

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
NGÀNH CÔNG NGHỆ THÔNG TIN



BÁO CÁO ĐỒ ÁN
CẤU TRÚC DỮ LIỆU & GIẢI THUẬT
GAME 2048
VẬN DỤNG CẤU TRÚC DỮ LIỆU STACK

Giảng viên hướng dẫn: Th.S TRẦN CÔNG TÚ
SVTH: NGUYỄN QUỐC LONG
MSSV: 18110150
SVTH: NGUYỄN NHẬT HÀO
MSSV: 18110102

Mục lục

Mục lục	2
Danh mục các hình ảnh sử dụng trong bài	3
Danh mục các bảng sử dụng trong bài	3
Lời dẫn - Lý do chọn đề tài	4
I. Giới thiệu đề tài	5
1. <i>Giới thiệu game 2048</i>	5
1.1. Lịch sử phát triển	5
1.2. Đối tượng người chơi	6
1.3. Quy luật	6
2. <i>Sơ lược các tính năng trong game</i>	8
2.1. Tính năng Undo	8
2.2. Tính năng New Game	8
2.3. Tính năng Score	8
2.4. Tính năng Best Score	9
3. <i>Cấu trúc dữ liệu và giải thuật</i>	9
3.1. Cấu trúc dữ liệu Stack	9
3.2. Ứng dụng Stack vào game 2048	11
II. Quá trình thực hiện	11
1. <i>Thiết kế giao diện</i>	11
1.1. Lưới game	12
1.2. Các khối mang giá trị	12
1.3. Các thành phần khác (nút lệnh, hiển thị điểm số)	13
2. <i>Lập trình các hàm vận hành game</i>	13
2.1. Hàm vẽ lưới game	13
2.2. Hàm tạo khối giá trị ngẫu nhiên	14
2.3. Hàm đọc phím điều hướng từ bàn phím	14
2.4. Hàm lưu lại điểm số cao nhất & bắt đầu game mới	16
2.5. Hàm kiểm tra điều kiện kết thúc trò chơi	17
2.6. Hàm thoát trò chơi	18
3. <i>Cấu trúc dữ liệu Stack - giải thuật</i>	18
4. <i>Cài đặt và kiểm thử</i>	19
III. Phân công công việc	21
IV. Kết luận	23
1. <i>Đánh giá mức độ hoàn thành</i>	23
2. <i>Các khó khăn trong quá trình thực hiện</i>	23
3. <i>Hướng phát triển sản phẩm</i>	23
Tài liệu tham khảo	24

Danh mục các hình ảnh sử dụng trong bài

<i>Hình 1: Giao diện chính game 2048</i>	5
<i>Hình 2: Người chơi giành chiến thắng (tạo được khối 2048)</i>	7
<i>Hình 3: Trò chơi kết thúc (không còn nước đi hợp lệ)</i>	7
<i>Hình 4: Các tính năng trong game</i>	8
<i>Hình 5: Minh họa Stack (nguồn: techtalk.vn)</i>	9
<i>Hình 6: Các thao tác chính trên Stack</i>	10
<i>Hình 7: Lưới game</i>	12
<i>Hình 8: Các khối mang giá trị</i>	12
<i>Hình 9: Các thành phần khác (nút lệnh, hiển thị điểm số)</i>	13

Danh mục các bảng sử dụng trong bài

<i>Bảng 1: Kế hoạch thực hiện đồ án cuối kỳ môn cấu trúc dữ liệu & giải thuật - 2019</i>	21
--	----

