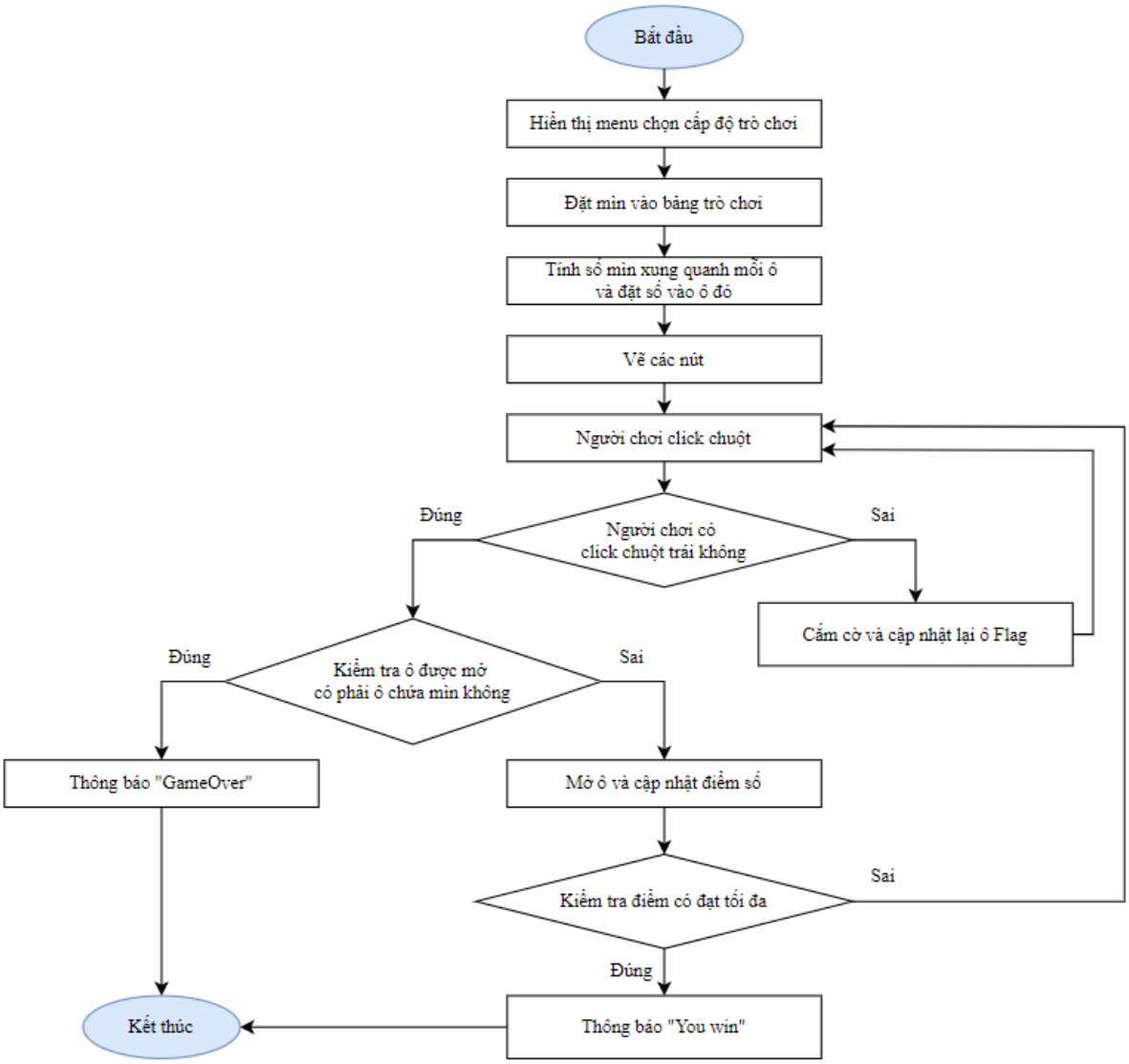


Hình 2. 17 Người chơi thắng và kết thúc trò chơi với thông báo "You win"



Hình 2. 18 Chương trình không cho phép người chơi điều khiển bảng trò chơi sau khi kết thúc

2.1.1.11 Quy trình thuật toán qua sơ đồ flowchart



Hình 2. 19 Quy trình thuật toán Game Minesweeper

❖ Tóm tắt quy trình thuật toán game Minesweeper như sau:

Bước 1: Hiển thị menu chọn cấp độ trò chơi.

Bước 2: Đặt mìn vào bảng trò chơi.

Bước 3: Tính số mìn xung quanh mỗi ô và đặt số vào ô đó.

Bước 4: Vẽ các nút vào bảng trò chơi.

Bước 5: Người chơi click chuột.

Bước 6: Kiểm tra người chơi có click chuột trái không. Nếu có thì chuyển sang bước 8 (bỏ qua bước 7). Nếu không thì chuyển sang bước 7.

Bước 7: Cắm cờ và cập nhật lại ô Flag. Quay lại bước 5.

Bước 8: Kiểm tra ô được mở có phải ô chứa mìn không. Nếu không thì chuyển sang bước 9. Nếu đúng thì chương trình hiển thị tất cả vị trí mìn rồi thông báo kết thúc game.

Bước 9: Mở ô và cập nhật điểm số.

Bước 10: Kiểm tra điểm số đã đạt tối đa chưa. Nếu chưa thì quay lại bước 5. Nếu đúng thì chương trình hiển thị tất cả vị trí mìn rồi thông báo người chơi chiến thắng rồi kết thúc.

2.1.2 Sơ lược về thuật toán

2.1.2.1 Các sự kiện click trong giao diện trò chơi

Trong WinForms, sự kiện click chuột là một trong những sự kiện phổ biến và quan trọng nhất. Sự kiện này xảy ra khi người dùng nhấp vào một control như button, label, textbox,… Có nhiều sự kiện liên quan đến chuột, nhưng phổ biến nhất là sự kiện Click, MouseClick, MouseDown, MouseUp, MouseMove.

Trong game này sẽ khai báo các phương thức sau để xử lý các sự kiện click chuột:

- **private void levelToolStripMenuItem_Click(object sender, EventArgs e):** phương thức xử lý sự kiện khi người chơi click chuột vào nút Menu (ToolStripMenuItem).
- **private void BtnClick(object sender, EventArgs e):** phương thức xử lý sự kiện khi người chơi click chuột vào một ô bất kì trên bảng trò chơi.
- **private void BtnMouseUp(object sender, MouseEventArgs e):** phương thức xử lý sự kiện khi người chơi click chuột và nhả ra trên một ô bất kì trên bảng trò chơi.
- **private void exitToolStripMenuItem_Click(object sender, EventArgs e):** phương thức xử lý sự kiện khi người dùng click chuột vào nút Exit (exitToolStripMenuItem) trong thanh Menu.

Chương trình cũng sử dụng toán tử gán cộng (+ =) để đính kèm xử lý sự kiện:

- **easyToolStripMenuItem.Click += new EventHandler(levelToolStripMenuItem _Click);:**

- **mediumToolStripMenuItem.Click += new EventHandler(levelToolStripMenuItem _Click);**
- **hardToolStripMenuItem.Click += new EventHandler(levelToolStripMenuItem _Click);**
- **btn.Click += BtnClick;**
- **btn.MouseUp += BtnMouseUp;**

Trong đó:

- easyToolStripMenuItem, mediumToolStripMenuItem, hardToolStripMenuItem là các mục tương ứng với 3 mục trong giao diện người dùng là Easy, Medium, Hard. Đây là các mục con trong thanh Menu.
- .Click là sự kiện xảy ra khi người dùng nhấn vào các mục cấp độ trò chơi.
- new EventHandler(levelToolStripMenuItem _Click) là cách gán phương thức xử lý sự kiện cho sự kiện Click.
- levelToolStripMenuItem _Click là tên của phương thức sẽ được gọi khi sự kiện click xảy ra.

Để ngăn chặn xử lý sự kiện không bị gọi khi sự kiện này được tăng lên thì hủy đăng ký từ sự kiện này. Sử dụng toán tử gán trừ (- =) để bỏ đăng ký từ một sự kiện. Trong chương trình có hủy bỏ đăng ký từ một sự kiện: **btn.Click -= BtnClick;**

2.1.2.2 Đặt vị trí các quả mìn

Trước khi người chơi bắt đầu trò chơi, chương trình sẽ cài đặt mìn vào các ô trong bảng trò chơi. Thuật toán "GenerateMines" được sử dụng để đặt ngẫu nhiên số mìn vào bảng. Phương thức này sẽ được gọi sau khi người chơi chọn cấp độ trò chơi.

2.1.2.3 Tạo giá trị cho các ô không chứa mìn

Bên cạnh việc cài đặt mìn, chương trình cũng sẽ cài đặt giá trị ở mỗi ô. Thuật toán "GeneratePositionValue" được sử dụng để đặt giá trị vào mỗi ô trong bảng trò chơi dựa trên sự tính toán số lượng mìn xung quanh ô đó. Phương thức này cũng sẽ được gọi sau khi người chơi chọn cấp độ trò chơi.

2.1.2.4 Tạo các ô trong bảng trò chơi

Các phần tử quan trọng cần có trong trò chơi chính là các nút để mở các ô trong bảng trò chơi. Thuật toán "GenerateButtons" được sử dụng để vẽ các nút cho bảng trò chơi. Số nút tạo ra dựa trên độ khó mà người chơi lựa chọn. Phương thức này cũng sẽ được gọi sau khi người chơi chọn cấp độ trò chơi.

2.1.2.5 Hiển thị các ô lân cận ô trống

Trong game Minesweeper, khi người chơi mở trúng ô trống, chương trình sẽ mở các ô trống lân cận ô đó cho đến khi gặp ô số. Thuật toán "OpenAdjacentEmptyTile" được sử dụng để làm điều đó. Phương thức này sẽ được gọi đệ quy để mở vùng gồm các ô trống ra. Bên cạnh đó, thuật toán này còn giúp cập nhật số điểm hiện tại của người chơi và hiển thị điểm lên ô Score.

2.1.2.6 Cắm cờ

Để hỗ trợ người chơi dễ dàng tìm ra dược các vị trí chứa số và ô trống và tránh được những vị trí chứa mìn, chương trình sử dụng thuật toán cắm cờ được cài đặt trong phương thức xử lý sự kiện **private void BtnMouseUp(object sender, MouseEventArgs e)**. Các câu lệnh xử lý việc cắm cờ sẽ được thực thi khi người chơi click chuột phải vào các ô trong bảng trò chơi. Bên cạnh đó, thuật toán cắm cờ còn giúp cập nhật số cờ hiện tại mà người chơi có thể sử dụng và hiển thị lên ô Flag.

2.1.2.7 Hiển thị tất cả vị trí của mìn

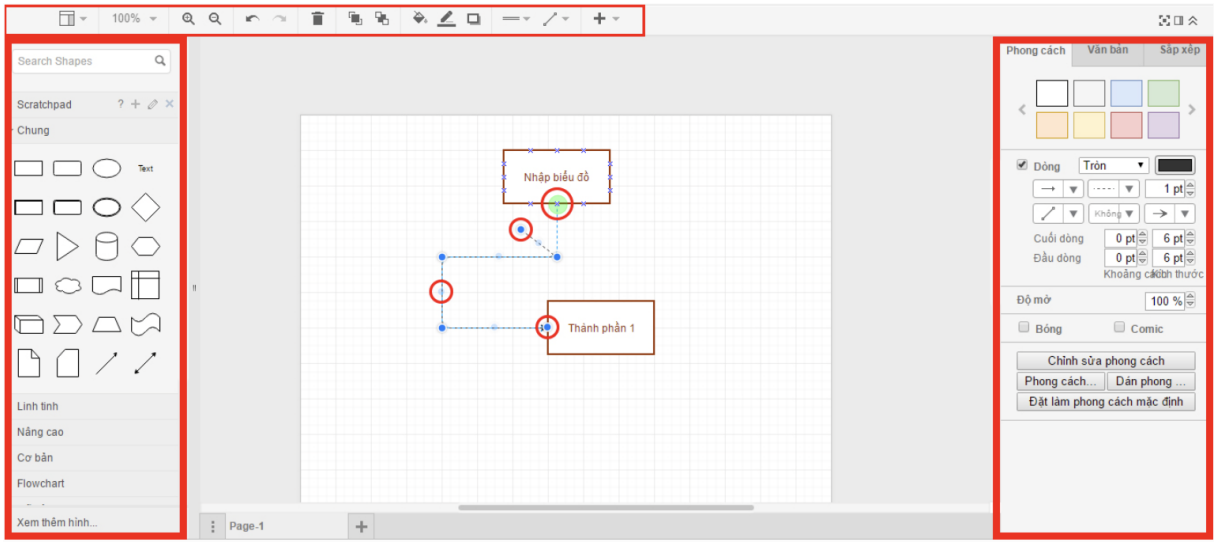
Để người chơi tin tưởng rằng mình chơi thua khi chương trình thông báo "GameOver" hoặc người chơi muốn biết kết quả chính xác vị trí của các quả mìn, chương trình sử dụng thuật toán "ShowAllMines" để hiển thị tất cả vị trí của mìn khi game kết thúc. Thuật toán kiểm tra các vị trí có mìn và hiển thị chúng lên cho người xem nhìn thấy.

2.2. Công cụ hỗ trợ

2.2.1 Draw.io

Draw.io là phần mềm biểu đồ trực tuyến miễn phí. Bạn có thể sử dụng nó để tạo lưu đồ, sơ đồ mạng, UML trực tuyến, biểu đồ ER, thiết kế sơ đồ cơ sở dữ liệu, xây dựng

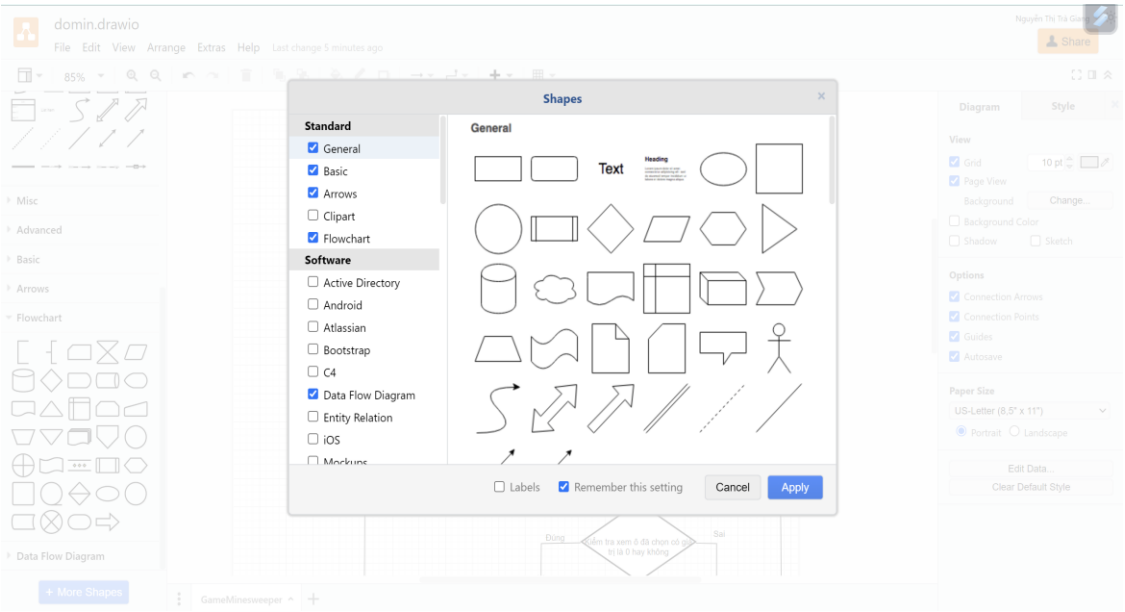
BPMN trực tuyến, sơ đồ mạch điện,... Draw.io cũng cho phép người dùng tương tác với các ứng dụng khác để nhập và xuất các tệp đồ hoạ.



Hình 2. 20 Giao diện Draw.io

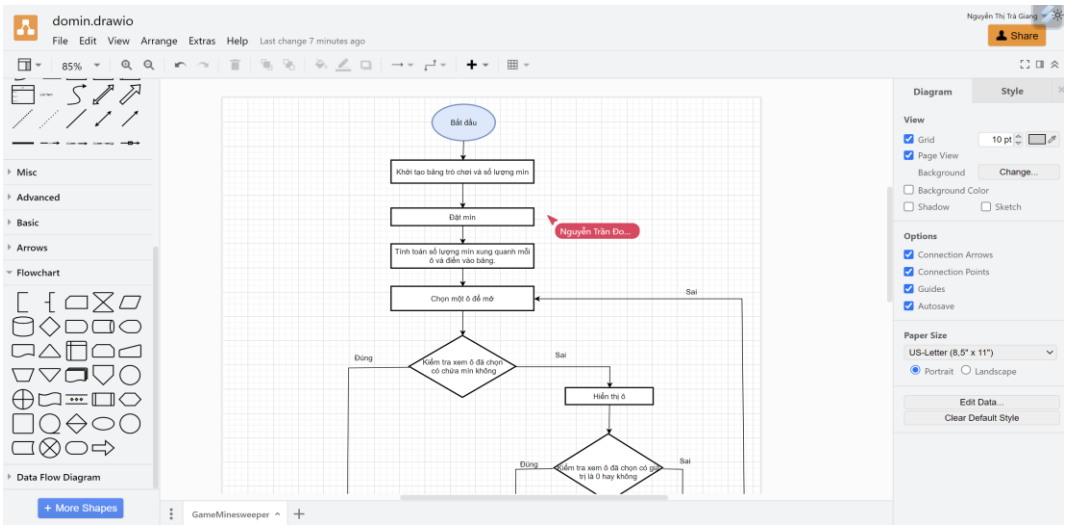
❖ Tính năng

- Tạo các hình ảnh, biểu đồ và sơ đồ:
  - Cung cấp các công cụ vẽ để tạo các hình ảnh, biểu đồ và sơ đồ cho các mục đích khác nhau, từ thiết kế giao diện đến tư duy và phân tích dữ liệu.
  - Cho phép bạn vẽ các đối tượng, hình dạng, đường thẳng, mũi tên và các ký hiệu khác để mô tả ý tưởng.



Hình 2. 21 Các tính năng tạo hình ảnh, vẽ sơ đồ trong Draw.io

- Chia sẻ và tương tác trực tuyến:
  - Cho phép người dùng chia sẻ tài liệu của mình với người khác, có thể chỉ định quyền truy cập của từng người nhận như: quyền xem, chỉnh sửa, chia sẻ lại.
  - Cho phép nhiều người dùng cùng làm việc trên cùng một bản vẽ. Khi có bất kỳ thay đổi nào trên bản vẽ, các thay đổi đó sẽ được đồng bộ hóa và hiển thị trực tiếp trên màn hình của tất cả người dùng khác đang làm việc trên bản vẽ đó.

