KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

Công nghệ JAVA

Đề tài: Giới thiệu game N Puzzle

Lớp: Công nghệ thông tin 1 – N06

Khoá: K61

Nhóm: 06

Người hướng dẫn: Nguyễn Trọng Phúc

Sinh viên thực hiện:

Nguyễn Đức Lộc (nhóm trưởng)

Nguyễn Việt Anh

Trần Trọng Nghĩa

Hà nội, ngày 03 tháng 04 năm 2022

Mục Lục

Lời nói đầu

Trong thời đại công nghệ thông tin phát triển thì game sẽ là 1 ngành phát triển nhanh nhất tại thời điểm này. Các game được tạo ra đáp ứng với như cầu của xã hội nhanh chóng, giải trí đồng thời các thao tác và giao diện dễ dàng sử dụng, và với việc thay đổi thường xuyên, các game hiện nay luôn được cập nhật phù hợp theo nhu cầu của người dùng. Tuy nhiên việc tạo game không bao giờ là dễ dàng cho các lập trình viên, trong đó ngôn ngữ Java là một trong những ngôn ngữ mà các lập trình viên thường sử dụng khi tiến vào chuyên nghành làm game. Vậy tại sao lại là Java? Lí do lớn nhất mà các lập trình viên lựa chọn Java đó là ngôn ngữ Java có khả năng hướng đối tượng như C++ nhưng nó lại có 1 khả năng đặc trưng là "viết code 1 lần, thực thi khắp nơi". Khi xây dựng chương trình bằng Java, ta có thể chạy ứng dụng trên mọi nền tảng hệ điều hành khác nhau, đặc biệt Java dễ dàng phát triển trên các thiết bị di động với lí do Java là 1 trong những ngôn ngữ phổ biến nhất thế giới.

Vậy lập trình Java nó có khó hay không, trên cơ sở lý thuyết Java cũng như C++ đều hướng đối tượng tuy nhiên với Java ta lại có 1 kho thư viện hỗ trợ các lập trình viên trong việc thiết kế giao diện và và trong việc chạy trên các hệ thống khác nhau. Và để hiểu rõ hơn về cấu trúc và các lệnh Java , tại bản báo cáo này nhóm chúng em sẽ ví dụ về cách lập trình Java thông qua 1 game – "sở trưởng của Java" , game mà nhóm chúng em chọn trên nguyên tắc đơn giản dễ hiểu, và dễ thực hiện sẽ giúp ai chưa biết về Java sẽ hiểu rõ hơn về nó và thao tác thực hiện và tạo 1 game.

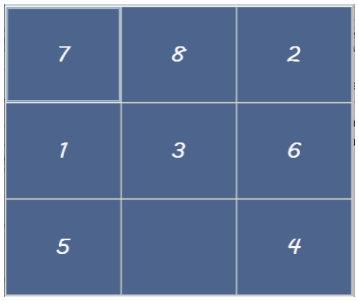
Game sẽ được tạo bằng Java mà nhóm chúng em chọn sẽ là **N puzzle** . Đây là một game cơ bản và chắc rằng bạn nào cũng sẽ chơi thử 1 lần rồi vì nó đã được tích hợp vào kho ứng dụng trò chơi của các điện thoại Nokia cũ nhưng với hình thức là ghép tranh. Ngoài ra lí do nhóm chúng em chọn **N puzzle** vì đây là một game khá hay về thuật toán cách giải trò chơi – về vấn đề này nếu có cơ hội nhóm chúng em sẽ trình bày sau trong trí tuệ nhân tạo.

Trong bản báo cáo này sẽ là game **N puzzle** được viết bằng Java, tuy còn nhiều sai sót nhưng mong các bạn và quý thầy cô có thể thông cảm và góp ý để có thể hoàn thiện thêm về việc lập trình game đồng thời là các kiến thức học thuật khác hỗ trợ chúng em về sau làm việc.