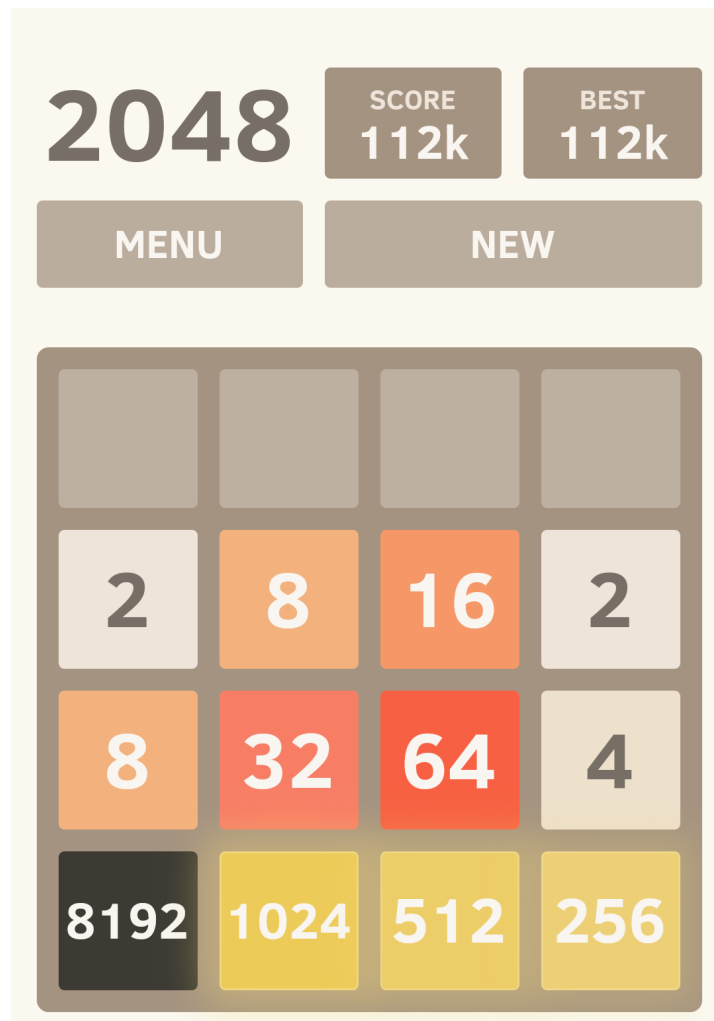
Lời dẫn - Lý do chọn đề tài

Sau mỗi giờ học, giờ làm việc căng thẳng, mệt mỏi, mỗi chúng ta đều mong tìm đến một hình thức giải trí nhẹ nhàng để làm giảm căng thẳng, lo lắng. Nắm bắt được điều này, nhóm tác giả gồm 2 thành viên, trong nhiều cách giải quyết vấn đề khác nhau, đã quyết định cùng nhau tự tay làm ra một trò chơi đơn giản với mong muốn trò chơi này sẽ được những người xung quanh sử dụng để giải trí, giải tỏa áp lực. Nhóm tác giả quyết định chọn làm game 2048 - một tựa game đơn giản, nhẹ nhàng nhưng cũng không kém phần lôi cuốn. Nhóm kết hợp dự án nhỏ này với đồ án Cấu trúc dữ liệu và giải thuật, cũng như vận dụng một số kiến thức từ môn học Lập trình hướng đối tượng để thực hiện. Nhóm chọn cách làm trên Windows application form sử dụng ngôn ngữ C# để đảm bảo bám sát với kiến thức được tiếp thu ở trường đại học. Rất mong sản phẩm nhỏ này nhận được sự ủng hộ từ quý thầy cô, học sinh, sinh viên hay với những người đang đi làm. Nhóm xin chân thành cảm ơn!

Các thành viên

I. Giới thiệu đề tài

1. Giới thiệu game 2048



Hình 1: Giao diện chính game 2048

1.1. Lịch sử phát triển

2048 là một trò chơi thể loại giải đố đơn người chơi được tạo ra vào 03/2014 bởi Gabriele Cirulli, một lập trình viên trẻ người Italia. Có thể xem 2048 là một dạng trò chơi giải đố trượt khối vuông (block puzzle) - thể loại trò chơi đã từng đạt được sự thành công trước đó với tựa game huyền thoại Tetris, song giao diện và luật chơi của 2048 hoàn toàn độc lập với những tựa game cùng thể loại.

Game được tạo ra lần đầu tiên dựa trên hai ngôn ngữ là Javascript và CSS. Chỉ trong vòng một tuần sau khi phát hành vào ngày 09/03/2014, game được đón nhận mạnh mẽ ở nhiều quốc gia khi đạt hơn 4 triệu lượt tải xuống. Từ đó game được tiếp tục phát triển để phục vụ nhiều nền tảng khác nhau như Web, Android, iOS,... với sự đa dạng về ngôn ngữ lập trình, giao diện và lối chơi của game.

1.2. Đối tượng người chơi

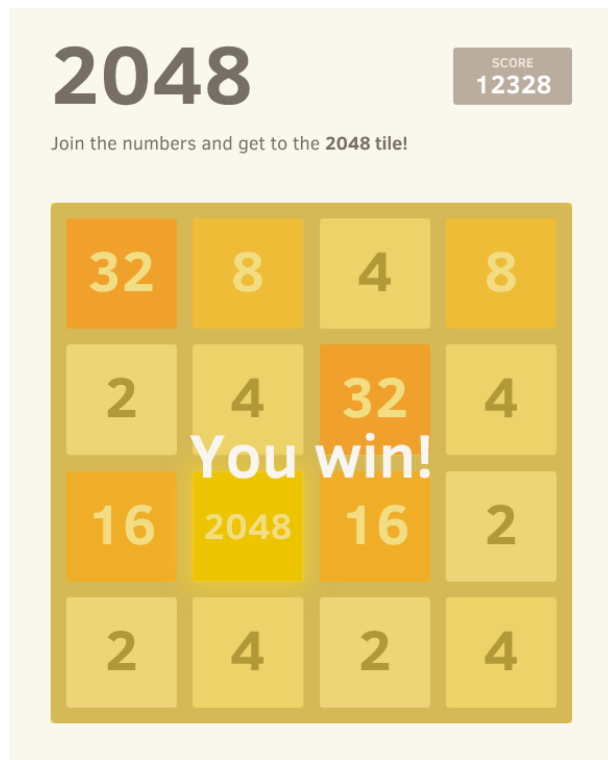
Do tính chất dễ chơi và số người chơi cần và đủ là 1, game phù hợp hầu hết tất cả mọi người từ trẻ em, học sinh, sinh viên đến những người đã đi làm - những người cần tìm đến một trò chơi hết sức đơn giản nhưng không kém phần gây nghiện để giải trí, thi nhau giành điểm cao với bạn bè, hay đơn giản để giải tỏa áp lực sau những giờ học, giờ làm việc căng thẳng.