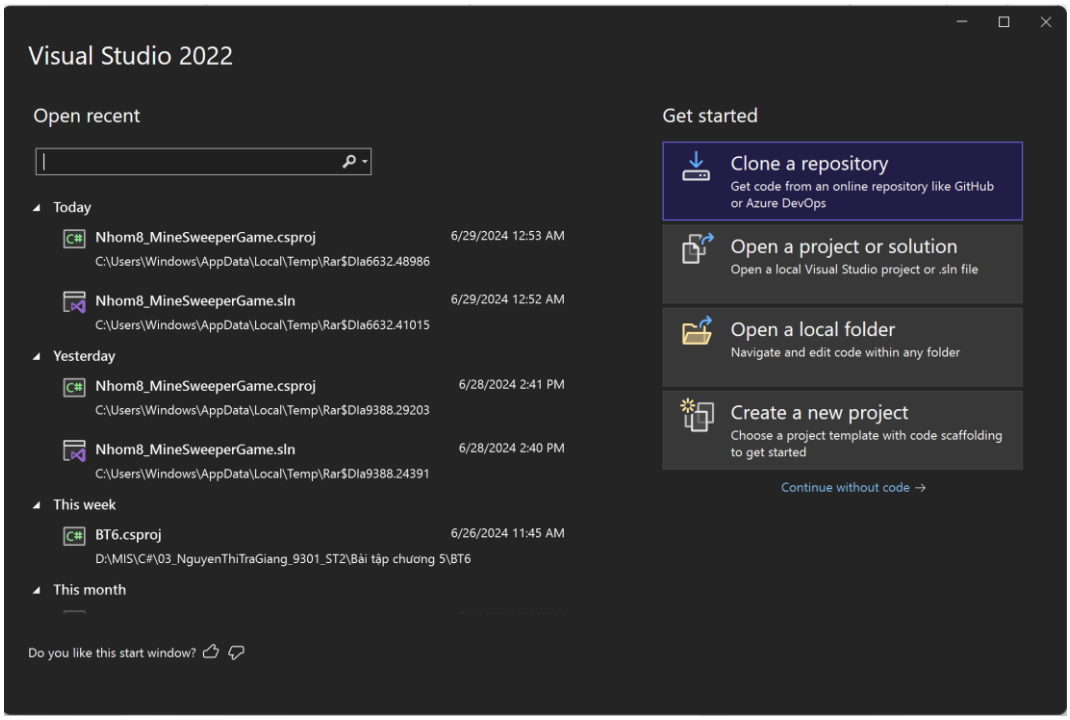


Hình 2. 22 Tính năng chia sẻ và tương tác trực tuyến trong Draw.io

2.2.2 Microsoft Visual Studio

Visual Studio IDE (Integrated Development Environment) là một công cụ hỗ trợ lập trình mã nguồn mạnh mẽ và toàn diện được phát triển bởi Microsoft. Visual Studio chủ yếu được sử dụng cho các lập trình viên làm việc với ngôn ngữ C#, nhưng cũng hỗ trợ nhiều ngôn ngữ lập trình khác như Visual Basic, C++, Python, và nhiều ngôn ngữ khác.

Visual Studio là môi trường phát triển tích hợp đa năng, cung cấp nhiều công cụ và tính năng giúp lập trình viên viết, kiểm tra và gỡ lỗi mã nguồn một cách hiệu quả. Công cụ này hoạt động tốt trên nhiều nền tảng hệ điều hành như Windows, MacOS, và có phiên bản Visual Studio Code hỗ trợ Linux.

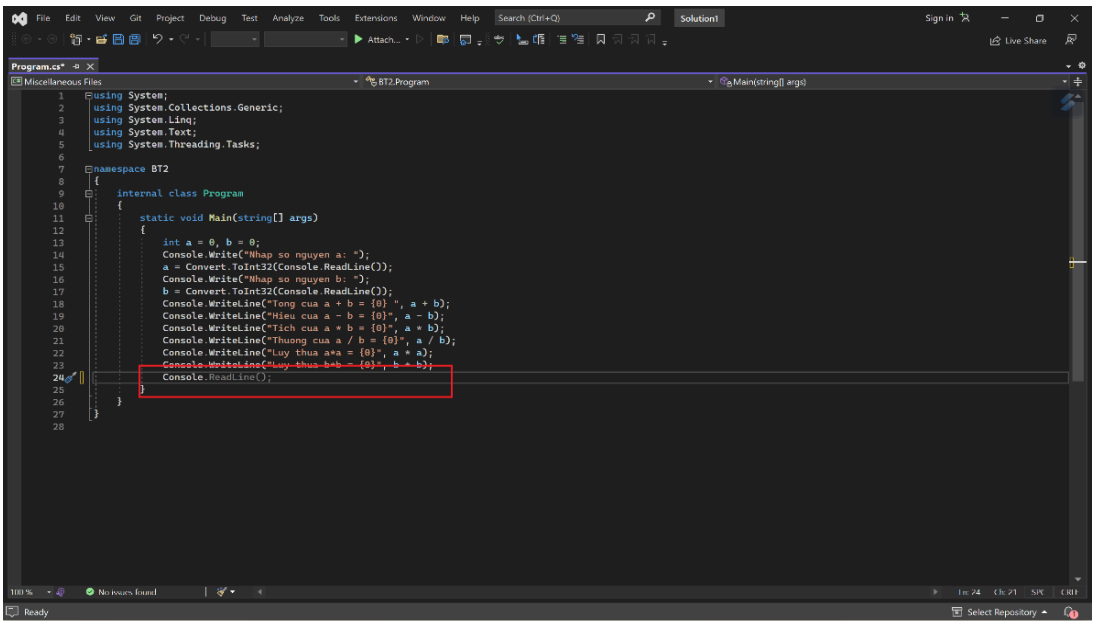


Hình 2. 23 Giao diện Visual Studio phiên bản 2022

❖ **Tính năng**

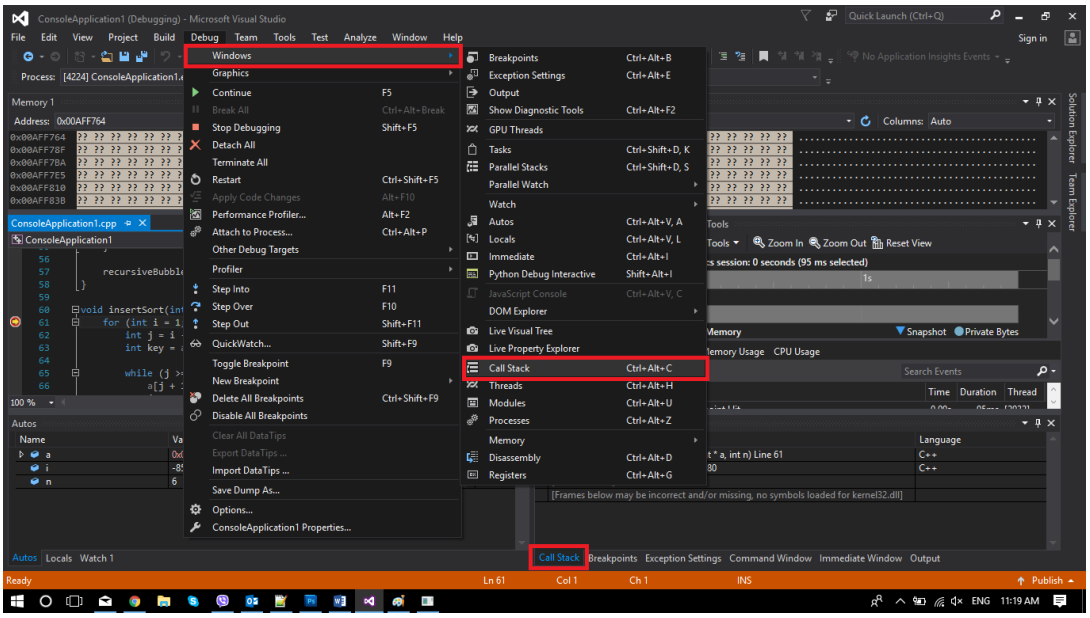
- ***Auto - complete:***
  - Tìm kiếm và gợi ý các từ khoá, phương thức, và biến trong các thư viện và module đã được cài đặt.
  - Đề xuất mã nguồn để hoàn thành các tên biến và phương thức.

- Cải thiện hiệu suất và tăng tốc độ phát triển.
- Giúp tránh sai sót khi viết mã bằng cách giảm thiểu sự phụ thuộc vào trí nhớ.

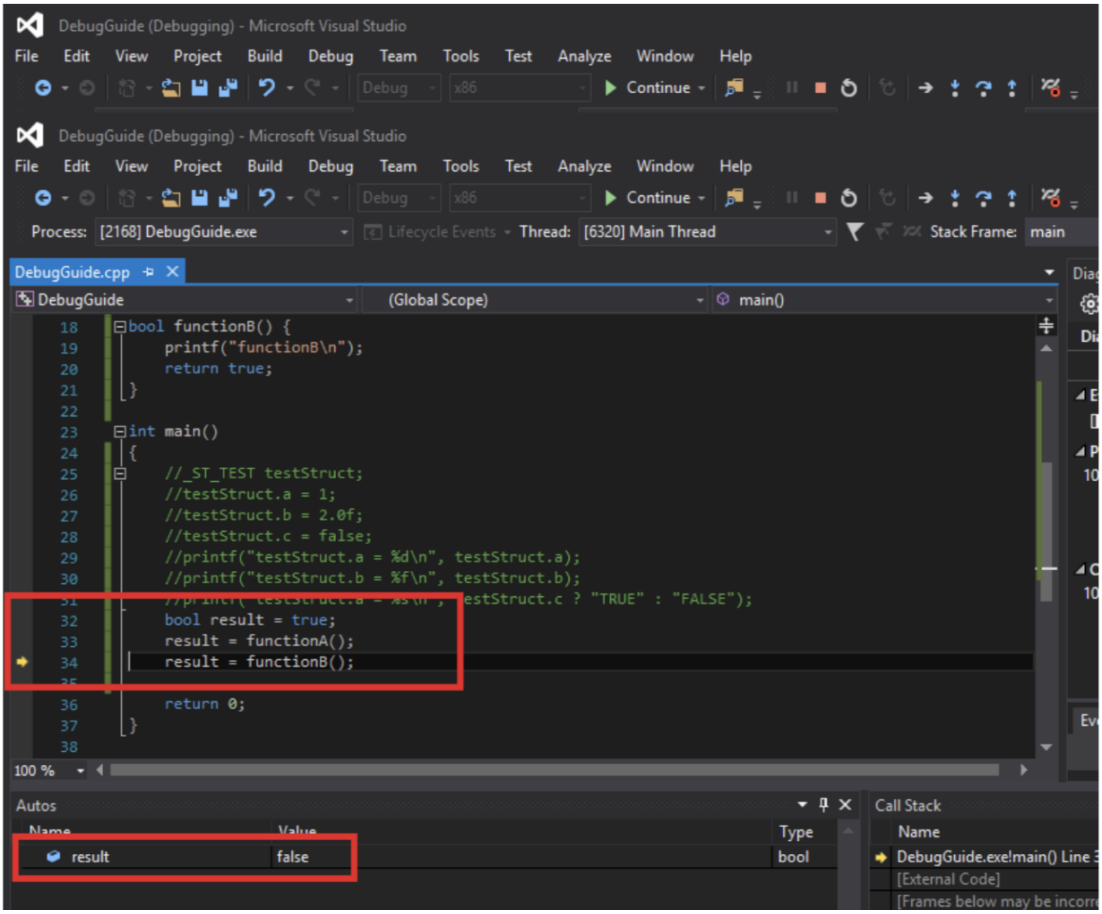


Hình 2. 24 Tính năng auto - complete trong Visual Studio

- ***Debugger:***
  - Giúp tìm ra các lỗi trong mã nguồn bằng cách cung cấp thông tin chi tiết về các biến và hành động của chương trình trong quá trình thực thi.
  - Cho phép chạy chương trình theo từng bước để kiểm tra lỗi và giải quyết các vấn đề phát sinh.
  - Đảm bảo mã nguồn chạy đúng và hiệu quả.



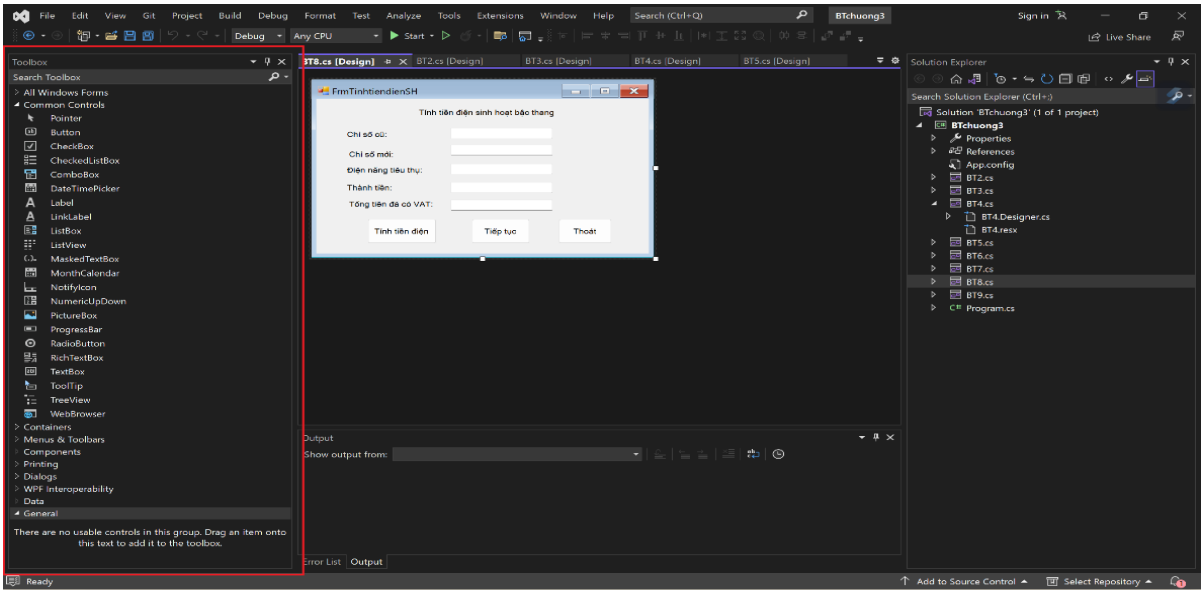
Hình 2. 25 Sử dụng Call Stack để Debug trong Visual Studio



Hình 2. 26 Chạy dòng lệnh được chỉ định để Debug trong Visual Studio

- **Toolbox:**

- Toolbox trong Visual Studio là một bảng chứa các điều khiển và thành phần bạn có thể kéo và thả vào các biểu mẫu (forms) trong ứng dụng của mình.
- Toolbox chứa các điều khiển tiêu chuẩn như nút (Button), hộp văn bản (TextBox), nhãn (Label), hộp kiểm (CheckBox), và nhiều điều khiển khác.
- Ngoài ra, Toolbox cũng hỗ trợ các điều khiển tùy chỉnh và thư viện của bên thứ ba mà bạn có thể thêm vào để mở rộng chức năng của ứng dụng.
- Các điều khiển trong Toolbox giúp lập trình viên tạo giao diện người dùng nhanh chóng và dễ dàng thông qua thao tác kéo và thả.
- Các nhóm điều khiển trong Toolbox được sắp xếp theo danh mục, giúp dễ dàng tìm kiếm và sử dụng các điều khiển cần thiết.



Hình 2. 27 Hộp thoại ToolBox trong Visual Studio

- ***Hỗ trợ đa nền tảng:***
  - Visual Studio hỗ trợ phát triển ứng dụng trên nhiều nền tảng khác nhau, bao gồm Windows, MacOS và Linux thông qua Visual Studio Code.

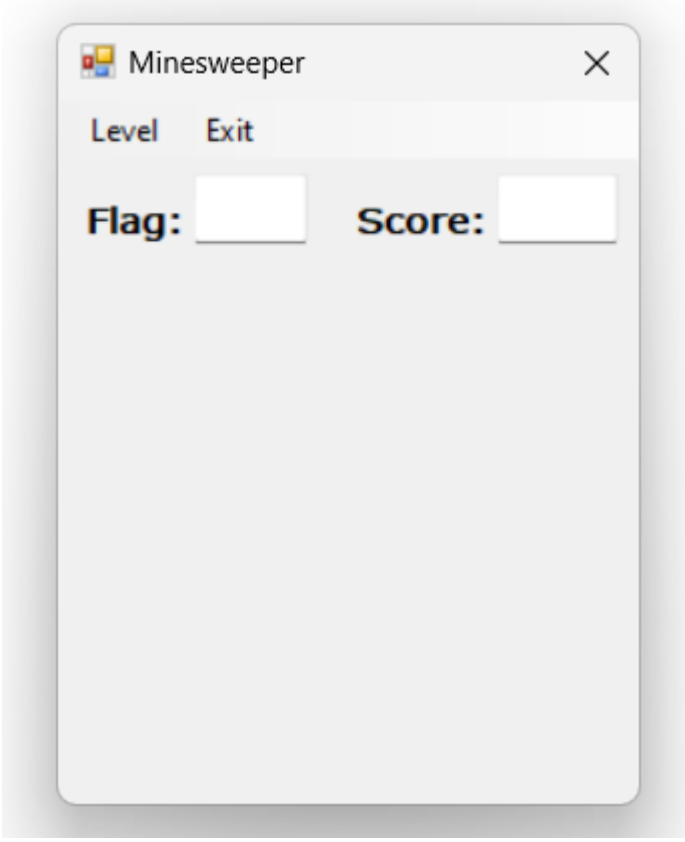
- Tạo điều kiện thuận lợi cho lập trình viên làm việc trên nhiều môi trường khác nhau.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ THUẬT TOÁN

3.1. Giao diện

3.1.1 Giao diện đầu trò chơi

Giao diện đầu trò chơi là giao diện hiển thị khi người chơi khởi động game. Đây là một phần quan trọng trong trò chơi vì nó cung cấp cho người chơi các tùy chọn để bắt đầu trò chơi. Giao diện đầu trò chơi thường bao gồm các tính năng như chọn cấp độ chơi, cài đặt game, hướng dẫn chơi. Giao diện đầu trò chơi thường được thiết kế đơn giản, dễ sử dụng.



Hình 3. 1 Giao diện đầu trò chơi

❖ ***Thành phần chính của giao diện:***

- menuStrip: là một MenuStrip nằm trong nhóm điều khiển Menu & Toolbars, gồm 2 nhóm chức năng:
    - levelToolStripMenuItem: là một item Level có 3 chức năng con:

- ✓ easyToolStripMenuItem: chức năng biểu thị cấp độ trò chơi Easy.
- ✓ mediumToolStripMenuItem: chức năng biểu thị cấp độ trò chơi Medium.
- ✓ hardToolStripMenuItem: chức năng biểu thị cấp độ trò chơi Hard.