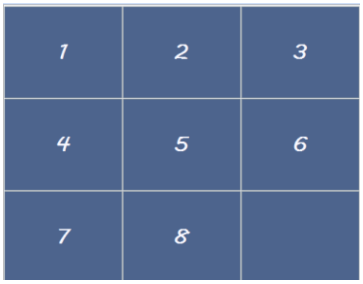
I. Giới thiệu game N puzzle

N puzzle là 1 game cơ bản và cũng rất quen thuộc với chúng ta. Đây là một game đã được tích hợp trên điện thoại nokia với tiêu đề chơi là ghép tranh. Về cấu trúc game và luật chơi sẽ như sau:

- Về cấu trúc game: Bài toán sẽ bao gồm 1 bảng n*n trong đó n là 1 số tự nhiên bất kì (tuy nhiên để dễ dàng trong việc thiết kế n sẽ có khoảng từ 3 đến 8). Tại mỗi ô của bảng sẽ có các số từ 1 đến n*n – 1và trong bảng sẽ có 1 ô không được đánh dấu.
- Về luật chơi: nhiệm vụ của người chơi là phải đưa được các ô về theo sắp xếp của thứ tự từ bé đến lớn hay nói cách khác theo kiểu lợp ngói (lớn dần từ trái qua phải và từ trên xuống) . Người chơi sẽ dành chiến thắng khi đưa các ô về kiểu lợp ngói. Tại mỗi lần chơi các ô sẽ được sắp xếp ngẫu nhiên nên mỗi lần sẽ là 1 cách chơi khác nhau.



Ảnh *1*: Khi bắt đầu trò chơi

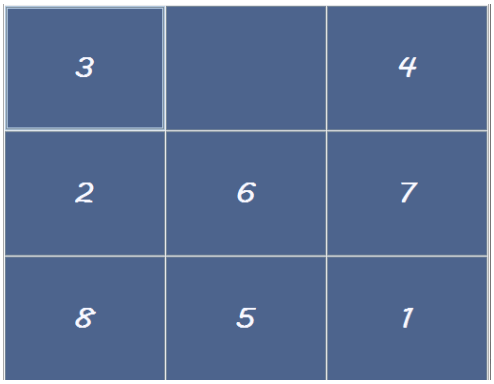


Ảnh 2: Khi bạn chiến thắng

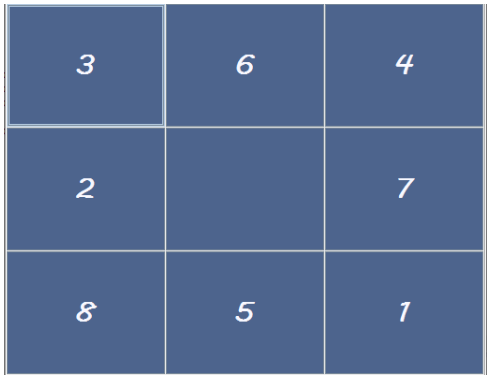
Đối với các trường hợp khác của N puzzle như 4*4 hay 5*5,… thì cách chơi cũng giống nhau ta chỉ chiến thắng khi các ô xếp theo kiểu lợp ngói (ảnh 2).

Với game **N puzzle** ta sẽ có những ý tưởng để xây dựng bài toán như sau:

- Do game gồm các ô nên ta sẽ tạo ra một lưới các ô (ví dụ Girdlayout, …). Tại mỗi ô sẽ là 1 giá trị từ 1 đến n*n-1.
- Vì lí do các ô đều hình vuông và game là 1 bảng n*n lên ta có thể thao tác các bước tạo game trên mảng 2 chiều với n dòng và n cột. Khi đó để gán giá trị cho mỗi ô ta sẽ dùng 2 vòng lặp for lồng nhau để gán giá trị cho ô. Trong đó để dễ dàng ô trống ta sẽ cho giá trị bằng 0 (sau ta sẽ thay đổi sau).
- Về thuật toán cách giải tại mỗi lần click 1 ô bất kì (khác ô trống) nếu nó không cạnh ô trống thì sẽ không thay đổi gì và ngược lại ô chúng ta click sẽ tự động rơi xuống ô trống và ô trống sẽ lên vị trí ô ta click (để hình dung dễ hiểu bạn có thể tưởng tượng 2 ô đó sẽ hoán đổi vị trí).