1.3. Quy luật

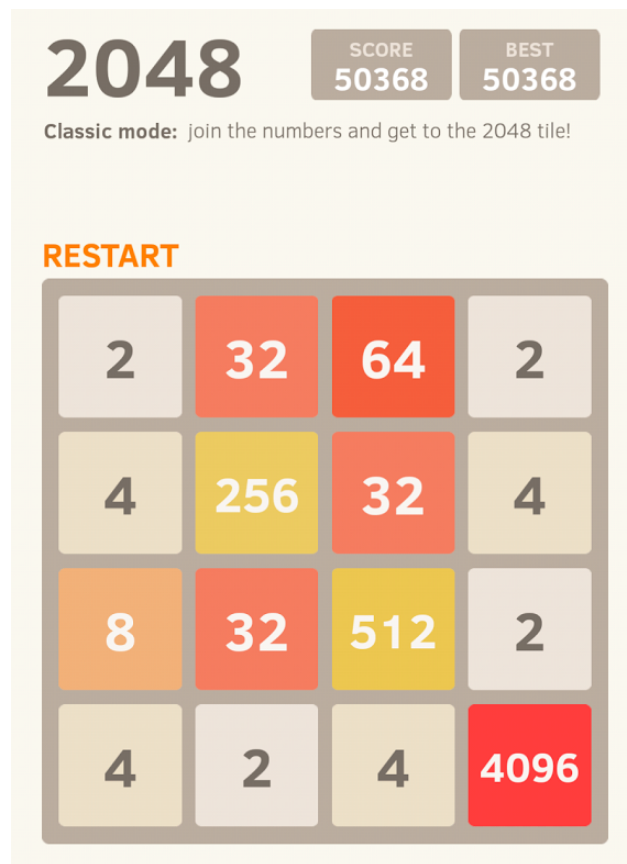
Người chơi 2048 sẽ thao tác trên một lưới ô vuông cạnh 4×4 . Người chơi sử dụng các phím mũi tên điều hướng lên, xuống, sang trái, sang phải để đưa các khối chứa một trong các giá trị thuộc lũy thừa của 2 (cao nhất là 1024 hoặc hơn, tùy theo luật chơi mà người lập trình quy định) trượt xa nhất có thể (các khối vuông ngừng trượt khi chạm biên của lưới, hoặc chạm vào khối vuông khác không mang cùng giá trị với chính nó) theo hướng tương ứng. Sau mỗi lượt di chuyển, một khối mang giá trị 2 hoặc 4 sẽ xuất hiện ngẫu nhiên vào một ô trống trên lưới. Nếu hai khối có cùng giá trị chạm vào nhau, chúng sẽ sáp nhập thành một khối mới mang giá trị bằng tổng giá trị hai khối cũ mang.

Theo luật chơi truyền thống, người chơi giành chiến thắng khi tạo được khối mang giá trị 2048. Lúc này người chơi có thể tiếp tục chơi để có thể đạt những giá trị cao hơn. Trò chơi kết thúc khi không còn nước đi hợp lệ (không còn ô trống trên lưới) hoặc khi người chơi chủ động kết thúc.

Ban đầu, người chơi có điểm số bằng 0. Sau mỗi lần hai khối sáp nhập thành một, nó cộng thêm vào tổng điểm số điểm bằng chính giá trị mà nó mang.

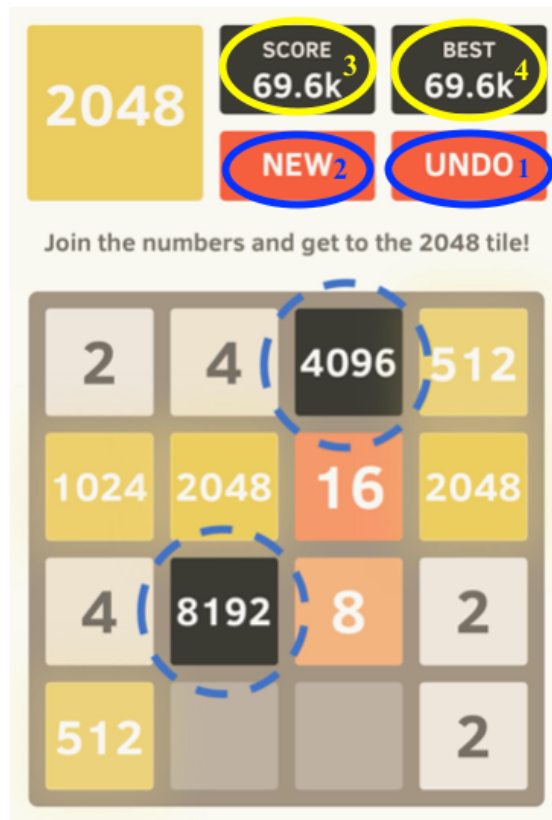


Hình 2: Người chơi giành chiến thắng (tạo được khối 2048)



Hình 3: Trò chơi kết thúc (không còn nước đi hợp lệ)

2. Sơ lược các tính năng trong game



Hình 4: Các tính năng trong game

2.1. Tính năng Undo

Trong quá trình chơi, người chơi không thể nào tránh khỏi việc di chuyển nhầm, hay mong muốn có thể đi lại 1 hoặc nhiều lượt đi gần đây để đạt kết quả tốt hơn. Tính năng Undo được ra đời để đáp ứng nhu cầu trên. Tính năng này sẽ được giải thích rõ hơn ở mục Ứng dụng cấu trúc dữ liệu Stack vào game.

2.2. Tính năng New Game

Khi người chơi gọi nút lệnh New Game, lưới trò chơi và điểm số sẽ được reset về ban đầu, sẵn sàng để chơi một ván hoàn toàn mới.

2.3. Tính năng Score

Tính năng này ghi lại và hiển thị điểm số của người chơi tính đến bước hiện hành, dựa theo luật tính điểm đã nêu ở mục Quy luật.