- exitToolStripMenuItem: là một item có chức năng Exit.

- pnlInfor: là một panel nằm trong nhóm điều khiển Containers dùng để chứa các điều khiển sau:
  - lbFlag: là một label hiển thị chữ "Flag" để cho người chơi biết ô bên cạnh là số lá cờ hiện tại.
  - txtFlag: là một textbox nằm bên cạnh lbFlag để hiển thị số lá cờ hiện tại.
  - lbScore: là một label hiển thị chữ "Score" để cho người chơi biết ô bên cạnh là số điểm hiện có.
  - txtScore: là một textbox nằm bên cạnh lbScore để hiển thị số điểm hiện có.
- pnlBody: là một panel. Đây là vùng chứa các ô tạo thành bảng trò chơi.

❖ **Cụ thể các xây dựng như sau:**

- Đầu tiên tạo một MenuStrip có tên là menuStrip và có các yếu tố sau:
  - Items.AddRange: phương thức này thêm một mảng ToolStripItem[] vào MenuStrip. Ở trong game này, 2 menu item được thêm vào là levelToolStripMenuItem và exitToolStripMenuItem.
  - Location: thuộc tính này đặt vị trí của MenuStrip trong form. Ở trong game này, MenuStrip nằm ở tọa độ (0, 0) ở góc bên trái form.
  - Name: thuộc tính này đặt tên cho MenuStrip là "menuStrip".

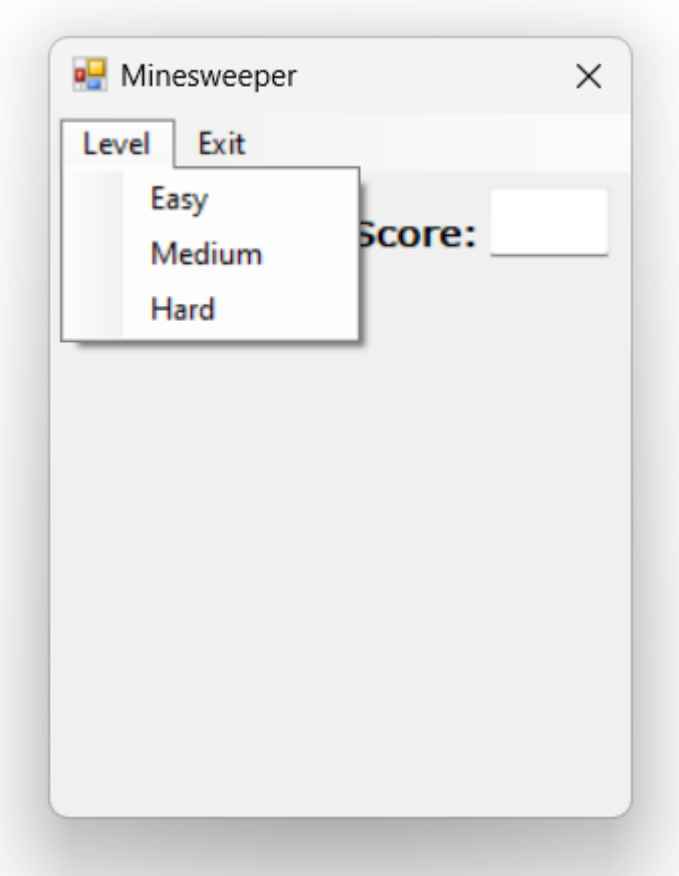
- Size: thuộc tính đặt kích thước cho MenuStrip có chiều rộng là 309 pixel và chiều cao là 28 pixel.

- Các menu item levelToolStripMenuItem (text = “Level”) và exitToolStripMenuItem (text = “Exit”) được thêm vào MenuStrip cũng có các thuộc tính tương tự như MenuStrip.
- Tiếp theo, thêm vào levelToolStripMenuItem các menu item easyToolStripMenuItem (text = “Easy”), mediumToolStripMenuItem (text = “Medium”) và hardToolStripMenuItem (text = “Hard”) cũng có các thuộc tính tương tự.
- Tạo panel có tên là pnlInfor gồm các thuộc tính sau:
    - Controls.Add: phương thức này để thêm các điều khiển vào panel. Trong giao diện này cần thêm 4 điều khiển là lbFlag, txtFlag, lbScore, txtScore.
    - Location: thuộc tính đặt vị trí cho panel ở tọa độ (0, 28).
    - Name: thuộc tính đặt tên cho panel là “pnlInfor”.
    - Padding: thuộc tính đặt khoảng cách giữa các cạnh của panel và các điều khiển bên trong nó là 4 pixel.
    - Size: thuộc tính đặt kích thước của panel có chiều rộng là 308 pixel và chiều cao là 46 pixel.
- Tạo các điều khiển lbFlag (text = “Flag”), txtFlag, lbScore (“text” = Score”), txtScore thêm vào pnlInfor với các thuộc tính:
    - Font chữ MS Reference Sans Serif và kiểu chữ là Bold.
    - Khi chưa bắt đầu chơi, txtFlag và txtScore là 2 ô trống không, trong lúc chơi sẽ hiển thị số cờ và số điểm tương ứng vào 2 ô và văn bản có màu đỏ (red).
    - TextAlign có giá trị là HorizontalAlignment.Center tức văn bản sẽ được căn giữa bên trong TextBox.

- Tạo panel có tên pnlBody là vùng chứa các ô của bảng trò chơi với vị trí trong form là (0, 75) và kích thước ban đầu là (308, 259).

3.1.2 Giao diện sau khi chọn cấp độ chơi

Để bắt đầu chơi, người dùng vào mục Level và chọn cấp độ mình muốn.



Hình 3. 2 Vào mục Level và lựa chọn cấp độ trò chơi

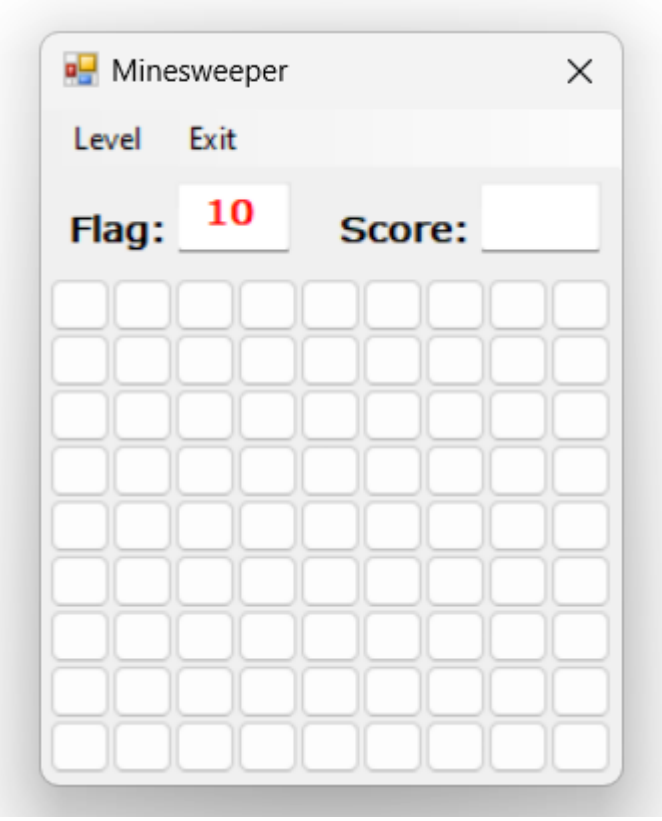
Chương trình có tạo phương thức xử lý cho sự kiện click chuột vào mục Level: **private void levelToolStripMenuItem_Click(object sender, EventArgs e)**. Khi người chơi chọn cấp độ chơi, chương trình sẽ nhận lệnh và kiểm tra trường hợp và vẽ các ô cho bảng trò chơi với kích thước tương ứng với mỗi cấp độ.

❖ **Giao diện trò chơi ở cấp độ Easy 9x9 với 10 mìn:**

Ở cấp độ này, bảng trò chơi được tạo ra với 9 hàng và 9 cột tức tổng cộng 81 button, trong đó có 10 ô chứa mìn.

pnlBody ở cấp độ này cũng thay đổi kích thước là 308 pixel x 259 pixel. Cùng với đó, form cũng được đặt lại kích thức là (232, 270).

Tại txtFlag hiển thị số 10 màu đỏ tức số mìn ở mức độ Easy là 10, đó cũng là số lá cờ mà người chơi có thể cắm lên bảng trò chơi.



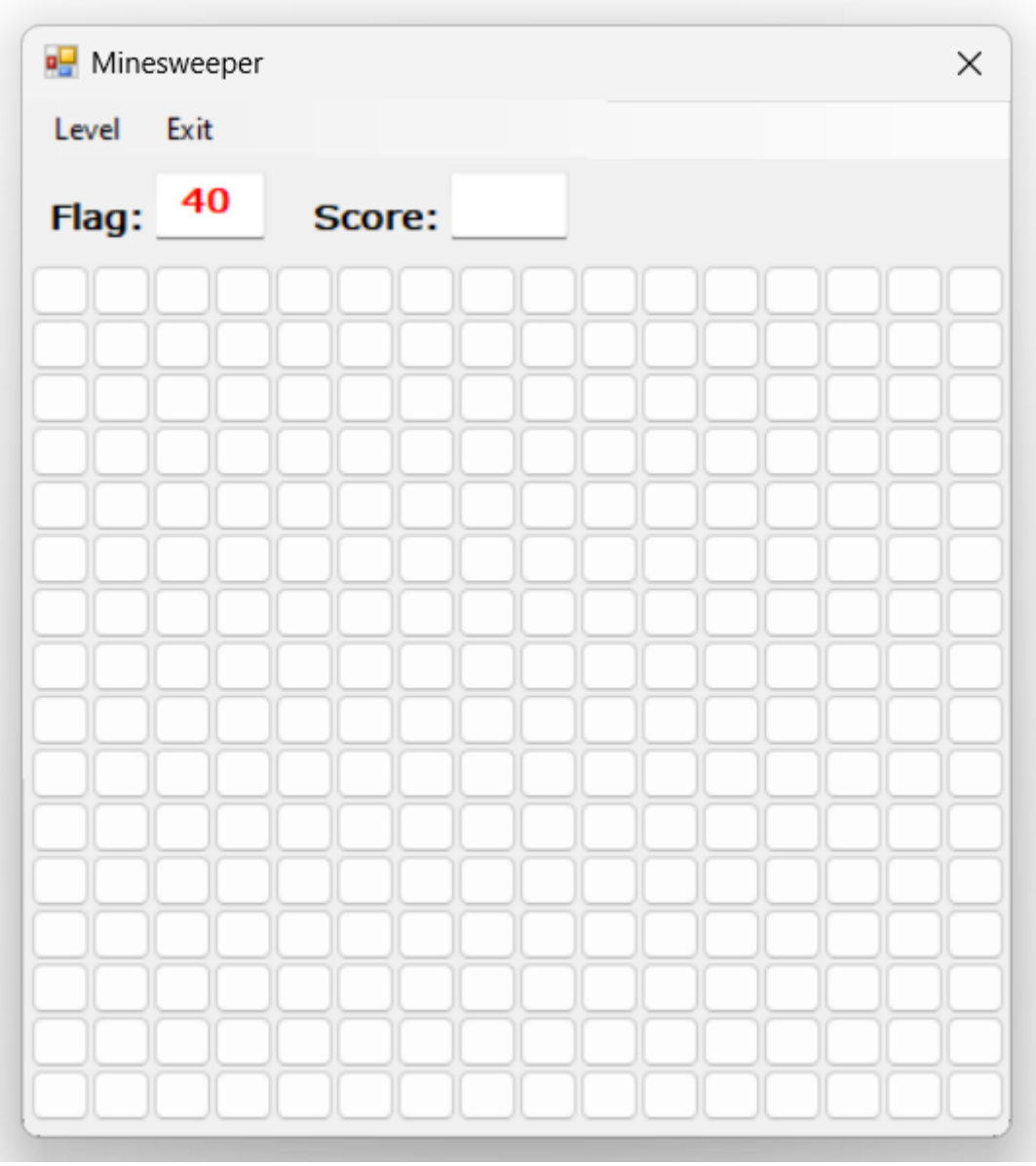
Hình 3. 3 Giao diện trò chơi ở cấp độ Easy

❖ **Giao diện trò chơi ở cấp độ Medium 16x16 với 40 mìn:**

Ở cấp độ này, bảng trò chơi được tạo ra với 16 hàng và 16 cột tức tổng cộng 256 button, trong đó có 40 ô chứa mìn.

pnlBody ở cấp độ này cũng thay đổi kích thước là 544 pixel x 442 pixel. Cùng với đó, form cũng được đặt lại kích thức là (405, 425).

Tại txtFlag hiển thị số 10 màu đỏ tức số mìn ở mức độ Medium là 40, đó cũng là số lá cờ mà người chơi có thể cắm lên bảng trò chơi.



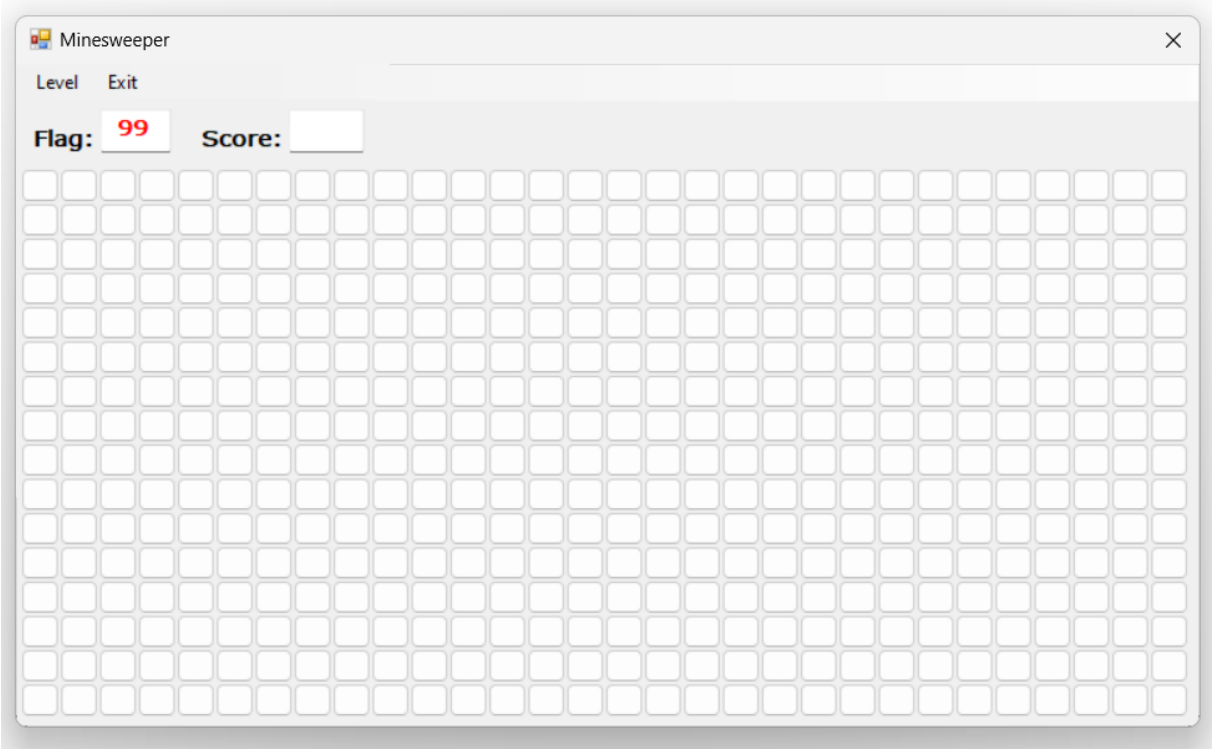
Hình 3. 4 Giao diện trò chơi ở cấp độ Medium

❖ **Giao diện trò chơi ở cấp độ Hard 16x30 với 99 mìn:**

Ở cấp độ này, bảng trò chơi được tạo ra với 16 hàng và 30 cột tức tổng cộng 480 button, trong đó có 99 ô chứa mìn.

pnlBody ở cấp độ này cũng thay đổi kích thước là 1008 pixel x 442 pixel. Cùng với đó, form cũng được đặt lại kích thức là (760, 425).

Tại txtFlag hiển thị số 10 màu đỏ tức số mìn ở mức độ Hard là 99, đó cũng là số lá cờ mà người chơi có thể cắm lên bảng trò chơi.



Hình 3. 5 Giao diện trò chơi ở cấp độ Hard

3.1.3 Bảng trò chơi

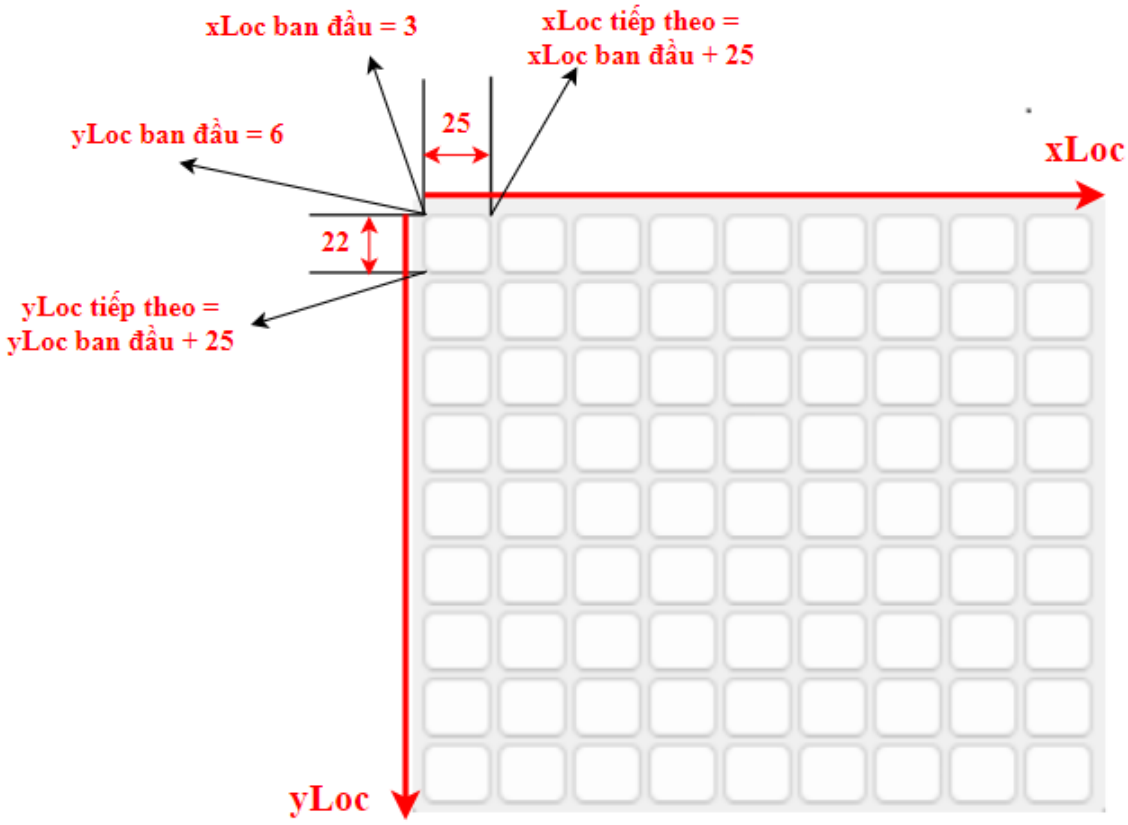
Bảng trò chơi của game Minesweeper là một mảng 2 chiều các nút để người chơi có thể tương tác và mở các nút đó.