Ảnh 3: Trước khi di chuyển



Ảnh 4: Sau khi nhấn vào ô có giá trị 6

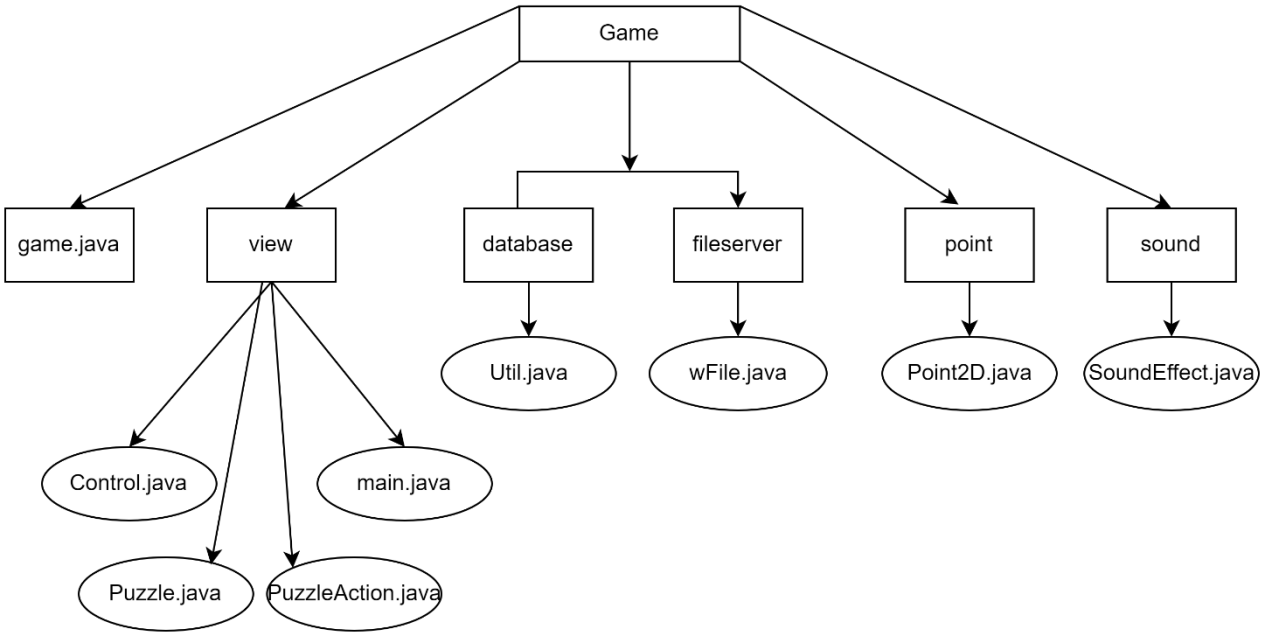
Như đã nói trường hợp bạn thắng khi các ô có giá trị lợp gói tại đây chúng ta sẽ có 1 hàm để kiểm tra (checkWin) nếu bạn thắng thì trò chơi sẽ thông báo "You Won" và ngược lại trò chơi sẽ tiếp tục. Trong trường hợp bạn thắng trò chơi sẽ kết thúc và khi ta bắt đầu lại ta sẽ có game với cách sắp xếp ô mới.

Ý tưởng bài toán cơ bản sẽ là như trên, và phần quan trọng của bài toán làm game này đó là làm về giao diện người dùng (UI) và các sự kiện (Event) để người chơi có thể thao tác với chương trình trò chơi. Trong suốt bản báo cáo sẽ có những phần sẽ thay đổi hoặc biến đổi thêm nhưng cơ bản chúng sẽ đều tuân theo luật chơi của bài toán.