2.4. Tính năng Best Score

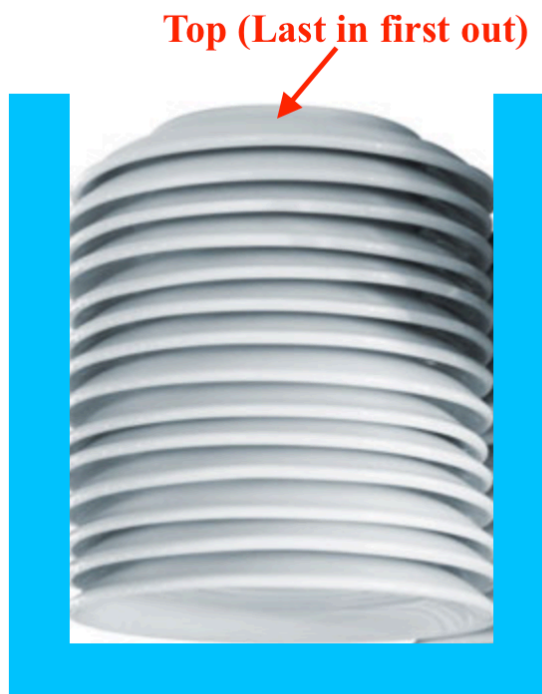
Sau mỗi lần trò chơi kết thúc, game so sánh điểm số với những lần chơi trước để hiển thị điểm số cao nhất. Best Score sẽ được cập nhật liên tục sau mỗi ván chơi.

3. Cấu trúc dữ liệu và giải thuật

3.1. Cấu trúc dữ liệu Stack

Stack (ngăn xếp), trong lĩnh vực khoa học máy tính, là một cấu trúc dữ liệu trừu tượng hoạt động theo nguyên lý Last In First Out (LIFO - vào sau ra trước). Để dễ dàng cho việc hình dung, có thể lấy ví dụ như sau:

Ta có một vài cái đĩa đang cần được xếp vào khay (giả sử diện tích đáy khay chỉ vừa để chứa một chiếc đĩa). Xếp lần lượt từng đĩa vào khay. Bây giờ, ta cần lấy đĩa ra để dùng. Chiếc đĩa được xếp vào khay cuối cùng sẽ được lấy ra đầu tiên (LIFO), tiếp đến là chiếc đĩa kế cuối được lấy ra,... và cuối cùng, ta lấy ra chiếc đĩa được xếp vào khay đầu tiên.



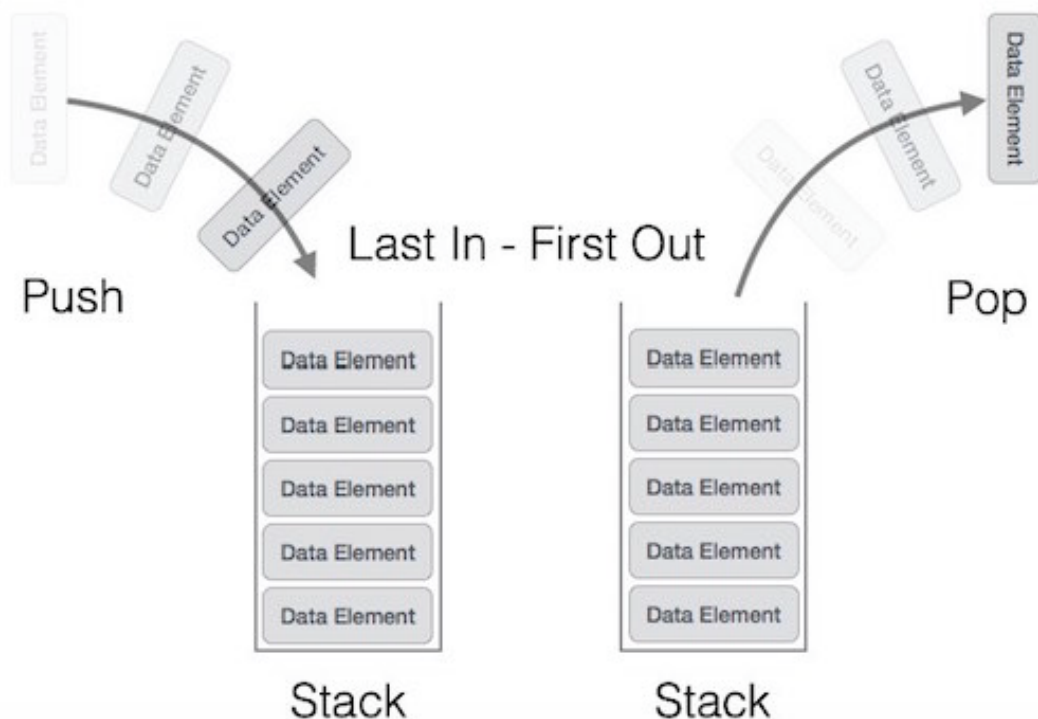
Hình 5: Minh họa Stack

Chiếc khay đựng đĩa bên trên minh hoạ cho container (khay chứa) của các node (phần tử) - những chiếc đĩa trong cấu trúc dữ liệu Stack. Có thể để bất kỳ chiếc đĩa nào vào khay trước, nhưng chỉ có thể lấy ra chiếc đĩa trên cùng (Top).

Dữ liệu chính trong ví dụ minh hoạ trên về Stack là chiếc đĩa. Rộng hơn, trong Stack có thể có nhiều loại dữ liệu: interger, double, string, array (1 hoặc nhiều dimension), struct,... được vận dụng linh hoạt trong các bài toán khác nhau, bởi những cách lập trình khác nhau.

Để xử lý dữ liệu trong một cấu trúc dữ liệu cần có những thao tác, phương thức hợp lý. Đối với Stack, một số thao tác chính trên cấu trúc dữ liệu này là:

- `isEmpty()`: Kiểm tra xem liệu Stack có rỗng hay không
- `Top()`: Trả về giá trị phần tử Top mà không huỷ nó khỏi Stack. Xảy ra lỗi nếu `isEmpty()` trả về giá trị true
- `Pop()`: Trả về giá trị phần tử Top và xoá nó khỏi Stack
- `Push(<Dữ liệu>)`: Thêm 1 phần tử vào đỉnh ngăn xếp.