Để tạo các nút trong bảng trò chơi, chương trình sử dụng phương thức GenerateButtons tạo các button trong khu vực bảng điều khiển pnlBody.

❖ **Chi tiết phương thức GenerateButtons:**

- 2 tham số truyền vào là row và col (đều có kiểu dữ liệu là int) lần lượt biểu thị cho số hàng và số cột của bảng trò chơi.
- 2 biến xLoc và yLoc (đều có kiểu dữ liệu là int) xác định tọa độ ban đầu của các nút trong pnlBody. Ở đây tọa độ ban đầu để bắt đầu vẽ nút là (3, 6)
- 2 vòng lặp for chạy trong nửa khoảng [0, row) và [0, col) để lặp qua tất cả các hàng và cột của mỗi ô để vẽ nút tại ô đó.
- new Button(): tạo một button mới có tên là btn.

- Thiết lập pnlBody làm cha cho nút mới, tức nút mới tạo sẽ được thêm vào trong pnlBody btn.Parent = pnlBody.
- Thuộc tính Location thiết đặt vị trí nút dựa trên tọa độ xLoc và yLoc.
- Size(): thiết lập kích thước của nút với chiều rộng 25 pixel và chiều cao 22 pixel.
- btn.Tag để gán một chuỗi tag cho nút, chứa thông tin về vị trí của nút trong mảng, sử dụng định dạng "x, y". Vì chúng ta sẽ tạo một ma trận các nút nên tag này sẽ giúp xác định vị trí của nút khi xảy ra sự kiện click.
- Gắn sự kiện Click của nút với phương thức BtnClick **btn.Click += BtnClick;**. Khi người chơi click chuột vào bất kì nút nào thì phương thức BtnClick cũng sẽ được gọi.
- Gắn sự kiện MouseUp của nút với phương thức BtnMouseUp **btn.MouseUp += BtnMouseUp;**. Khi người chơi click chuột phải vào nút bất kì thì phương thức BtnMouseUp cũng sẽ được gọi, nhưng người chơi chỉ thấy sự thay đổi khi click chuột phải vào những nút chưa được mở. Phần này sẽ được trình bày chi tiết hơn ở các phần sau.
- Ở mỗi vòng lặp for trong nửa khoảng (0, col) (tức lặp qua từng vị trí trong 1 hàng, sau khi vẽ xong một nút cần thiết lại tọa độ x để đặt nút tiếp theo. Tọa độ xLoc mới sẽ bằng xLoc ban đầu cộng thêm chiều rộng của nút tức xLoc + 25.
- Lưu trữ nút vừa tạo vào mảng 2 chiều ButtonList tại vị trí [x, y].
- Kết thúc 1 hàng, chúng ta cần tăng tọa độ y cho hàng tiếp theo. Tọa độ yLoc mới sẽ bằng tọa độ yLoc ban đầu cộng thêm chiều cao của nút tức yLoc + 22.
- Đặt lại tọa độ x về giá trị ban đầu là 3 để bắt đầu vẽ hàng tiếp theo.



Hình 3. 6 Minh họa cách tính vị trí các nút

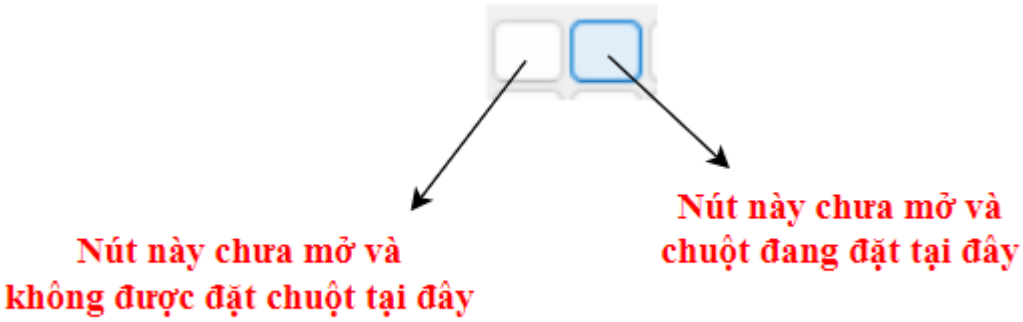
3.1.4 Giao diện các ô trong lúc chơi

Trong lúc chơi game, các nút trong bảng trò chơi sẽ có nhiều trạng thái. Ở mỗi trạng thái, các nút mang giao diện khác nhau để người chơi dễ dàng nhận diện.

❖ ***Giao diện nút khi chưa được mở***

Khi chưa được mở, nút mang giao diện giống như lúc vừa khởi tạo. Kích thước của nút với chiều rộng 25 pixel và chiều cao 22 pixel, có nền trắng và viền màu xám.

Khi di chuyển chuột lên trên nút chưa mở, nút sẽ nổi bật lên với viền màu xanh và nền xanh nhạt. Điều này được tạo ra là vì trong Windows Forms, khi di chuyển chuột lên một button, màu của button thay đổi để cung cấp phản hồi trực quan cho người dùng. Đây được gọi là hiệu ứng hover. Hiệu ứng này là mặc định của Button trong Windows Forms và được quản lý bởi hệ điều hành và giao diện người dùng của .NET Framework.

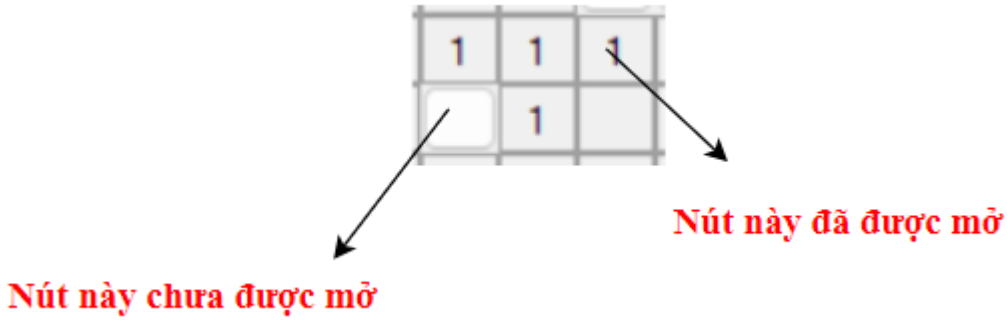


Hình 3. 7 Giao diện nút khi chưa được mở

❖ ***Giao diện nút khi đã được mở***

Khi đã được mở, giao diện của nút sẽ thay đổi. Cụ thể như sau:

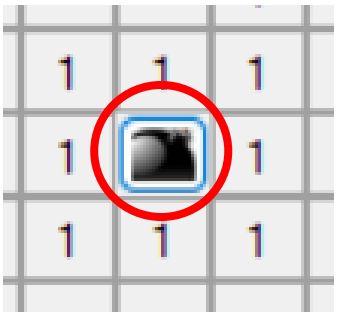
- FlatStyle được thiết đặt lại thành Flat. FlatStyle để xác định kiểu dáng của nút và FlatStyle.Flat làm cho nút có vẻ ngoài phẳng hơn, không có đường viền nổi và không có hiệu ứng bóng đổ.
- FlatAppearance.BorderColor: thiết đặt màu sắc của viền nút khi ở chế độ Flat. Ở đây khi đã mở nút thì viền nhạt và có màu xám tối SystemColors.ControlDark.
- FlatAppearance.BorderSize: thiết đặt kích cỡ đường viền nút khi ở chế độ Flat. Ở đây đặt viền nút là 1 pixel.



Hình 3. 8 Giao diện nút đã được mở

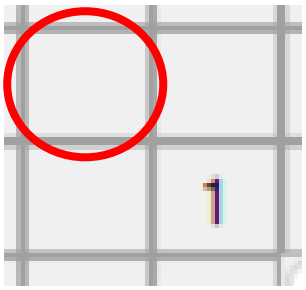
Khi nút được mở, giá trị được gán tại vị trí bên dưới nút sẽ hiện ra. Chúng ta có 3 loại giá trị sẽ xuất hiện khi mở nút:

- Mở nút tại vị trí có giá trị là 10: đây là giá trị gán cho vị trí có mìn. Như vậy khi người chơi click vào nút này thì hiển thị hình quả mìn ngay trên giao diện nút khi được khởi tạo.



Hình 3. 9. Người chơi mở ô chứa mìn

- Mở nút tại vị trí có giá trị là 20: đây là giá trị gán cho vị trí trống. Như vậy khi người chơi click vào nút này thì hiển thị giao diện nút đã mở và không có nội dung bên trong.



Hình 3. 10 Người chơi mở ô trống

- Mở nút tại vị trí có giá trị khác 2 trường hợp trên: đây là vị trí hiển thị số mìn xung quanh nút được mở. Khi người chơi click vào nút này thì hiển thị giao diện nút đã mở và nội dung bên trong là một số (1 – 8).

Hình 3. 11 Người chơi mở ô chứa số

❖ ***Giao diện nút khi cắm cờ***

Khi người chơi phân vân vị trí đó có chứa bom không thì có thể cắm cờ bằng cách nhấn nút phải chuột. Lúc này lá cờ sẽ xuất hiện trên nền giao diện nút vừa tạo. Người chơi có thể nhấn nút phải chuột lên ô có lá cờ để hủy cắm cờ, chương trình sẽ hủy hiển thị nút chứa cờ và quay lại giao diện nút chưa mở ban đầu.

Người chơi sẽ không được đặt cờ ở các vị trí nút đã được mở trước đó.

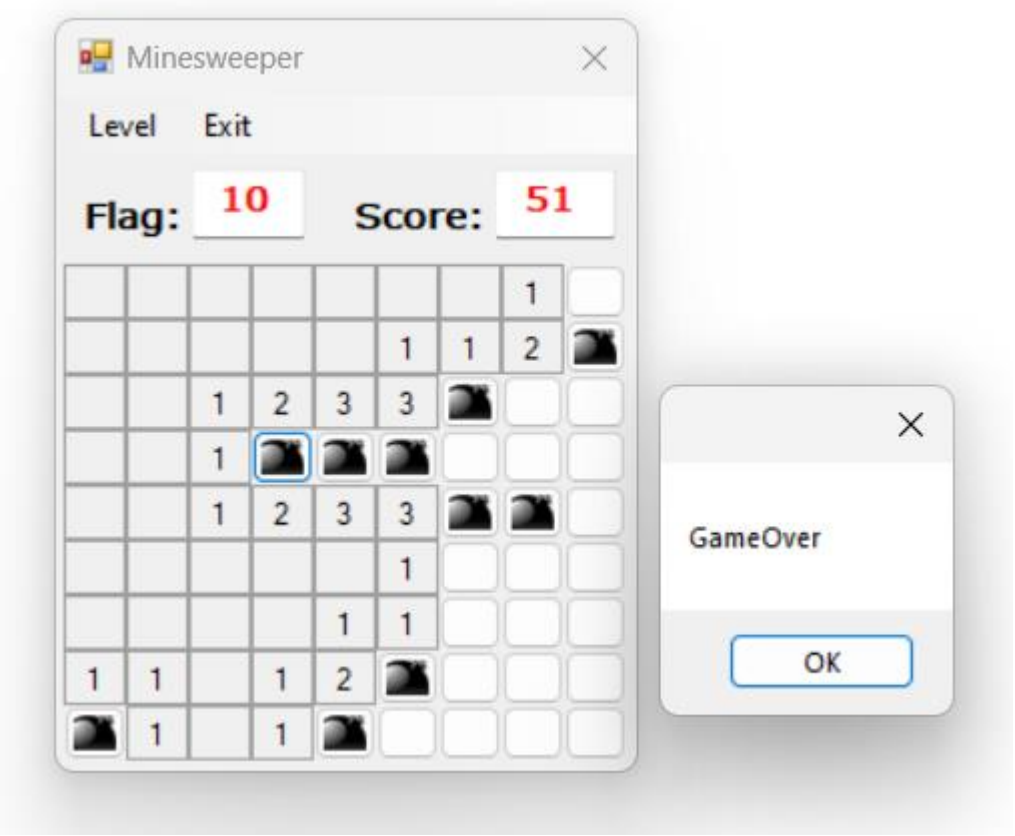
Về phần thuật toán cắm cờ sẽ được trình bày chi tiết ở phần sau.

Hình 3. 12 Giao diện nút được cắm cờ

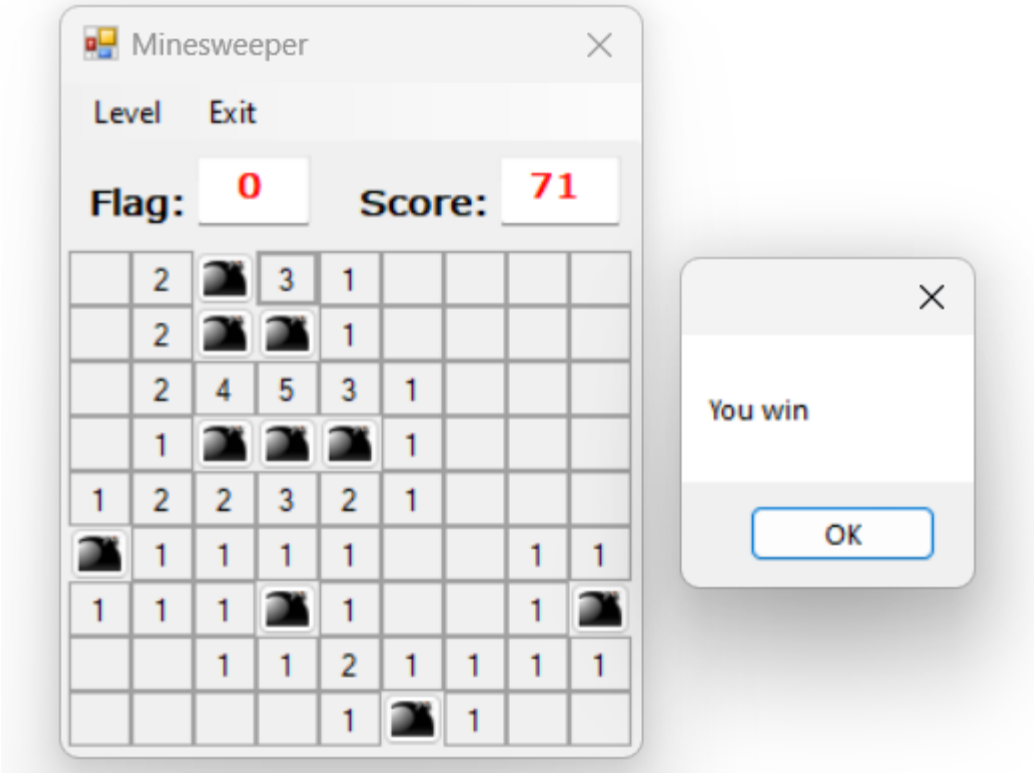
3.1.5 Giao diện kết thúc trò chơi

Khi trò chơi kết thúc, chương trình sẽ mở tất cả các nút chứa mìn hiển thị một MessageBox để thông báo cho người chơi về kết quả của họ sau khi chơi.

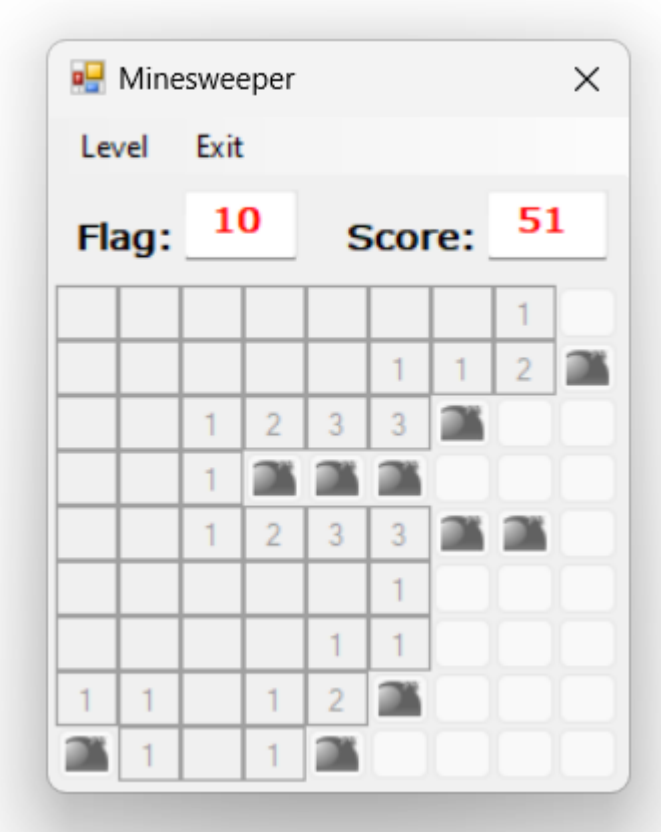
Giao diện khi trò chơi kết thúc bị bao phủ bởi màu xám ở phần pnlBody bởi thuộc tính Enable. Khi thuộc tính Enable có giá trị false, chương trình ngăn chặn sự giao tiếp của người dùng đến các nút của bảng trò chơi, không cho phép người chơi điều khiển pnlBody nữa.



Hình 3. 13 MessageBox hiển thị thông báo người chơi thua



Hình 3. 14 MessageBox hiển thị thông báo người chơi thắng



Hình 3. 15 Chương trình không cho phép người dùng điều khiển pnlBody và các nút

3.2. Thuật toán

3.2.1 Thiết đặt Form

Chương trình xây dựng nhiều chức năng và sự kiện phục vụ cho trò chơi. Đầu tiên, chương trình sẽ khởi tạo 2 mảng 2 chiều là Positions (kiểu dữ liệu byte) để lưu trữ giá trị ở các vị trí trong bảng trò chơi và ButtonList (kiểu dữ liệu Button) để lưu trữ vị trí các nút trong bảng trò chơi.

- Hàm khởi tạo Form trò chơi **public Form1()**.
- **InitializeComponent()**: đây là phương thức tự động sinh ra bởi Visual Studio để khởi tạo và thiết lập các thành phần giao diện của form. Nó sẽ chứa các đoạn mã để tạo và sắp xếp các điều khiển (controls) như button, textbox, label, panel,… Phương thức này được IDE (Intergrated Development Environment) tạo tự động khi thiết kế giao diện bằng cách kéo thả các điều khiển trên form.