II. Thao tác thực hiện

Đối với Java thì Java sẽ cung cấp các thư viện và các công cụ giúp cho người dùng dễ dàng thiết kế giao diện và các điều khiển khác. Trong bản báo này, để thiết kế được game chúng em sẽ sử dụng đó là Java Swing, Java AWT dùng để hỗ trợ tạo giao diện đồ hoạ người dùng, ngoài ra còn có lệnh sự kiện Event trong Java.

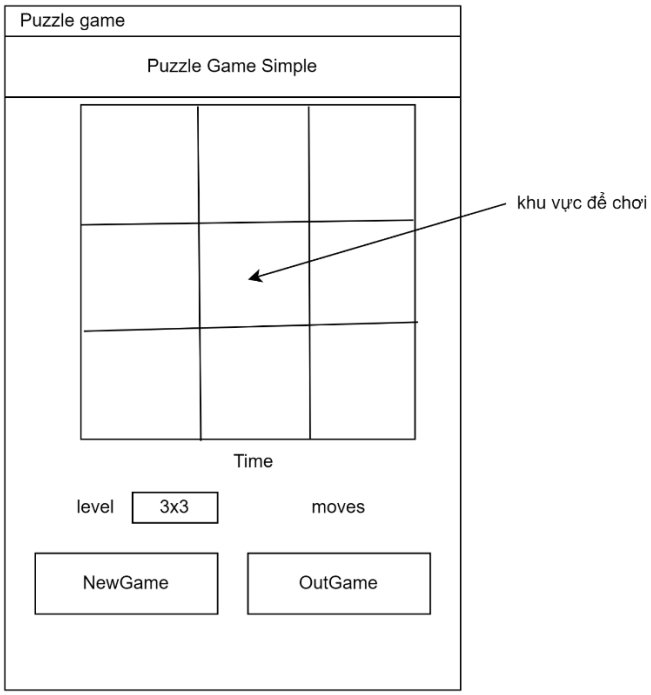
Đối với bài toán tạo game **N puzzle** ta sẽ tạo ra 1 package với tên Game để phân biệt với các package của chương trình khác. Và tại package Game ta lại có các package con khác trong đó thao tác chính sẽ được thực hiện trong package view. Ta sẽ có sơ đồ cấu trúc file như sau:



1. Tạo giao diện game

Như đã nói ở phần trước để thực hiện tạo giao diện ta sẽ tạo ra 1 khung để chứa tất cả màn hình của 1 game. Trước đó ta sẽ tạo ra 1 màn hình khởi động để bắt đầu vào game. Màn hình này sẽ được khảo tạo bởi 1 Jframe , tại đó ta sẽ thiết lập 1 số thông số cơ bản như vị trí hiện trên màn hình, tiêu đề , logo,…

Ta sẽ tạo 1 class tên ***puzzle*** để chứa giao diện màn hình trò chơi. Sơ đồ ban đầu của game có dạng:



Ngoài ra ta sẽ tạo 1 số đối tượng sẽ có trong game:

- Các button tạo bởi Jbutton chứa lệnh chơi game mới (NewGame), thoát game (Outgame).

```
JButton ButtonOut = new JButton("OutGame");
JButton ButtonNew = new JButton("NewGame");
```

- Các label tạo bởi Jlabel để tạo một vùng chứa văn bản. Các văn bản cần chứa như: số bước di chuyển (moves), thời gian xong trò chơi (time), mức độ chơi (level).

```
JLabel fJLabel1 = new JLabel("Moves");
JLabel fJLabel2 = new JLabel("Level");
JLabel fJLabel3 = new JLabel("Time:");  JLabel timeLabel = new JLabel("00:00:00");
```

- Các label để đặt tên (title) và xác định kiểu biểu diễn chính là flowlayout
-
```
JLabel title = new JLabel("PUZZLE GAME SIMPLE");
```

- Một combobox để chứa các chế độ chơi (3x3, 4x4,…)

```
JComboBox<String> ComboBox = new JComboBox<>();
```