Hình 6: Các thao tác chính trên Stack (nguồn: techtalk.vn)

3.2. Ứng dụng Stack vào game 2048

Trong quá trình chơi game, người chơi thường có nhu cầu huỷ lượt đi gần nhất để chơi lại lượt này. Tính năng Undo được xây dựng để phục vụ nhu cầu trên của người chơi. Ở đây, nhóm ứng dụng cấu trúc dữ liệu Stack để tạo ra tính năng này.

Nguyên lý là sau mỗi lượt đi, lưới game hiện hành (mảng 4×4 các số nguyên) sẽ được đẩy (`Push(array)`) vào Stack (tức mảng 4×4 là dữ liệu của Stack này). Mỗi khi người dùng gọi tính năng Undo, lưới game gần nhất sẽ được lấy ra (`Pop()`) khỏi Stack để trở thành lưới game hiện hành đồng thời điểm số được hoàn tác. Tương tự cho những lần Undo tiếp theo mà người chơi gọi.

Ta có thể ứng dụng Stack để tạo ra tính năng Redo (hoàn tác thao tác Undo) bằng cách: luôn đẩy lưới game hiện hành vào `Redo_Stack` (trước đó cần lấy lưới game cũ ra). Mỗi khi người chơi gọi tính năng Undo, đẩy lưới game hiện hành vào `Redo_Stack` mà không lấy lưới game cũ ra. Khi người chơi gọi Redo, lấy lưới game gần nhất trong `Redo_Stack` làm lưới game hiện hành.

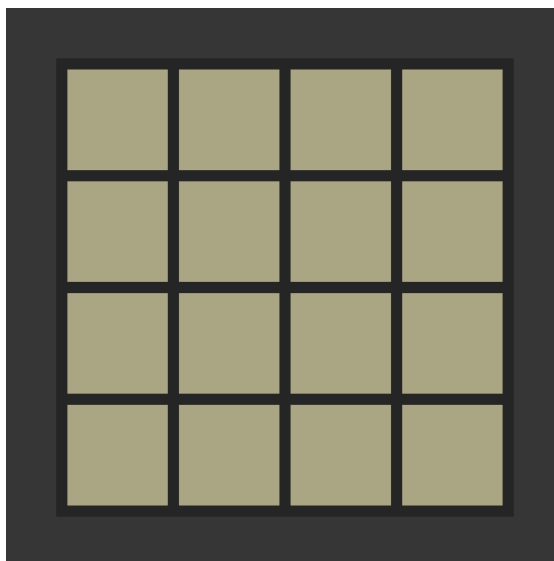
Tuy nhiên, do tính năng Redo không phù hợp lắm với tính chất ngẫu nhiên của game, ở đây nhóm không mang tính năng này vào game.

II. Quá trình thực hiện

1. Thiết kế giao diện

Nhóm sử dụng Windows application form dựa trên ngôn ngữ C# để thiết kế giao diện game 2048.

1.1. Lưới game



Hình 7: Lưới game

Lưới game chiếm phần lớn diện tích phần giao diện chính. Lưới game gồm 4×4 ô vuông diện tích 100×100 (pixel), khoảng cách giữa mỗi lưới là 5 (pixel)

1.2. Các khối mang giá trị



Hình 8: Các khối mang giá trị

Các khối mang giá trị là những ô vuông diện tích 100×100 (pixel), sẽ được đặt chồng lên những phần diện tích 100×100 trên lưới game mỗi khi nó xuất hiện. Ở đây minh hoạ đến giá trị 2048.

1.3. Các thành phần khác (nút lệnh, hiển thị điểm số)



Hình 9: Các thành phần khác (nút lệnh, hiển thị điểm số)

Một số nút lệnh và thông tin được thêm vào để mang đến trải nghiệm chơi game tốt hơn cho người chơi:

- New game: reset lưới game để sẵn sàng cho ván mới
- Undo: trả về lưới game ngay trước lượt đi gần nhất
- Score: Tính và hiển thị điểm số của ván game
- Best Score: Hiển thị điểm số cao nhất trong những ván đã chơi

2. Lập trình các hàm vận hành game

2.1. Hàm vẽ lưới game

Hàm bên dưới (sử dụng thư viện System.Drawing) được dùng để vẽ 4×4 ô vuông diện tích 100×100, khoảng cách giữa mỗi khối là 5 mỗi khi một ván mới được bắt đầu

```
private void Load_Form()
{
    gameOverPanel.Visible = false;
    gameOverPanel.Enabled = false;
    lb_diem.Text = "0";
    lb_bestScore.Text = doc.ReadToEnd();
    doc.Close();
    if (lb_bestScore.Text == "")
        lb_bestScore.Text = "0";





    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            blockLabel[i, j] = new Label();
            blockLabel[i, j].Location = new Point(KhoangCach + i * (100 + KhoảngCach), KhoảngCach + j * (100 + KhoảngCach));
            blockLabel[i, j].Size = new Size(100, 100);
            blockLabel[i, j].TabIndex = i * 4 + j;
            blockLabel[i, j].Name = String.Format("lb%d%d", i, j);
            blockLabel[i, j].BackColor = Color.FromName("ActiveBorder");
            blockLabel[i, j].Font = new Font("FC BARCELONA", 40F, FontStyle.Bold, GraphicsUnit.Point, ((byte)0));
            blockLabel[i, j].TextAlign = ContentAlignment.MiddleCenter;
            groupBox1.Controls.Add(blockLabel[i, j]);
        }
    }
    Load0();
}
```