- Đăng kí sự kiện Click cho các mục menu: easyToolStripMenuItem (Easy), mediumToolStripMenuItem (Medium), và hardToolStripMenuItem (Hard). Khi các mục này được nhấn, sự kiện levelToolStripMenuItem_Click sẽ được gọi.
    - easyToolStripMenuItem.Click += new EventHandler (levelToolStripMenuItem_Click);
    - mediumToolStripMenuItem.Click += new EventHandler (levelToolStripMenuItem_Click);
    - hardToolStripMenuItem.Click += new EventHandler (levelToolStripMenuItem_Click);

3.2.2 Xử lý lựa chọn cấp độ trò chơi và hiển thị giao diện trò chơi

Sau khi chương trình xử lý các thiết đặt ban đầu cho Form, trên giao diện người dùng, người chơi đã có thể click chuột vào thanh menu để chọn các cấp độ của trò chơi.

- Phương thức **levelToolStripMenuItem_Click** được gọi khi sự kiện Click được kích hoạt
- Lấy đối tượng **ToolStripMenuItem**: **sender** là đối tượng kích hoạt sự kiện, trong game này là một mục trong menu. **ClickedItem** được ép kiểu về **ToolStripMenuItem** để có thể truy cập các thuộc tính và phương thức của đối tượng này.
- Khởi tạo các biến có kiểu dữ liệu int: **row**, **col**, **mine**, **flag** lần lượt biểu thị cho số hàng, số cột, số mìn và số lá cờ.
- Cấu trúc điều khiển if – else if – else if để xác định cấp độ khó của trò chơi và thiết lập kích thước cho bảng điều khiển pnlBody và Form.

Sau khi xử lý lựa chọn cấp độ trò chơi thì chương trình sẽ thiết lập lại trò chơi và hiển thị giao diện trò chơi mới ngay trên cửa sổ trò chơi hiện tại.

- Phương thức Controls.Clear() xóa tất cả các điều khiển con (child controls) khỏi pnlBody, tức tất cả các ô trò chơi (nếu có) sẽ bị xóa. Điều

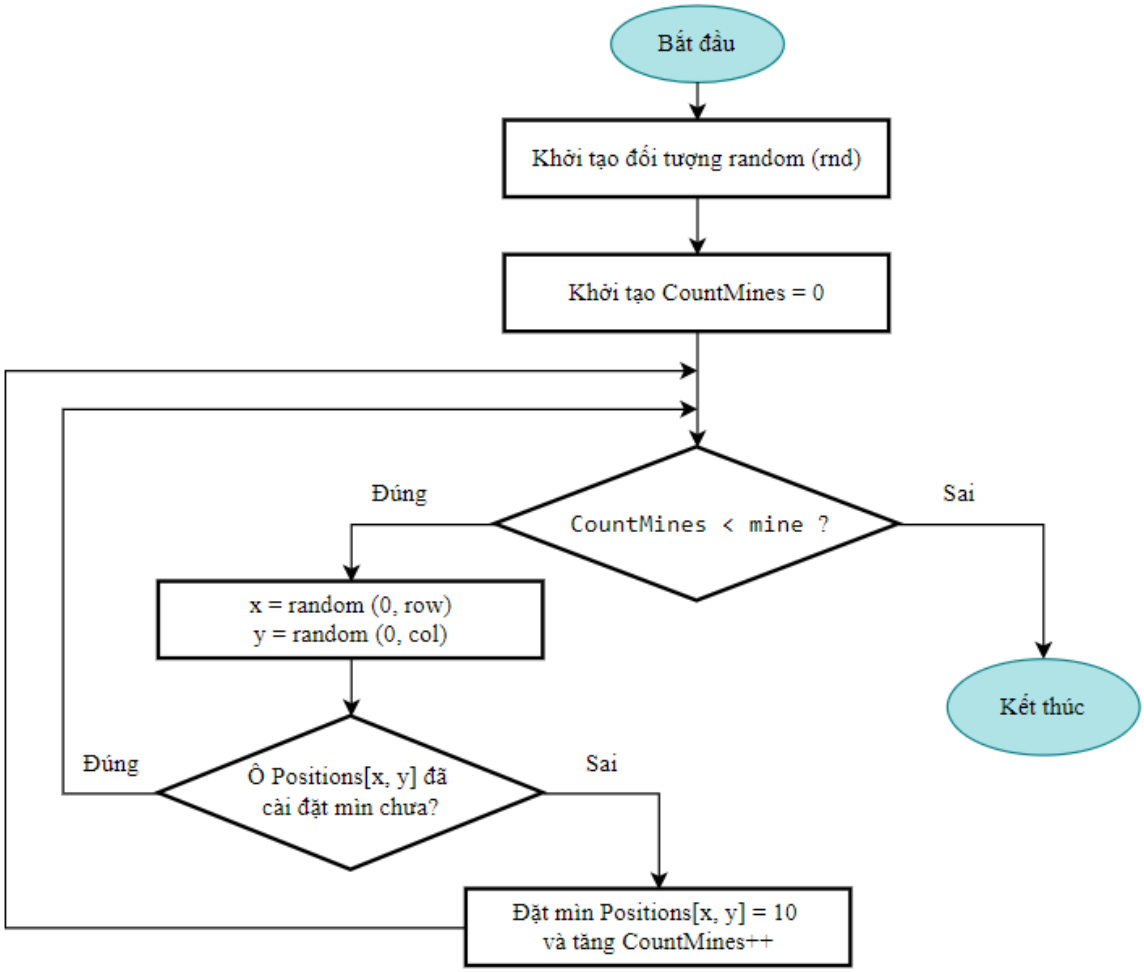
này dễ dàng cho việc tạo ra các bảng trò chơi mới ngay tại cửa sổ trò chơi vừa tạo.

- Phương thức Clear() xóa các nội dung văn bảng trong các textbox txtFlag và txtScore để làm chúng trở lại trạng thái rỗng.
- Cài đặt lại giá trị cho biến points về 0 và biến flag = giá trị mine. Biến points dùng để lưu điểm số người chơi. Khi mới bắt đầu trò chơi, points = 0.
- Cài đặt lại giá trị cho biến flag = giá trị mine. Biến mine đại diện cho số mìn trong trò chơi, biến flag đại diện cho số lượng cờ mà người chơi có thể cắm. Khi mới bắt đầu chơi, flag = số mìn.
- Thiết lập nội dung văn bảng của txtFlag bằng giá trị của mine, chuyển đổi nó thành chuỗi để hiển thị số lượng mìn trong textbox txtFlag.
- Khởi tạo lại mảng 2 chiều Positions với kích thước row x col. Mảng này được sử dụng để lưu tthông tin về vị trí của các mìn và các giá trị khác trên bảng trò chơi.
- Khởi tạo lại mảng 2 chiều ButtonList với kích thước row x col. Mảng này được sử dụng để lưu trữ các nút (buttons) tương ứng với từng ô trong bảng trò chơi.
- Thiết lập thuộc tính Enabled của pnlBody thành true. Điều này cho phép người dùng tương tác với các điều khiển bên trong pnlBody.
- Gọi phương thức GenerateMines để đặt vị trí các quả mìn.
- Gọi phương thức GeneratePositionValue để tạo giá trị cho các ô không chứa mìn.
- Gọi phương thức GenerateButtons để tạo các ô trong bảng trò chơi.

3.2.3 Đặt vị trí các quả mìn

Để đặt mìn vào các vị trí trong bảng trò chơi, chương trình tạo phương thức GenerateMines để đặt ngẫu nhiên số mìn.

- Trước tiên, sử dụng lớp Random để tạo số ngẫu nhiên. Đối tượng rnd sẽ được sử dụng để sinh các vị trí ngẫu nhiên cho mìn.
- Phương thức GenerateMines có 3 tham số là row, col và mine biểu thị lần lượt cho số hàng, số cột và số mìn.
- Tạo biến CountMines kiểu int và gán giá trị bằng 0. Biến này để tính số mìn đã được đặt vào bảng trò chơi.
- Cho vòng lặp while chạy đến khi số mìn đã đặt (CountMines) bằng số mìn cần đặt (mine). Lúc này ta sẽ tạo 2 biến x nhận giá trị ngẫu nhiên thông qua rnd trong khoảng (0, row) và biến y nhận giá trị ngẫu nhiên thông qua rnd trong khoảng (0, col). Mục đích là chọn ngẫu nhiên một ô ở vị trí (x, y).
- Trước khi đặt mìn vào ô đã được chọn ở bước trên, ta cần kiểm tra vị trí đó đã được đặt mìn hay chưa (Positions[x, y] == 0). Nếu vị trí đó đã đặt mìn rồi thì bỏ qua điều kiện này và tiếp tục vòng lặp while. Nếu vị trí đó chưa được đặt mìn thì gán giá trị cho vị trí đó bằng 10 (mặc định ô có giá trị 10 chứa mìn) rồi tăng CountMines lên 1 đơn vị.
- Tiếp tục vòng lặp cho đến khi đặt đủ số mìn cần đặt.



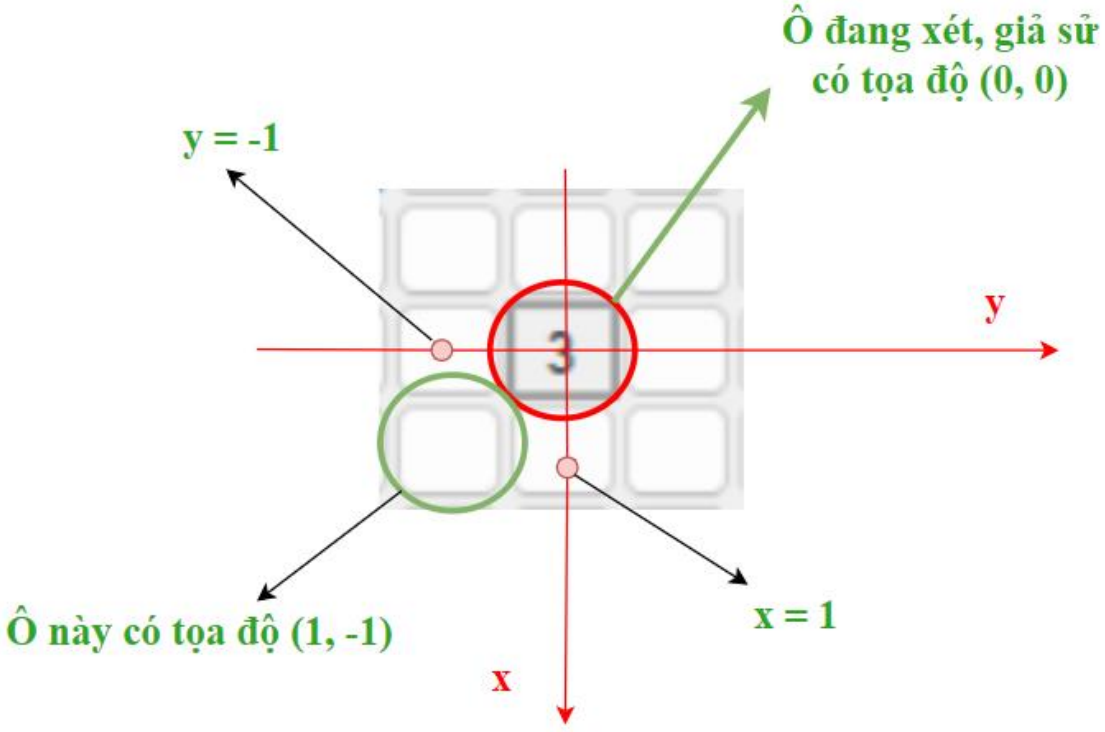
Hình 3. 16 Lưu đồ thuật toán GenerateMines

3.2.4 Tạo giá trị cho các ô không chứa mìn

Sau khi tạo xong ô chứa mìn, chúng ta phải tạo giá trị cho các ô không chứa mìn bằng phương thức GeneratePositionValue. Phương thức này sẽ đếm số mìn xung quanh một ô để gán giá trị cho ô đó.

- Phương thức GeneratePositionValue có 2 tham số được truyền vào là row và col biểu thị lần lượt cho số hàng và số cột của bảng trò chơi.
- Sử dụng 2 vòng lặp for lồng nhau để duyệt lần lượt từng ô trong bảng trò chơi.
- Ở mỗi ô, sử dụng điều kiện if để kiểm tra ô hiện tại có chứa mìn không (chứa giá trị 10), nếu có thì bỏ qua và tiếp tục vòng lặp, vì nó là ô chứa mìn đã được gán giá trị là 10 thì không cần phải xử lý nữa.

- Tính toán số lượng mìn xung quanh mỗi ô:
    - Khởi tạo biến mineCounts kiểu byte và gán giá trị ban đầu là 0, biến này dùng để đếm số lượng mìn xung quanh ô hiện tại.
    - Sử dụng 2 vòng lặp for lồng nhau để duyệt qua các ô xung quanh ô hiện tại.
    - Dùng 2 biến checkerX = x + counterX và checkerY = y + counterY để xác định tọa độ của các ô xung quanh. Trong đó counterX và counterY lần lượt nhận các giá trị -1, 0, 1. counterX và counterY nhận 3 giá trị này là vì nếu lấy vị trí ô đang xét là trung tâm (0, 0) thì những ô bên trái sẽ ở vị trí (checkerX, -1), những ô bên phải sẽ ở vị trí (checkerX, 1), những ô bên trên sẽ ở vị trí (-1, checkerY), những ô bên dưới sẽ ở vị trí (1, checkerY).



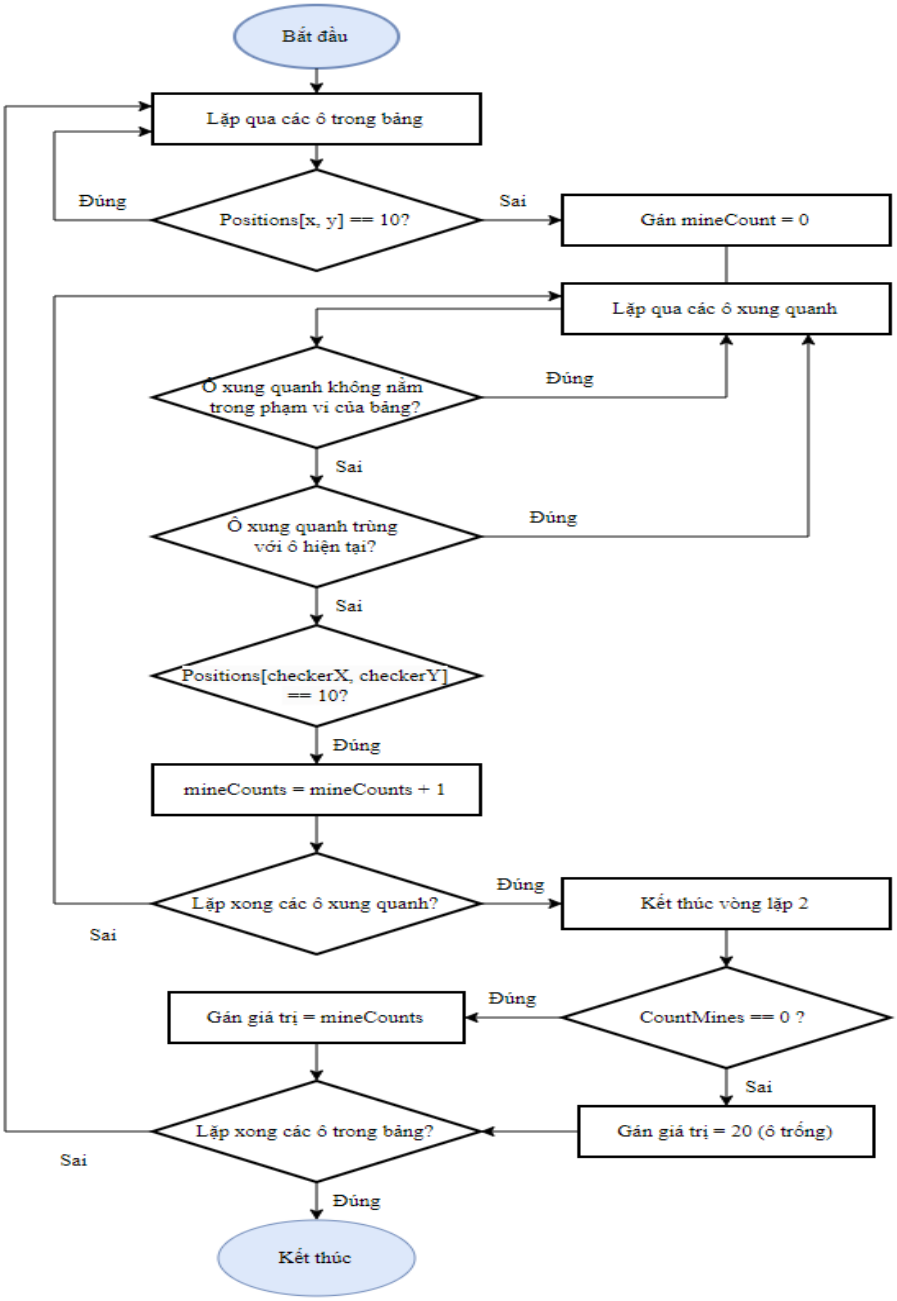
Hình 3. 17 Minh họa tọa độ các ô xung quanh ô đang xét

- Kiểm tra ô xung quanh có nằm ngoài giới hạn của bảng trò chơi không (checkerX == -1 hoặc checkerY == -1 hoặc checkerX > row - 1 hoặc checkerY > col – 1), nếu có thì bỏ qua.

- Kiểm tra ô xung quanh có bị trùng tọa độ với ô hiện tại không (checkerX == x và checkerY == y) (vì lặp qua từng ô trong 1 mảng 9 vị trí thì vị trí trung tâm sẽ trùng với ô đang xét), nếu có thì bỏ qua.
- Kiểm tra ô xung quanh có chứa mìn không (Positions[checkerX, checkerY] == 10), nếu có thì tăng biến mineCounts lên 1 đơn vị.

- Gán giá trị cho ô hiện tại:
  - Sau khi tính toán số mìn xung quanh thì gán giá trị cho ô hiện tại.
  - Kiểm tra có mìn xung quanh không (mineCounts == 0), nếu không có mìn xung quanh thì gán giá trị bằng 20 (tức ô trống).

  - Nếu có mìn xung quanh thì gán giá trị bằng mineCounts (tức bằng số lượng mìn xung quanh).



Hình 3. 18 Lưu đồ thuật toán GeneratePositionValue

3.2.5 Tạo các nút trong bảng trò chơi

Để chơi được thì không thể thiếu các nút trò chơi. Để vẽ ma trận 2 chiều các nút trò chơi, chương trình sử dụng phương thức GanerateButton.

- 2 tham số truyền vào là row và col (đều có kiểu dữ liệu là int) lần lượt biểu thị cho số hàng và số cột của bảng trò chơi.

- 2 biến xLoc và yLoc (đều có kiểu dữ liệu là int) xác định tọa độ ban đầu của các nút trong pnlBody. Ở đây tọa độ ban đầu để bắt đầu vẽ nút là (3, 6)
- 2 vòng lặp for chạy trong nửa khoảng [0, row) và [0, col) để lặp qua tất cả các hàng và cột của mỗi ô để vẽ nút tại ô đó.
- new Button(): tạo một button mới có tên là btn.
- Thiết lập pnlBody làm cha cho nút mới, tức nút mới tạo sẽ được thêm vào trong pnlBody btn.Parent = pnlBody.
- Thuộc tính Location thiết đặt vị trí nút dựa trên tọa độ xLoc và yLoc.
- Size(): thiết lập kích thước của nút với chiều rộng 25 pixel và chiều cao 22 pixel.
- btn.Tag để gán một chuỗi tag cho nút, chứa thông tin về vị trí của nút trong mảng, sử dụng định dạng "x, y". Vì chúng ta sẽ tạo một ma trận các nút nên tag này sẽ giúp xác định vị trí của nút khi xảy ra sự kiện click.
- Gắn sự kiện Click của nút với phương thức BtnClick **btn.Click += BtnClick;**. Khi người chơi click chuột vào bất kì nút nào thì phương thức BtnClick cũng sẽ được gọi.
- Gắn sự kiện MouseUp của nút với phương thức BtnMouseUp **btn.MouseUp += BtnMouseUp;**. Khi người chơi click chuột phải vào nút bất kì thì phương thức BtnMouseUp cũng sẽ được gọi, nhưng người chơi chỉ thấy sự thay đổi khi click chuột phải vào những nút chưa được mở. Phần này sẽ được trình bày chi tiết hơn ở các phần sau.
- Ở mỗi vòng lặp for trong nửa khoảng (0, col) (tức lặp qua từng vị trí trong 1 hàng, sau khi vẽ xong một nút cần thiết lại tọa độ x để đặt nút tiếp theo. Tọa độ xLoc mới sẽ bằng xLoc ban đầu cộng thêm chiều rộng của nút tức xLoc + 25.
- Lưu trữ nút vừa tạo vào mảng 2 chiều ButtonList tại vị trí [x, y].

- Kết thúc 1 hàng, chúng ta cần tăng tọa độ y cho hàng tiếp theo. Tọa độ yLoc mới sẽ bằng tọa độ yLoc ban đầu cộng thêm chiều cao của nút tức yLoc + 22.
- Đặt lại tọa độ x về giá trị ban đầu là 3 để bắt đầu vẽ hàng tiếp theo.

3.2.6 Sự kiện click chuột trái

Phương thức BtnClick được sử dụng để xử lý sự kiện khi người chơi nhấp chuột vào một nút trong bảng trò chơi.

- Xác định ô được click vào:
    - Lấy đối tượng btn dạng Button được nhấp chuột từ đối tượng gửi sự kiện sender.
    - Lấy tọa độ x và y từ thuộc tính Tag của button, trong đó sử dụng phương thức Split để tách chuỗi thành các giá trị riêng biệt. Như vậy x sẽ xác định hàng và y xác định cột của button được click chuột.
    - Tạo biến value để lấy giá trị tại vị trí (x, y) của mảng Positions, tức là ta đã xác định được giá trị của ô mà ta click chuột vào.
- Kiểm tra giá trị value được lấy ra:
    - Nếu value == 10 (tức nút bấm vào chứa mìn) thì hiển thị hình ảnh quả mìn trên nút bằng cách gán btn.Image là ảnh quả mìn. Khi đó người chơi đã thua, chương trình sẽ hiển thị tất cả vị trí chứa mìn có trong bảng trò chơi bằng cách gọi phương thức ShowAllMines (phương thức này sẽ được trình bày rõ ở phần sau). Đồng thời một MessageBox hiện thông báo “GameOver” qua phương thức Show. Sau đó chương trình sẽ vô hiệu hóa pnlBody, không cho phép người dùng tương tác với các nút trong bảng trò chơi nữa.
    - Nếu value == 20 (tức nút bấm vào là ô trống), hệ thống sẽ hiển thị giao diện ô trống (đặc điểm giao diện đã được trình bày ở phần 3.1.4). Chương trình sẽ không cho phép người dùng tương tác với

ô này nữa qua thuộc tính Enabled = false. Khi người chơi mở ô trống thì hệ thống sẽ tự động mở các ô trống lân cận cho đến khi gặp được ô số qua phương thức OpenAdjacentEmptyTile. Chương trình cũng sẽ tăng điểm lên 1 đơn vị mỗi khi 1 ô được mở points++.

- Nếu nút bấm vào là ô số, hệ thống cũng sẽ hiển thị giao diện ô đã mở (đặc điểm giao diện đã được trình bày ở phần 3.1.4). Đồng thời số lượng mìn xung quanh sẽ được hiển thị trên nút qua thuộc tính Text, dùng ToString() để chuyển số sang chuỗi mới hiển thị được btn.Text = Positions[x, y].ToString(). Chương trình cũng sẽ tăng điểm số lên points++.
- Chương trình gỡ bỏ sự kiện Click ra khỏi nút hiện tại để không bị tính điểm lại những ô đã mở btn.Click -= BtnClick.

- Cập nhật điểm số vào ô txtScore bằng với số điểm points hiện tại txtScore.Text = points.ToString().
- Kiểm tra điều kiện thắng:
    - Ở sự kiện Click chuột vào nút này, chúng ta không truyền tham số mine (tổng số mìn) ban đầu, vì thế chúng ta cần một biến mineCount để đếm số mìn có trong bảng trò chơi. Tạo biến mineCount có giá trị ban đầu là 0. Lặp qua từng giá trị trong mảng Positions, nếu gặp giá trị bằng 10 (tức ô chứa mìn) thì tăng mineCount lên 1 đơn vị.
    - Kiểm tra số điểm points có bằng với số điểm tối đa cho mỗi cấp độ chơi hay chưa (số điểm tối đa = Positions.GetLength(0) * Positions.GetLength(1) – mineCount), nếu số điểm đã đạt tối đa tức người chơi đã giành chiến thắng thì chương trình sẽ tự động mở tất cả các ô chứa mìn bằng phương thức ShowAllMines. Đồng thời một MessageBox hiện lên thông báo “You win”. Chương trình cũng sẽ ngăn chặn sự giao tiếp của người chơi vào các nút trong bảng trò chơi bằng thuộc tính Enabled = false.

3.2.7 Hiển thị các ô lân cận ô trống

Khi người chơi click vào một ô trống, chương trình sẽ tự động mở các ô trống lân cận ô đó cho đến khi gặp ô số. Để giải quyết điều đó, chương trình đã chuẩn bị phương thức OpenAdjacentEmptyTile. Phương thức này được gọi trong sự kiện click khi người chơi mở trúng ô trống.

- Lấy tọa độ x và y từ thuộc tính Tag của button, trong đó sử dụng phương thức Split để tách chuỗi thành các giá trị riêng biệt. Như vậy x sẽ xác định hàng và y xác định cột của button được click chuột.
- Khởi tạo danh sách các nút trông có tên là emptyButtons: List<Button> emptyButtons = new List<Button>();
- Duyệt qua các ô xung quanh: Tương tự như khi tính toán số lượng mìn xung quanh mỗi ô, dùng 2 biến checkerX = x + counterX và checkerY = y + counterY để xác định tọa độ của các ô xung quanh. Trong đó counterX và counterY lần lượt nhận các giá trị -1, 0, 1. Phương thức OpenAdjacentEmptyTile không truyền biến row và col vào nên ta thay bằng row = Positions.GetLength(0) và col = Positions.GetLength(1). Phương thức GetLength giúp trả về chiều dài của list.
- Kiểm tra ô liền kề có nằm ngoài giới hạn của bảng trò chơi hoặc trùng với ô hiện tại hay không. Nếu có thì bỏ qua.
- Xử lý ô trống liền kề:
    - Tạo nút btnAdjacent = ButtonList[checkerX, checkerY] để chuẩn bị cho bước xử lý.
    - Tạo biến value để lấy giá trị tại vị trí (checkerX, checkerY) của mảng Positions.
    - Kiểm tra ô lân cận này có giá trị bằng 20 không (tức là ô trống). Nếu phải thì chương trình tự động mở ô đó và thay đổi giao diện nút đã mở.

- Sau đó vô hiệu hóa nút btnAdjacent.Enabled = false và thêm nút vào danh sách các nút trống emptyButtons.Add(btnAdjacent).
- Tăng điểm points++.

- Xử lý ô không chứa mìn: khi người dùng click trúng ô trống thì các ô trống lân cận sẽ mở ra cho đến khi gặp ô số. Vậy ô số này cũng sẽ bị mở bằng cách gọi phương thức PerformClick để kích hoạt sự kiện Click của nút.
- Xử lý ô mà người chơi cắm cờ nhưng bị chương trình mở ra:
  - Người chơi có thể nghi ngờ một ô bất kì chứa mìn thì cắm cờ vào ô này. Nhưng có thể nghi ngờ đó là không đúng. Khi mở một mảng các ô trống có thể chương trình sẽ tự động mở ô đã cắm cờ đó luôn.
  - Kiểm tra nếu ô đó đã được mở (tức đã bị vô hiệu hóa btn.Enabled == false) và ô đó đã được gắn ảnh (tức ảnh cờ) thì xóa ảnh ở ô đó và chỉ hiển thị giá trị của ô đó.
- Đệ quy OpenAdjacentEmptyTile để mở các ô trống liền kề.
- Cập nhật điểm số vào txtScore.

3.2.8 Hiển thị tất cả vị trí của mìn

Chương trình sẽ hiển thị tất cả các vị trí của mìn khi trò chơi kết thúc. Phương thức ShowAllMines sẽ làm điều này và sẽ được gọi ở các trường hợp trong sự kiện Click chuột.

Chương trình sẽ dùng 2 vòng lặp for để lặp qua vị trí tất cả các nút trong bảng trò chơi và ở mỗi nút sẽ kiểm tra nút đó có chứa mìn hay không (Positions[x, y] == 10), nếu đúng thì hiển thị ảnh mìn tại ô đó.

3.2.9 Sự kiện click chuột được nhả ra trên một button và xử lý chức năng cắm cờ

Sự kiện click chuột phải BtnMouseUp được gọi khi người chơi click chuột phải vào một nút trong bảng trò chơi. Sự kiện này giúp cắm cờ hoặc loại bỏ cờ ở một nút chưa mở.

- Tạo biến flag bằng giá trị tại txtFlag để lưu số cờ hiện tại.
- Kiểm tra xem nút chuột nào được nhả ra. Chỉ thực hiện các thao tác tiếp theo nếu đó là nút chuột phải.
- Lấy đối tượng Button đã kích hoạt sự kiện này và gán nó cho biến btn.
- Kiểm tra nếu nút đó không chứa hình ảnh và văn bản và số lá cờ hiện tại lớn hơn không thì mới gán hình ảnh cờ vào nút. Sau đó loại bỏ sự kiện Click chuột khỏi nút này để ngăn người chơi click vào nút này sau khi đã cắm cờ. Chỉ khi bỏ cắm cờ thì người chơi mới được click chuột trái vào nút đó. Sau khi đã cắm thì giảm số cờ hiện có đi 1 đơn vị.
- Nếu nút đã có hình ảnh (tức đã cắm cờ) thì khi click chuột phải vào nút sẽ xóa hình ảnh cờ đi và quay lại nút chưa mở ban đầu. Đồng thời thêm lại sự kiện Click cho nút để cho phép người chơi nhấp vào nút này lại. Sau đó tăng số cờ hiện có lên 1 đơn vị.
- Cập nhật lại giá trị txtFlag bằng số cờ hiện có txtFlag.Text = flag.ToString();.

3.2.10 Thoát khỏi giao diện trò chơi

Để thoát khỏi giao diện trò chơi, người chơi thực hiện click vào Exit trên thanh menu. Khi đó sự kiện exitToolStripMenuItem_Click được gọi. Chương trình sẽ thực hiện đóng form trò chơi.

CHƯƠNG 4: HIỆN THỰC CHƯƠNG TRÌNH

Game Minesweeper chủ yếu sẽ có 3 giao diện chính :

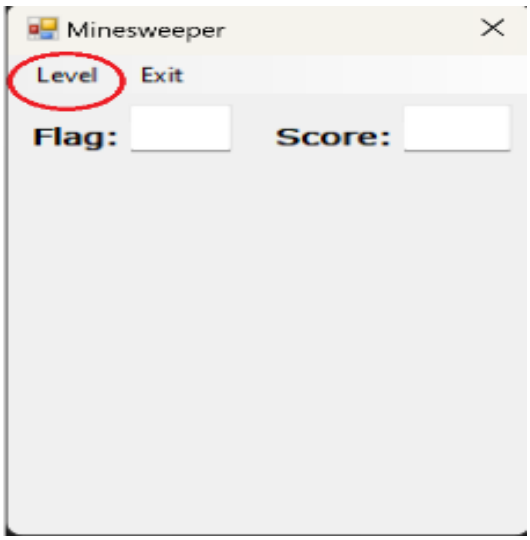
- Giao diện mở đầu.
- Giao diện người chơi.
- Giao diện trò chơi.

Ở từng giao diện sẽ có những phím tắt và những chức năng riêng biệt. Ở chương này sẽ đi sâu vào tìm hiểu về các giao diện cũng như cách chơi của game.

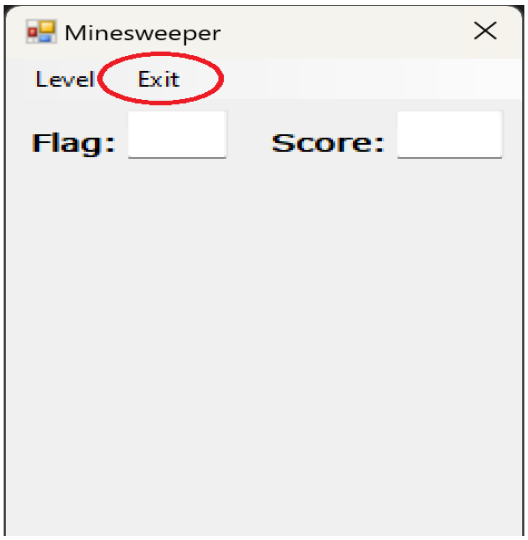
4.1. Giao diện mở đầu

Sau khi chạy, trò chơi sẽ xuất hiện một khung hình chữ nhật nằm dọc trên màn hình, đó là giao diện mở đầu của trò chơi. Giao diện này chứa các thông tin như:

- Menu Strip gồm ô Level có người chơi lựa chọn cấp độ chơi và ô Exit để thoát game.
- Ô Flag hiển thị số bom tương ứng với số bom được mặc định trong từng Level, ô Score hiển thị số điểm mà người chơi đạt được qua mỗi lượt chơi.