2.2. Hàm tạo khối giá trị ngẫu nhiên

Sau khi người chơi nhấn các phím di chuyển để thực hiện lượt chơi đầu tiên cũng như sau mỗi lượt chơi, một khối mang giá trị ngẫu nhiên 2 hoặc 4 được sinh ra ở vị trí ngẫu nhiên (trường hợp lưới còn ô trống). Hàm bên dưới đảm nhận vai trò này.

```
bool randomViTriHien()
{
    bool isDo = false;
    List<int> test = new List<int>();
    for (int i = 0; i < 16; i++)
    {
        if (block[i / 4, i % 4] == 0)
        {
            test.Add(i);
            isDo = true;
        }
    }
    if (test.Count > 0)
    {
        int set = test[Rand.Next(0, test.Count - 1)];
        while (block[set / 4, set % 4] != 0 && test.Count > 1)
        {
            test.Remove(set);
            set = test[Rand.Next(0, test.Count - 1)];
        }
        block[set / 4, set % 4] = Rand.Next(1, 100) > 90 ? 4 : 2;
    }
    return isDo;
}
```

2.3. Hàm đọc phím điều hướng từ bàn phím

Sử dụng lệnh Keys thuộc thư viện System.Windows.Forms để đọc các thao tác người chơi thực hiện từ bàn phím (các phím mũi tên chỉ hướng). Ở đây cho phép người chơi sử dụng các phím W, A, S, D thay thế lần lượt cho , ,  và . Ngoài ra trò chơi cũng đọc được thao tác tổ hợp phím Ctrl + Z để thực hiện thao tác Undo.

```

private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyData == Keys.Up || e.KeyData == Keys.W)
        PhimLen();
    if (e.KeyData == Keys.Down || e.KeyData == Keys.S)
        PhimXuong();
    if (e.KeyData == Keys.Right || e.KeyData == Keys.D)
        PhimPhai();
    if (e.KeyData == Keys.Left || e.KeyData == Keys.A)
        PhimTrai();

    this.Refresh();
    if (CheckGameOver())
    {
        drawGameOver();
        tabUndo.Enabled = false;
        if (Convert.ToInt32(lb_diem.Text) > Convert.ToInt32(lb_bestScore.Text))
        {
            lb_bestScore.Text = lb_diem.Text;
            lbDiem_.Text = lb_diem.Text;
            lbBestScore_.Text = lb_bestScore.Text;
            doc.Close();
            StreamWriter ghi = new StreamWriter("LuuDiem.txt");
            ghi.WriteLine(lb_diem.Text);
            ghi.Flush();
            ghi.Close();
        }
        else
        {
            lbDiem_.Text = lb_diem.Text;
            lbBestScore_.Text = lb_bestScore.Text;
        }
    }
}

```

Sau khi đọc thao tác di chuyển, trò chơi sẽ đẩy mảng trước đó vào Stack, di chuyển và hợp nhất các khối (nếu có), đồng thời sinh ngẫu nhiên một khối mới mang giá trị 2 hoặc 4 ở vị trí ngẫu nhiên theo quy luật. Mỗi thao tác phím sẽ gọi hàm tương ứng để thực hiện điều này. Ở đây hàm `PhimLen()` thực hiện những thao tác mà chương trình thực hiện sau khi người chơi nhấn phím di chuyển lên. Cách viết các hàm cho phím di chuyển xuống, sang trái và sang phải tương tự như cho phím di chuyển lên. (`Undo()` là thao tác đẩy mảng trước khi di chuyển vào Stack).

```

bool PhimLen()
{
    Undo();
    bool isDo = false;

    for (int x = 0; x < 4; x++)
    {
        for (int y = 0; y < 3; y++)
        {
            for (int y1 = y + 1; y1 < 4; y1++)
            {
                if (block[x, y1] > 0)
                {
                    if (block[x, y] == 0)
                    {
                        block[x, y] = block[x, y1];
                        block[x, y1] = 0;
                        y--;
                        isDo = true;
                    }
                    else if (block[x, y] == block[x, y1])
                    {
                        block[x, y] *= 2;
                        block[x, y1] = 0;
                        isDo = true;
                        Diem += block[x, y];
                    }
                    break;
                }
            }
        }
    }
}

```

2.4. Hàm lưu lại điểm số cao nhất & bắt đầu game mới

Khi người dùng gọi tính năng New Game, trò chơi sẽ so sánh điểm số ván hiện tại với điểm số cao nhất (High Score) được lưu trong file "LuuDiem.txt". Nếu điểm ván hiện tại lớn hơn điểm số cao nhất thì ghi dữ liệu điểm số hiện tại thành điểm số cao nhất vào file.


```

private void tabNewGame_Click(object sender, EventArgs e)
{
    gameOverPanel.Visible = false;
    gameOverPanel.Enabled = false;
    if (Convert.ToInt32(lb_diem.Text) > Convert.ToInt32(lb_bestScore.Text))
    {
        doc.Close();
        StreamWriter ghi = new StreamWriter("LuuDiem.txt");
        ghi.WriteLine(lb_diem.Text);
        ghi.Flush();
        ghi.Close();
        this.Hide();
        Form1 frm = new Form1();
        frm.Show();
    }
    else
    {
        this.Hide();
        Form1 frm = new Form1();
        frm.Show();
    }
}

```

2.5. Hàm kiểm tra điều kiện kết thúc trò chơi

```

bool CheckGameOver()
{
    for (int x = 0; x < 4; x++)
    {
        for (int y = 0; y < 4; y++)
        {
            if (block[x, y] == 0 ||
                (y < 3 && block[x, y] == block[x, y + 1]) ||
                (x < 3 && block[x, y] == block[x + 1, y]))
            {
                return false;
            }
        }
    }
    return true;
}

```

Sau mỗi lượt di chuyển, nếu trên lưới vẫn còn vị trí trống thì game sẽ được tiếp tục. Ngược lại sẽ hiển thị thông báo trò chơi kết thúc khi không còn vị trí trống nào trên lưới (không còn 2 khối liền kề nào mang giá trị bằng nhau).