Hình 4. 1 Ô Level trong Memu Strip

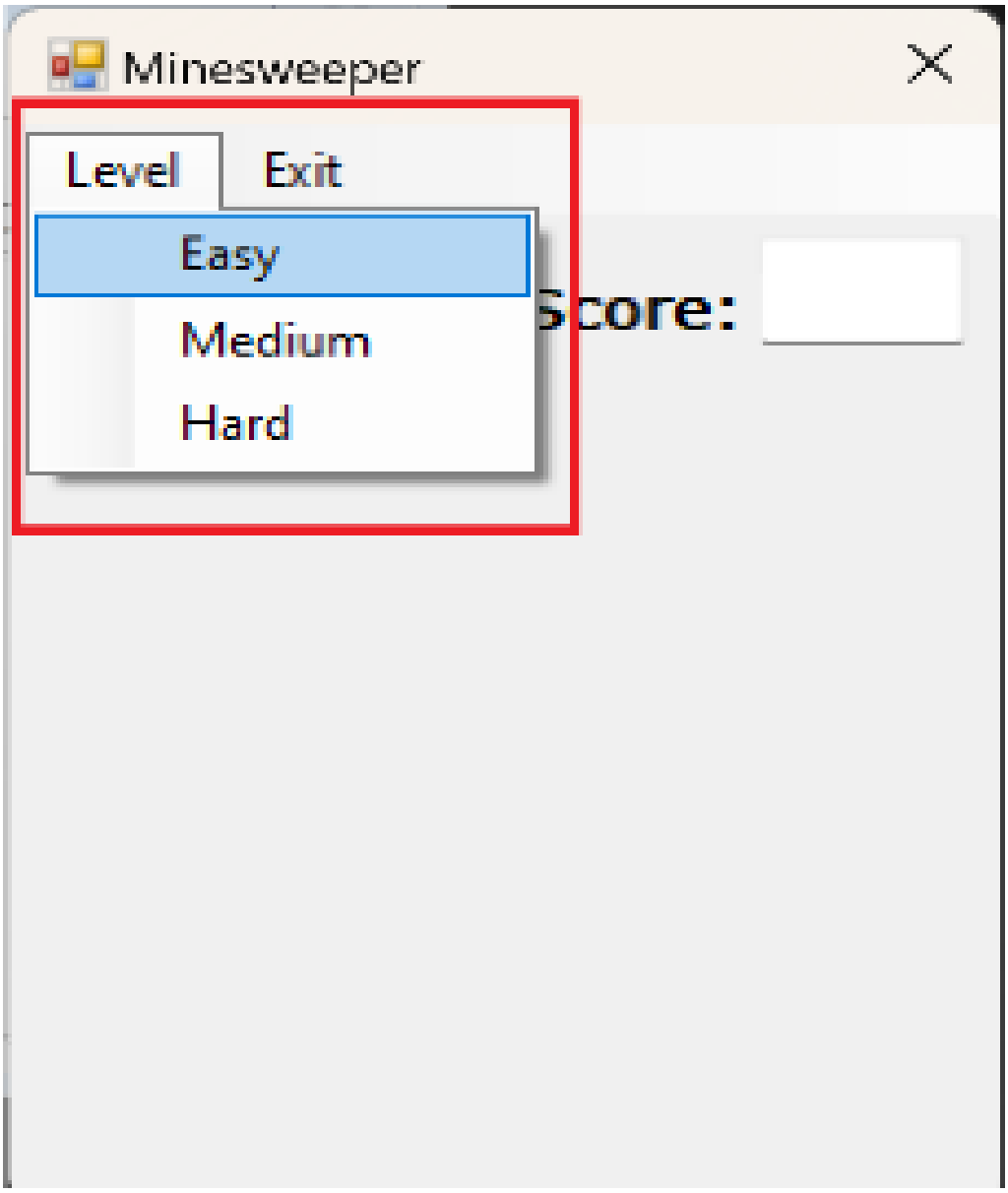


Hình 4. 2 Ô Exit trong Menu Strip

4.2. Giao diện người chơi

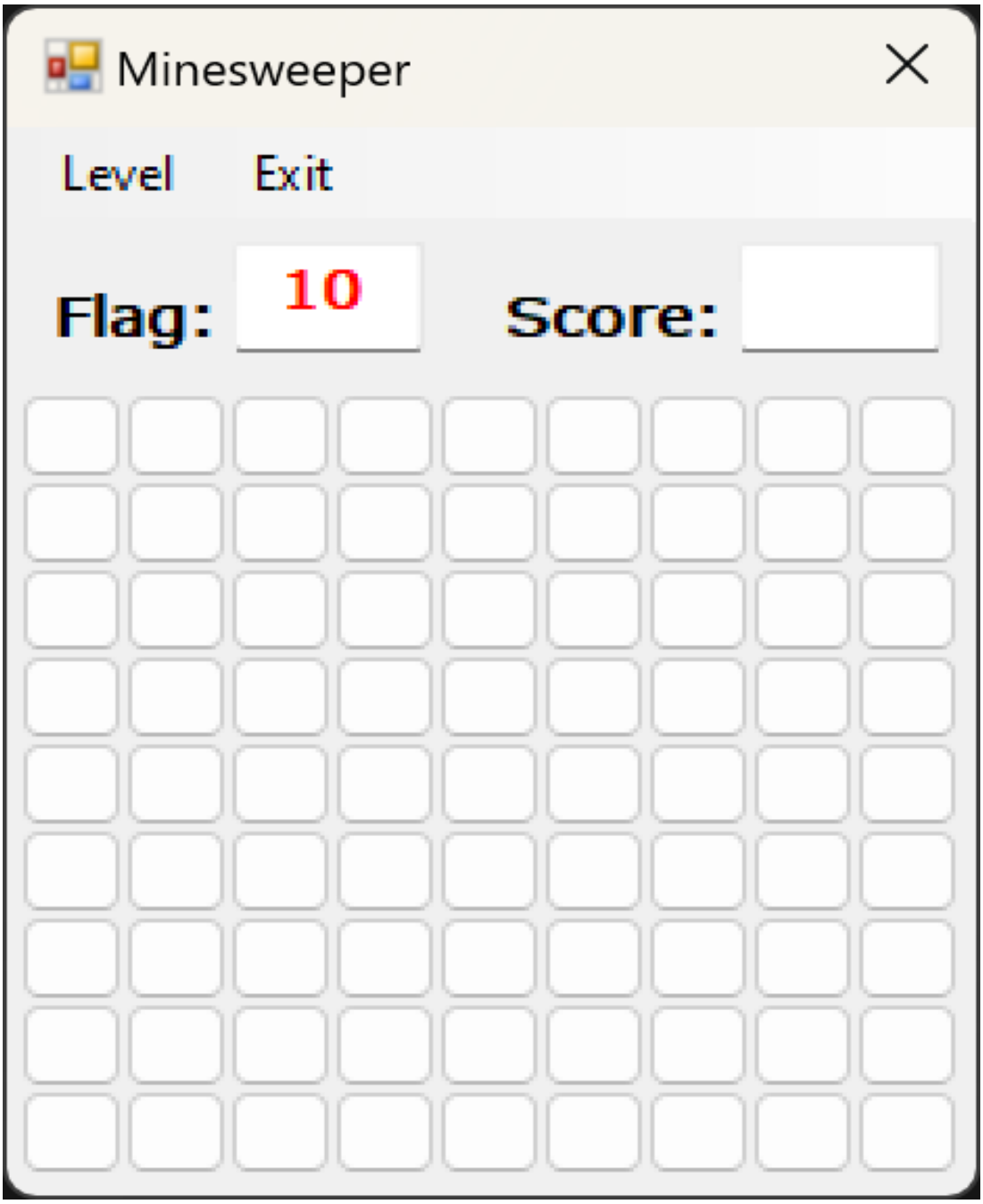
Giao diện này cho phép người chơi chọn cấp độ chơi trong ô Level.

Đối với ô Level: khi người chơi bấm chọn Level sẽ xuất hiện 3 cấp độ chơi cho người chơi lựa chọn là “Easy, Medium, Hard”.

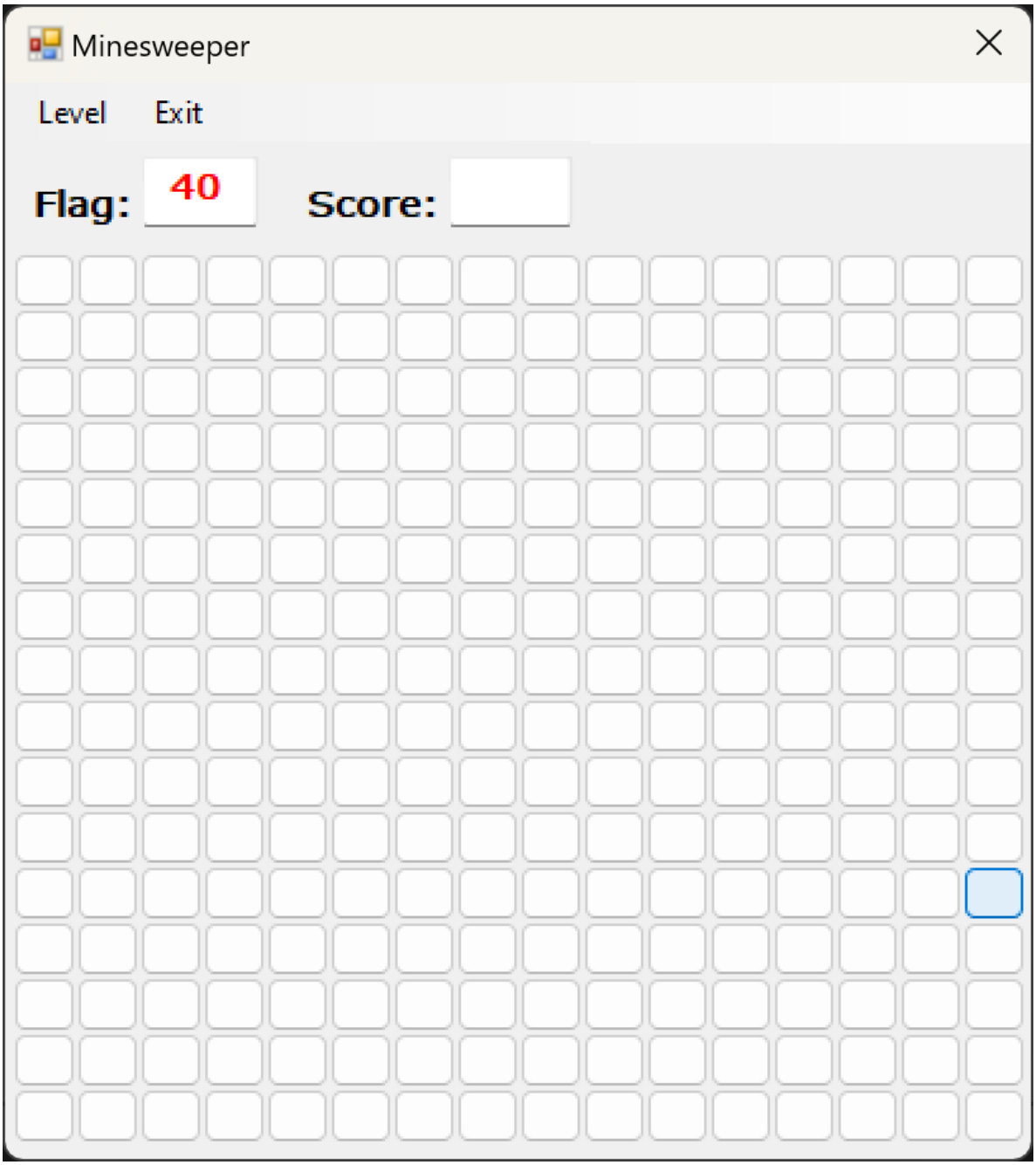


Hình 4. 3 Ba cấp độ chơi để người chơi lựa chọn

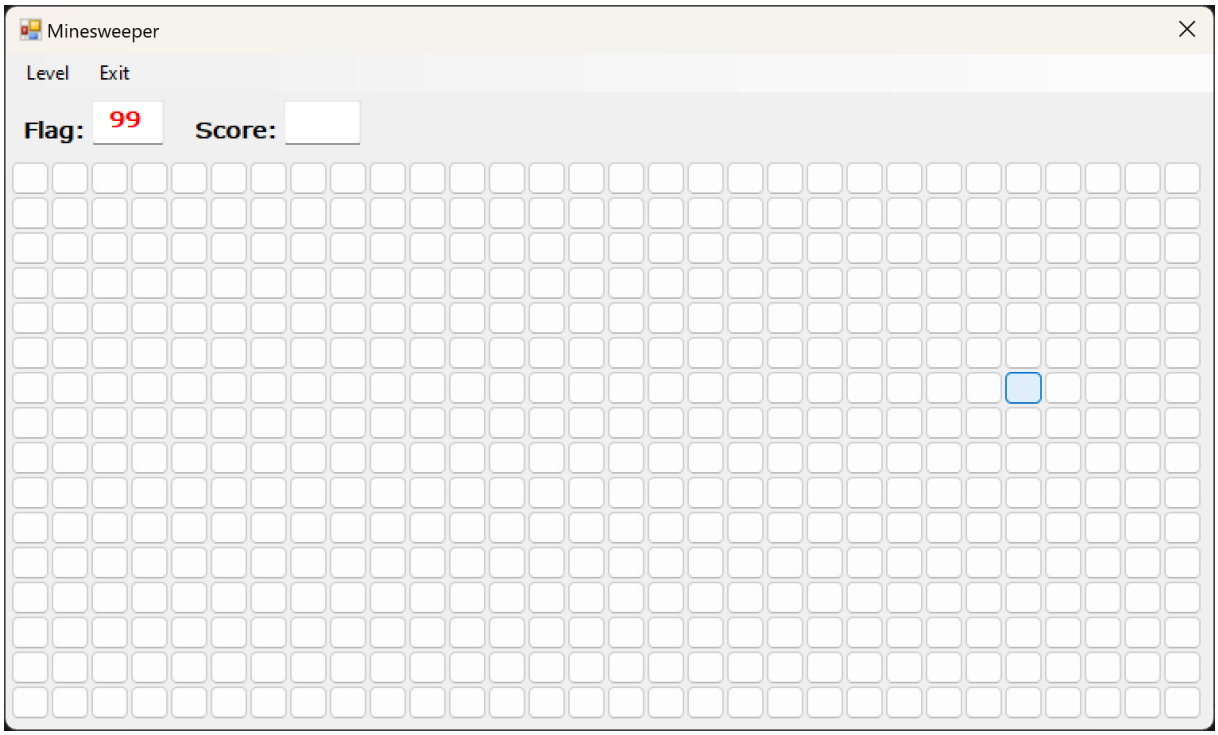
Khi người chơi đã bấm chọn Level muốn chơi, bảng trò chơi sẽ xuất hiện với số ô tương ứng trong từng Level và thông tin số bom của Level được hiển thị trong ô Flag.



Hình 4. 4 Bảng trò chơi khi người chơi chọn cấp độ Easy



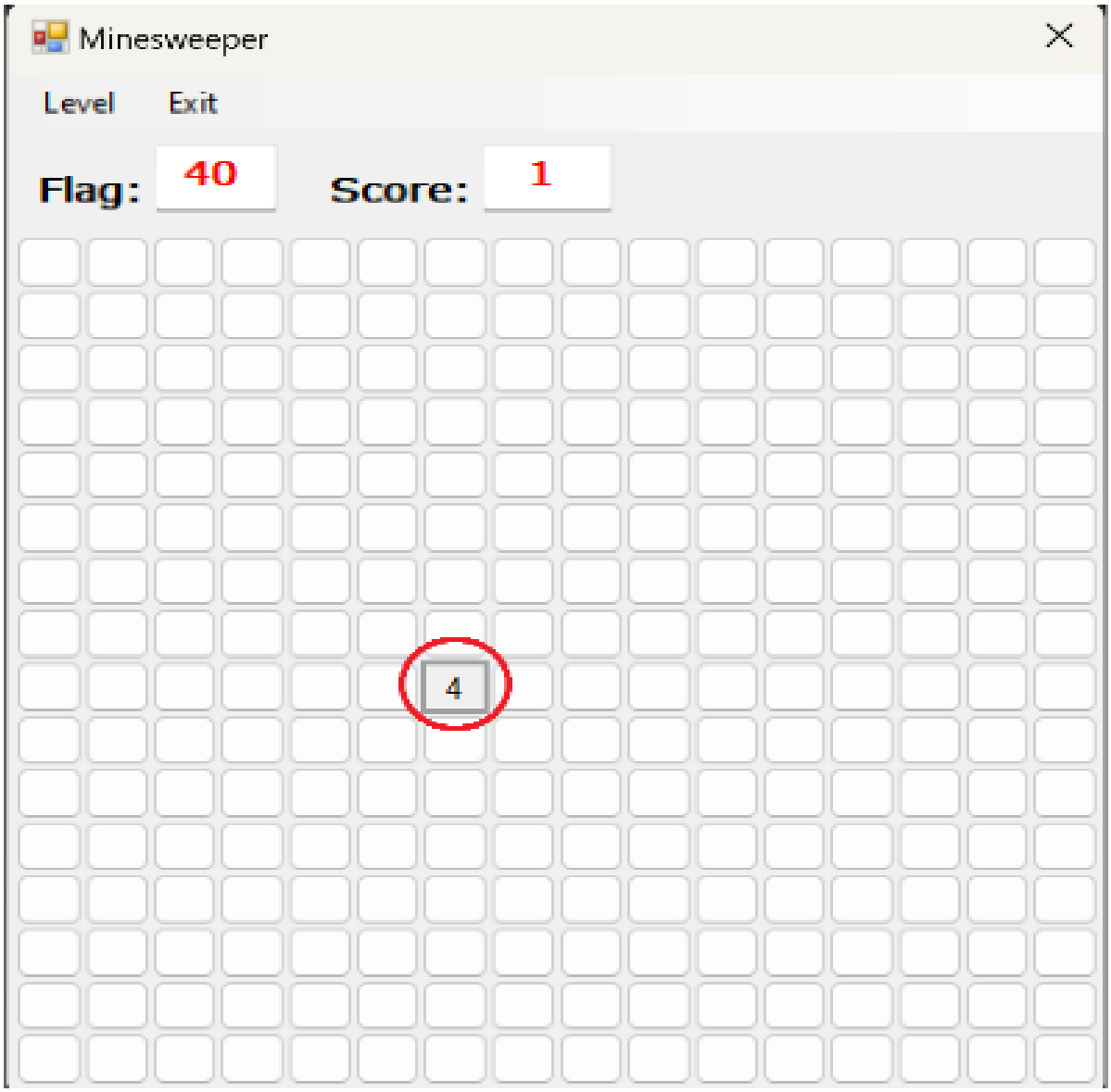
Hình 4. 5 Bảng trò chơi khi người chơi chọn cấp độ Medium



Hình 4. 6 Bảng trò chơi khi người chơi chọn cấp độ Hard

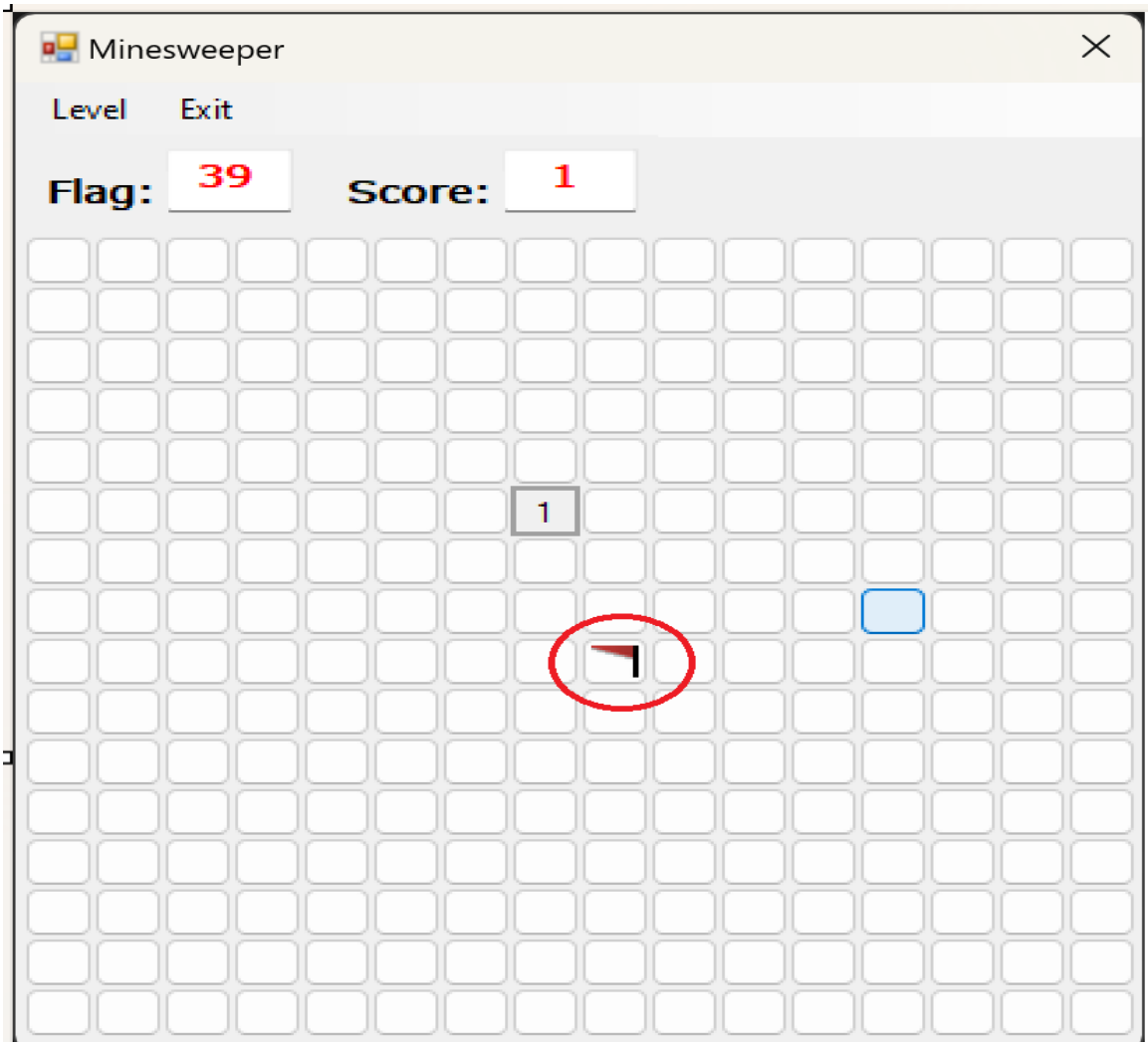
4.3. Giao diện trò chơi

Sau khi bảng trò chơi đã mở ra, người chơi có thể bắt đầu bằng cách nhấn vào bất kỳ ô nào trên bảng trò chơi. Nếu ô đó không chứa mìn, nó sẽ hiển thị một con số. Con số này cho biết có bao nhiêu quả mìn nằm ở các ô kề bên (cả ô nằm chéo).