2.6. Hàm thoát trò chơi

Một messageBox sẽ hiển thị để hỏi người chơi chắc chắn muốn thoát trò chơi hay không

```
private void tabExit_Click(object sender, EventArgs e)
{
    if (MessageBox.Show(" Are you sure ? ", " Thoát ", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.No)
        Application.Exit();
    else
    {
        return;
    }
}
```

3. Cấu trúc dữ liệu Stack - giải thuật

Như đã giới thiệu ở phần I.3.2, đối với tính năng Undo trong game 2048, mỗi Node trong Stack là một mảng 4×4 chứa giá trị của toàn bộ vị trí trên lưới (nhận giá trị bằng 0 với những vị trí chưa có khối giá trị). Một Stack khác được dùng để lưu lại điểm số (điểm số là Node trong Stack này) sau mỗi lần di chuyển.

```
void Undo()
{
    int[,] undo = new int[4, 4];
    for (int i = 0; i < 4; i++)
        for (int j = 0; j < 4; j++)
        {
            undo[i, j] = block[i, j];
        }
    PreScore.Push(Diem);
    myStack_Undo.Push(undo);
}
```

Do hàm trên đảm nhận việc đẩy mảng giá trị cũng như điểm vào Stack ngay sau mỗi lần người chơi nhấn phím di chuyển, cần gọi hàm này ngay khi khai báo hàm `PhimLen()`, `PhimXuong()`, `PhimTrai()`, `PhimPhai()` (mục II.2.3)

Mỗi khi người chơi gọi tính năng Undo, hàm `tabUndo_Click` thực thi, điểm sẽ được `Pop()` từ Stack chứa điểm, đồng thời mảng chứa giá trị toàn bộ lưới cũng sẽ được `Pop()` từ Stack. Một vòng lặp sẽ đọc dữ liệu từ mảng này và ghi vào lưới game hiện hành.

```

private void tabUndo_Click(object sender, EventArgs e)
{
    if (myStack_Undo.Count == 0)
        return;
    if (myStack_Undo != null)
    {
        int[,] tempUndo = new int[4, 4];
        tempUndo = myStack_Undo.Pop();
        Diem = PreScore.Pop();
        //PreScore.Clear();
        for (int i = 0; i < 4; i++)

            for (int j = 0; j < 4; j++)
            {
                block[i, j] = tempUndo[i, j];
            }
        for (int i = 0; i < 4; i++)
        {
            for (int j = 0; j < 4; j++)
            {
                if (block[i, j] == 0)
                    blockLabel[i, j].Text = "";
                else
                    blockLabel[i, j].Text = block[i, j].ToString();
                setMauCho0(i, j);
            }
        }
        lb_diem.Text = Diem.ToString();
    }
    else
    {
        return;
    }
}

```

4. Cài đặt và kiểm thử

Để chắc chắn mọi tính năng trong trò chơi hoạt động đúng đắn, cần trải qua 1 số trường hợp kiểm thử sau:

- Kiểm tra xem các phím di chuyển có được đọc và thực hiện đúng nhiệm vụ hay chưa.
- Chơi trò chơi cho đến khi trò chơi đạt điều kiện kết thúc. Lúc này nếu một message box hiện ra thông báo Trò chơi kết thúc thì tính năng hoạt động. Nếu trong quá trình chơi, điều kiện kết thúc game chưa đạt mà thông báo này đã hiện ra thì cần kiểm tra lại chương trình.

- Chơi game & quan sát lượt đi đầu tiên cũng như trong các lượt kế tiếp xem cách sinh khối mang giá trị ngẫu nhiên đã đúng đắn hay chưa, đồng thời quan sát cách các khối di chuyển cũng như hợp nhất đã đúng với quy luật hay chưa, các khối mang giá trị có thay đổi đúng màu sắc chưa.
- Chơi một vài lượt sau đó nhấn New Game xem tính năng này hoạt động hay không. Kiểm tra tương tự với nút lệnh Exit.
- Chơi một vài lượt xem sự thay đổi của điểm số và điểm số cao nhất. Kết thúc game và chơi lần nữa (kết quả cao hơn lần trước) xem sự thay đổi của Best Score. Chơi thêm lần nữa (kết quả thấp hơn lần trước) xem Best Score thay đổi hay không.
- Chơi một vài lượt, sau đó nhấn nút Undo xem tính năng này đã hoạt động hay chưa, có quay về đúng lưới giá trị trước đó hay không. Kiểm tra tương tự cho tổ hợp phím Ctrl + Z.

Do sau khi trò chơi có một khối đạt giá trị 2048, người chơi vẫn có thể tiếp tục chơi để đạt những khối mang giá trị cao hơn, nhóm không hiển thị thông báo Chiến thắng khi người chơi đạt khối mang giá trị này.

Tính đến thời điểm hoàn thành báo cáo này, nhóm đã kiểm tra nhiều lần cho mỗi trường hợp kiểm thử và chưa nhận thấy lỗi nào phát sinh làm chương trình chạy sai.

III. Phân công công việc

Dưới đây là bảng phân công nhiệm vụ của các thành viên trong nhóm, cũng như thời gian bắt đầu - kết thúc dự kiến, thời gian bắt đầu - kết thúc thực tế từng nhiệm vụ trong quá trình thực hiện sản phẩm

KẾ HOẠCH THỰC HIỆN ĐỒ ÁN CUỐI KỲ MÔN CẤU TRÚC DỮ LIỆU & GIẢI THUẬT - 2019							
DANH SÁCH CÔNG VIỆC	NỘI DUNG CHI TIẾT CÔNG VIỆC	Ng. Quốc Long	Ng. Nhật Hào	Ngày bắt đầu (dự kiến) (2019)	Ngày kết thúc (dự kiến) (2019)	Ngày bắt đầu (thực tế) (2019)	Ngày kết thúc (thực tế) (2019)
Thiết kế giao diện game	Lên ý tưởng giao diện chính trên giấy	×	×	26/9	28/9	27/9	27/9
	Tạo Windows form application cho giao diện chính (bao gồm lưới 4×4, các mục Score và Best Score, các button New Game và Undo, bảng chọn Help)	×		28/9	30/9	30/9	30/9
	Vẽ các khối mang giá trị trên Windows form application	×		28/9	30/9	30/9	30/9

	Tạo Windows form application cho bảng chọn Help: Hiển thị luật chơi và cách chơi 2048		×	1	3/10	11/11	11/11
Lập trình các hàm vận hành game	Hàm tạo lưới 4×4		×	2/10	5/10	4/10	4/10
	Hàm đọc các phím di chuyển từ bàn phím	×		3/10	5/10	4/10	5/10
	Hàm quy luật di chuyển và hợp nhất các khối	×		3/10	5/10	4/10	8/10
	Hàm chiến thắng & kết thúc game		×	6/10	9/10	11/10	11/10
	Hàm tính điểm & đọc ghi điểm số cao nhất	×		6/10	9/10	11/10	12/10
	Hàm Undo sử dụng cấu trúc dữ liệu Stack	×	×	10/10	16/10	21/10	10/11
Cài đặt và kiểm thử game	Sửa các cú pháp và ngữ nghĩa gặp phải	×	×	18/10	26/10	25/10	14/11
	Trình bày đồ án với giảng viên để kiểm tra, nhận xét & sửa chữa	×	×	27/10	21/11	26/10	16/11
Làm tiểu luận báo cáo & bài trình chiếu			×	27/10	21/11	16/11	22/11

IV. Kết luận

1. Đánh giá mức độ hoàn thành

Nhìn chung, sản phẩm đáp ứng hầu hết những tính năng và quy luật mà một game 2048 cần đạt được, cũng như yêu cầu mà giảng viên và nhóm đặt ra từ đầu. Ưu điểm của sản phẩm là gần như đã hoàn thiện và có thể sử dụng ngay cho nhu cầu giải trí. Tuy nhiên ở đây nhóm lượt đi hai tính năng cảm thấy chưa cần thiết lắm là phân âm thanh in-game và phân thông báo chiến thắng khi người chơi đạt khối mang giá trị 2048. Ngoài ra, một khuyết điểm lớn còn tồn đọng ở sản phẩm mà nhóm chưa sửa chữa là trong quá trình di chuyển, các khối mang giá trị không di chuyển từ từ mà đi thẳng đến biên của lưới hoặc hợp nhất với ô khác ngay, làm người chơi không nhận ra được chuyển động của từng khối, khó hình dung hơn về cách các khối di chuyển. Nhóm xin tự đánh giá mức độ hoàn thiện của sản phẩm game 2048 cho đồ án là vào khoảng 85%.

2. Các khó khăn trong quá trình thực hiện

Đề tài game 2048 nhìn chung tương đối khó, là thử thách đối với nhóm thực hiện, khi cần kết hợp kiến thức từ hai môn học chưa hoàn thành là *Cấu trúc dữ liệu và giải thuật* và *Lập trình hướng đối tượng*. Từ đó nhóm cần tự tìm hiểu về cấu trúc dữ liệu Stack cũng như lập trình Windows application form sử dụng ngôn ngữ C#. Tuy nhiên nhóm đã nỗ lực tìm tòi, tham khảo từ nhiều nguồn cũng như thường xuyên họp nhóm, xin ý kiến từ giảng viên phụ trách môn học để tìm ra nguyên nhân và cách khắc phục.

3. Hướng phát triển sản phẩm

Ý tưởng phát triển đồ án của nhóm bắt nguồn từ những vấn đề chính & dễ dàng xử lý hơn (chọn ngôn ngữ & phương pháp lập trình, làm thế nào để game vận hành được). Sau đó là cách thiết kế giao diện game và các nút lệnh cũng như thông báo để nâng cao trải nghiệm người chơi. Phần Ứng dụng Stack vào tính năng Undo được nhóm hướng đến cuối cùng vì đối với nhóm đây là phần khó và cần thời gian trau dồi thêm.

Tài liệu tham khảo

- Lê Minh Hoàng (1999). *Giải thuật & Lập trình*, Đại học Sư phạm Hà Nội.
- Wikipedia (2019). *Stack (abstract data type)*, [https://en.wikipedia.org/wiki/Stack_\(abstract_data_type\)](https://en.wikipedia.org/wiki/Stack_(abstract_data_type))
- Wikipedia (2019). *2048 (video game)*, [https://en.wikipedia.org/wiki/2048_\(video_game\)](https://en.wikipedia.org/wiki/2048_(video_game))
- Robert V. Huang (2019). *A Walkthrough to Implement 2048 Game*, https://www.codeproject.com/Articles/1173541/A-Walkthrough-to-Implement-Game?fbclid=IwAR1low5zQ9kRPZ60Jv_Uy9JhM2MtessyoeEb_kzodJDmXVRG9JK-P77ZSg