Hình 4. 7 Người chơi nhấn vào 1 ô trên bảng trò chơi

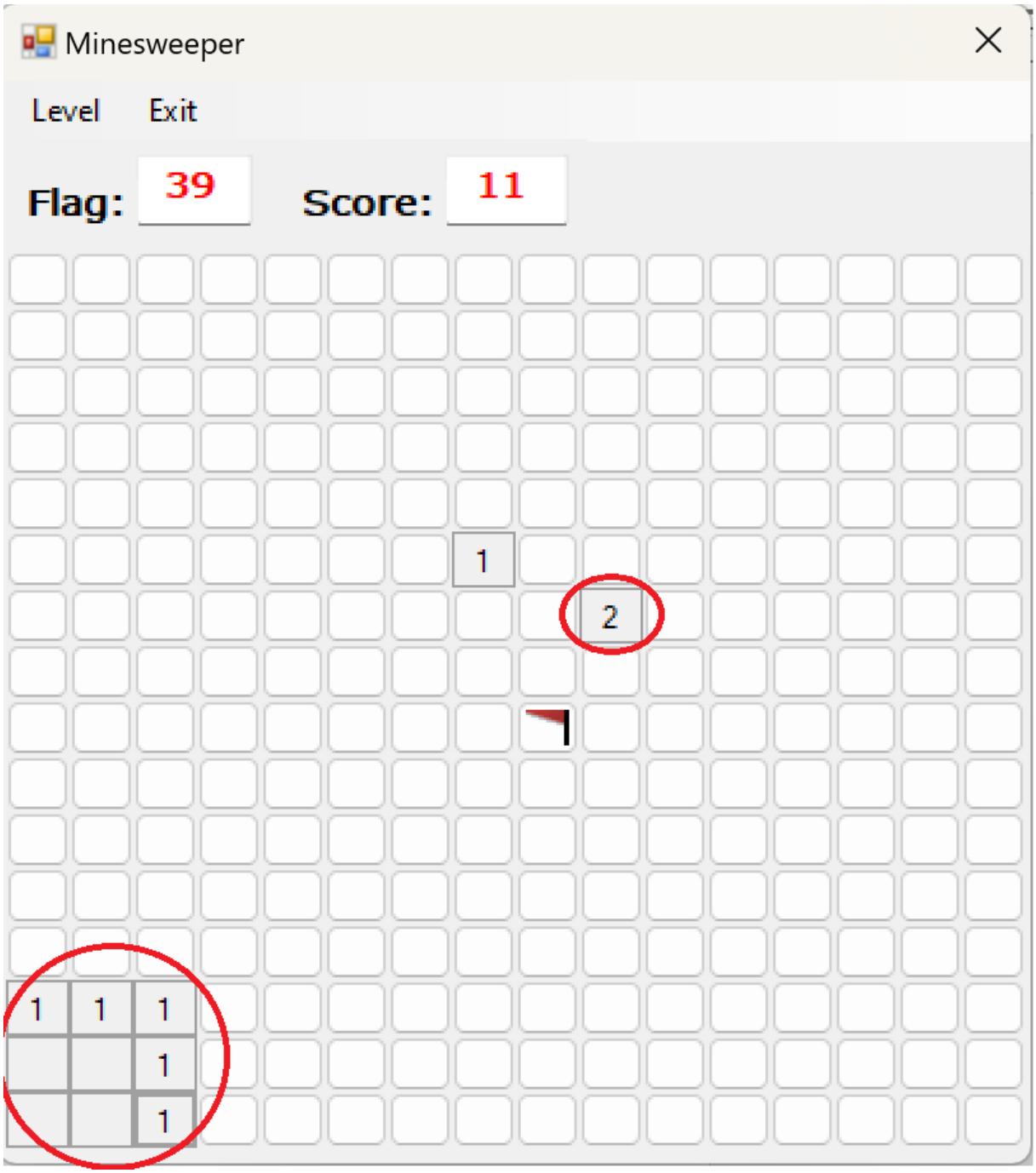
Nếu người chơi cho rằng một ô nào đó có mìn, người chơi có thể đánh dấu nó bằng cách nhấn chuột phải. Khi ô được đánh dấu, một lá cờ nhỏ sẽ xuất hiện trên ô đó, giúp họ ghi nhớ vị trí của mìn.



Hình 4. 8 Người chơi cắm cờ vào vô nghi ngờ có mìn

Người chơi tiếp tục nhấn vào các ô mà họ cho rằng không có mìn. Nếu một ô không có mìn và không có mìn nào ở các ô kề bên, tất cả các ô kề bên đó sẽ tự động được mở.

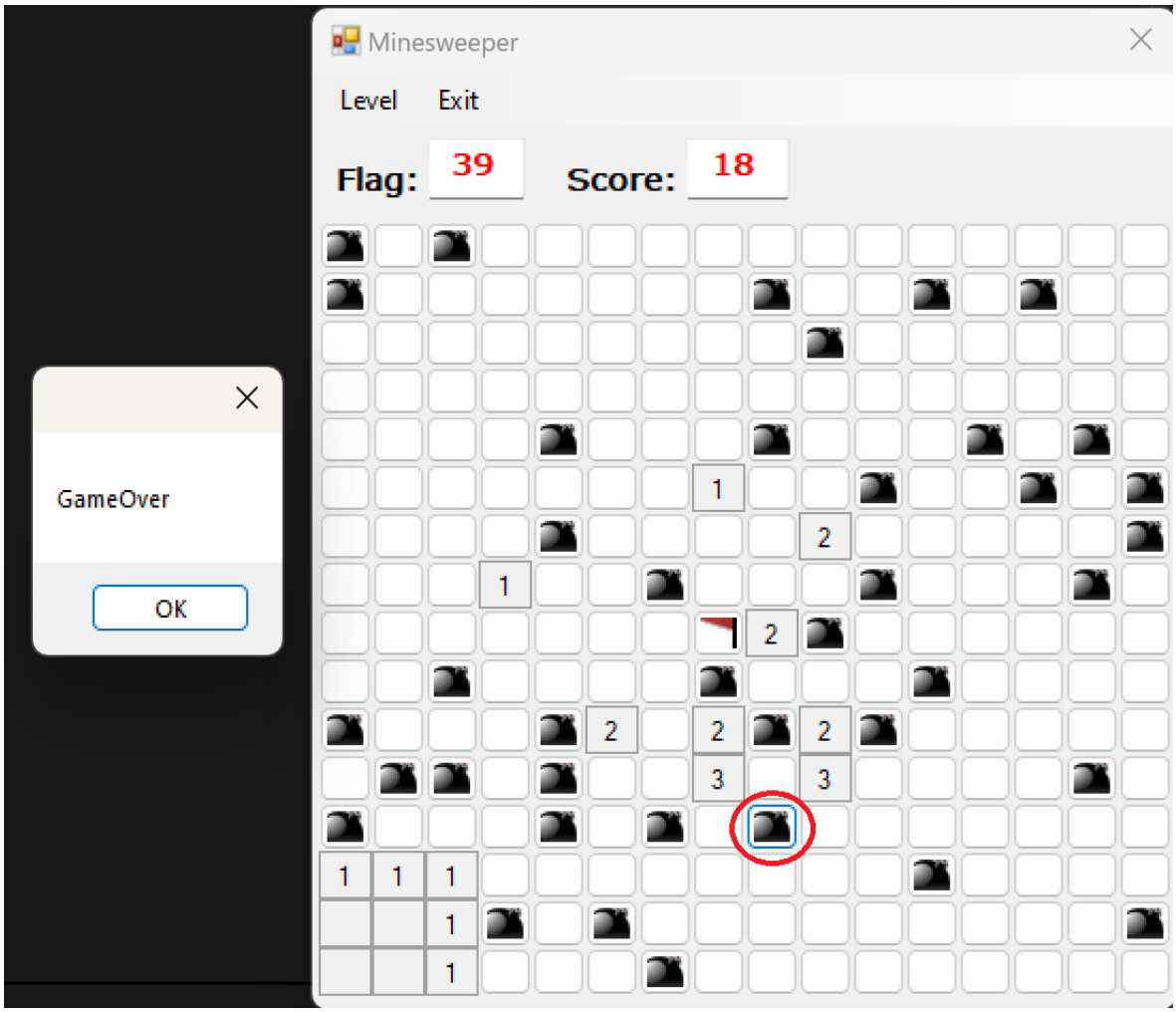
Nếu người chơi muốn thay đổi đánh dấu hoặc di chuyển khác, có thể nhấn lại vào ô đã đánh dấu để gỡ bỏ cờ hoặc chọn ô khác để di chuyển.



Hình 4. 9 Tiếp tục mở các ô người chơi cho rằng không có mìn

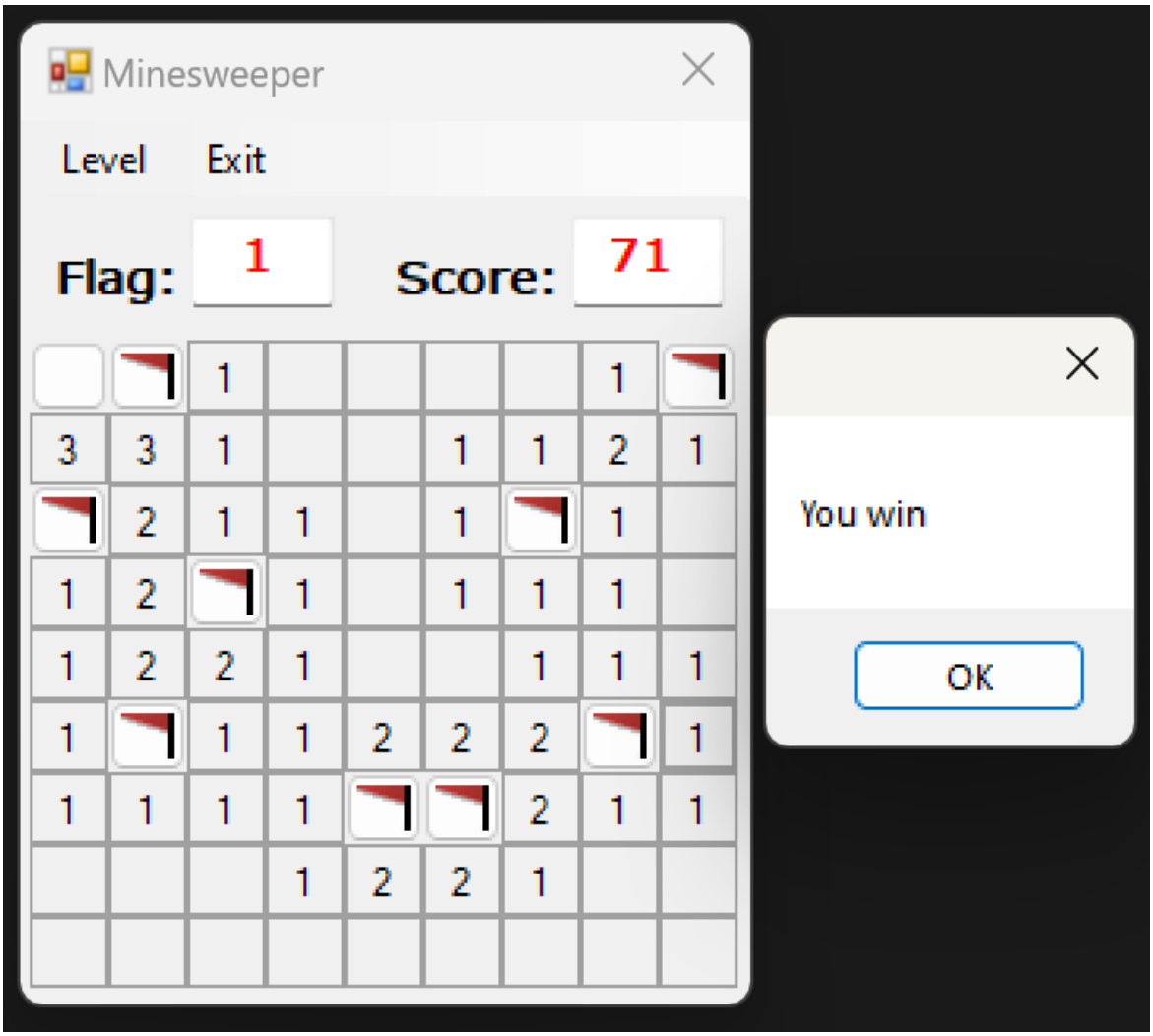
Dựa vào các con số xuất hiện trên bảng trò chơi, người chơi cần tính toán và suy luận để tìm ra các ô không chứa mìn. Ví dụ, nếu một ô có số 1 và kề bên nó chỉ có một ô chưa được mở, thì ô đó chắc chắn chứa mìn.

Nếu người chơi nhấn vào một ô chứa mìn, trò chơi sẽ kết thúc ngay lập tức (GameOver), người chơi thua cuộc và sẽ thấy tất cả các mìn được hiển thị trên bảng trò chơi. Người chơi có thể chơi lại bằng cách khởi động lại trò chơi.



Hình 4. 10 Trò chơi kết thúc khi người chơi nhấn vào ô có mìn

Người chơi tiếp tục mở các ô và họ chỉ chiến thắng khi tất cả các ô không chứa mìn được mở ra. Khi đó người chơi sẽ thấy một thông báo chiến thắng và có thể bắt đầu chơi lại hoặc chơi với cấp độ khác.



Hình 4. 11 Trò chơi kết thúc với kết quả người chơi chiến thắng

CHƯƠNG 5: KẾT LUẬN

5.1. Kết luận

Sau một thời gian thực hiện đề tài, cơ bản đã hoàn thành trò chơi đúng thời gian và kết quả thu được gần như đáp ứng được các yêu cầu cơ bản của một game Minesweeper. Thông qua quá trình tìm hiểu và xây dựng trò chơi Minesweeper chúng em đã có thêm nhiều kinh nghiệm, cũng như hiểu hơn về việc lập trình bằng ngôn ngữ C# như: thiết kế code, xây dựng source code và xử lý các sự kiện trong WinForm….

Việc xây dựng trò chơi chỉ đạt được các yêu cầu về các chức năng, vẫn còn nhiều thiếu sót do hạn chế về thời gian cũng như kinh nghiệm lập trình. Tuy nhiên trò chơi vẫn đảm bảo các chức năng cơ bản, cũng như các tính năng hỗ trợ người chơi trong suốt quá trình trải nghiệm.

Trong quá trình giảng dạy, nhờ có sự hướng dẫn và giúp đỡ tận tình của **thầy Nguyễn Thanh Trường**, chúng em đã hoàn thành được đồ án với đề tài "Lập trình Game Minesweeper". Từ đó chúng em đã thấy được 1 trò chơi được tạo ra như thế nào và cách vận hành nó ra sao,… đồng thời giúp chúng em nắm vững được các kiến thức cơ bản về ngôn ngữ lập trình C#, phương pháp xây dựng thuật toán, các biểu đồ flowchart.

5.2. Những kết quả đạt được

- Xây dựng được giao diện mở đầu cho trò chơi.
- Xây dựng hoàn chỉnh các chức năng cơ bản của trò chơi dò mìn: chọn ô, đánh dấu cờ, …
- Trò chơi có giao diện tương đối, người chơi dễ tiếp cận.
- Phát triển xây dựng được trò chơi ở ba cấp độ: dễ, trung bình, khó.

5.3. Hạn chế

- Giao diện của trò chơi còn đơn giản, chưa có nhiều sự đa dạng về hình ảnh hay âm thanh, làm giảm sự hấp dẫn của trò chơi đối với một số người chơi.
- Một khi đã thực hiện một bước đi, người chơi không thể hoàn tác lại bước đi đó. Điều này đặc biệt gây khó chịu khi người chơi vô tình nhấn nhầm

và thua trò chơi do lỗi này.

- Minesweeper là trò chơi đơn, thiếu các tính năng xã hội như bảng xếp hạng, chế độ nhiều người chơi.
- Thiếu chế độ hướng dẫn giúp người chơi mới học cách chơi. Điều này có thể làm cho người mới cảm thấy khó khăn trong việc bắt đầu.

5.4. Hướng phát triển

5.4.1 Hướng khắc phục những hạn chế

Tìm hiểu và xây dựng thêm các tính năng như:

- Chứng năng hoàn tác (Undo).
- Chế độ hướng dẫn chi tiết.
- Tính năng xếp hạng người chơi.
- Xây dựng giao diện đẹp mắt hơn, thêm âm thanh cho trò chơi sinh động hơn.

5.4.2 Hướng phát triển mở rộng đề tài

Nâng cao khả năng đồ họa của phần mềm trò chơi này, với nhiều hiệu ứng, âm thanh và màu sắc bắt mắt hơn, đồng thời sẽ xây dựng thêm một số chức năng như:

- Cải thiện giao diện:
    - Sử dụng thư viện đồ họa hiện đại như Unity, Unreal Engine, hoặc các framework UI như Qt, Electron để nâng cấp giao diện.
    - Thêm các hình ảnh và biểu tượng đẹp mắt, sử dụng sprite sheets để tạo các hiệu ứng chuyển động mượt mà.
- Thêm âm thanh: Sử dụng thư viện âm thanh như FMOD, OpenAL, hoặc các API âm thanh tích hợp trong các game engine để thêm nhạc nền và hiệu ứng âm thanh.
- Chức năng hoàn tác (Undo): Sử dụng cấu trúc dữ liệu stack để lưu lại các bước đi của người chơi, cho phép hoàn tác các bước trước đó.

- Bảng xếp hạng: Sử dụng cơ sở dữ liệu như Firebase, MySQL, hoặc MongoDB để lưu trữ điểm số của người chơi và hiển thị bảng xếp hạng.
- Chế độ nhiều người chơi: Sử dụng WebSockets hoặc một framework mạng như Photon, Mirror để tạo chế độ nhiều người chơi.
- Chế độ hướng dẫn chi tiết: Tạo một chế độ hướng dẫn tương tác với các bài học cụ thể và hiển thị từng bước.

Những cải tiến này sẽ giúp làm cho trò chơi Minesweeper trở nên hấp dẫn, thân thiện và thú vị hơn đối với người chơi ở mọi cấp độ.

TÀI LIỆU THAM KHẢO

[1] Wikipedia, *Dò mìn (trò chơi)*, Dò mìn (trò chơi) – Wikipedia tiếng Việt

[2] ThS. Nguyễn Thanh Trường (2024), *Giáo trình bài giảng Lập trình C#1*, Trường Đại học Tài chính – Marketing.

[3] Kteam, *Event Hover*, [C#] Event hover | How Kteam

[4] Kteam, *Tự động điều chỉnh size các control khi size Form thay đổi?*, C#, Tự động điều chỉnh size các control khi size Form thay đổi? | How Kteam