- Và quan trọng nhất là tạo ra các Panel dùng để chứa các phần tử khác và hỗ trợ định danh các phần tử sao cho hợp lý

```
JPanel jPanel1 = new JPanel();//khu vực các ô
JPanel s2Panel = new JPanel();// chứa 2 button newgame và outgame
JPanel SPanel = new JPanel();// khu vực chứa s1Panel, s2Panel và sPanel
JPanel mPanel = new JPanel();// panel chính chứa tất cả các panel khác
JPanel sPanel = new JPanel();// khu vực chứa dòng time
JPanel s1Panel = new JPanel(); // khu vực chứa level và moves
JLabel JTextCount = new JLabel();// khu vực chứa giá trị lượt di chuyển
```

Sau khi ta tạo được 1 số thông tin (về việc đặt tên ta sẽ tự lựa chọn theo ý muốn) trên ta sẽ bố trí lại các giao thông tin giao diện sao cho ổn nhất:

- Dùng phương thức setDefaultCloseOperation để đóng giao diện; setResizable (boolean b) quy định xem Jframe có thay đổi được kích thước không; setVisible(boolean b) để đặt JFrame ẩn/hiện; setLocationRelativeTo(null) để cửa sổ hiện ra giữa màn hình; setHorizontalAlignment căn chỉnh các Jlabel…
- Dùng phương thức setFont để quy định phông chữ cho văn bản; phương thức setBackground để tạo màu cho giao diện; setForeground để đặt màu cho chữ; setFocusPainted để cho các button mềm mại hơn…

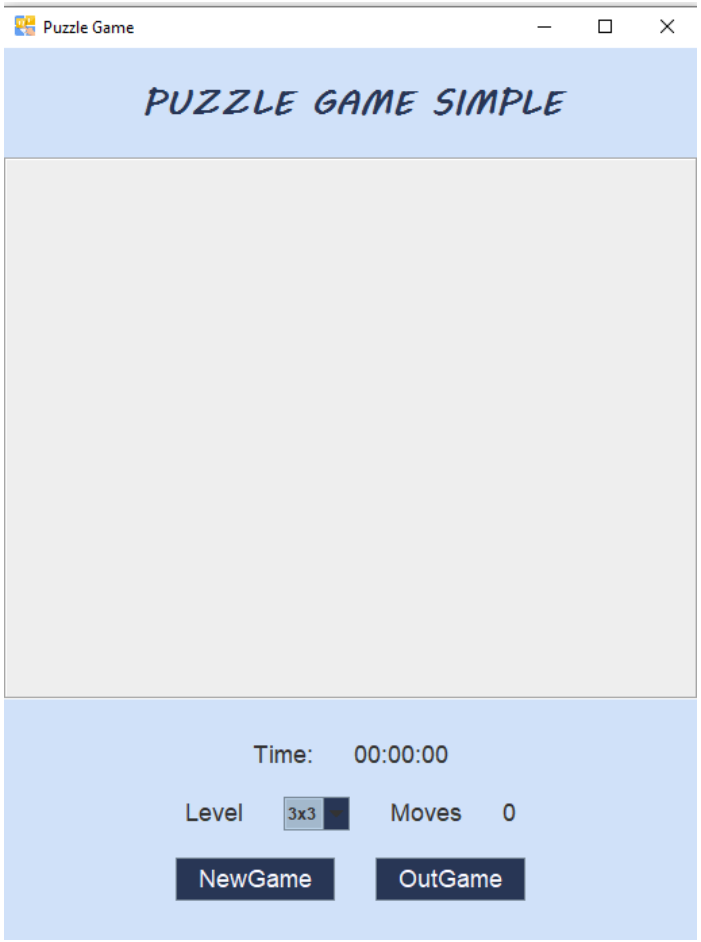
Tại đây ta có thể dùng thêm các phương thức khác để cho giao diện thân thiện nhất có thể. Sau khi tạo ra các đối tượng cơ bản thì ta tiến hàng thêm các đối tượng đó vô Jpanel bằng phương thức add(tên đối tượng). VD:

```
s1Panel.setLayout(flowLayout);
s1Panel.add(fJLabel2);
s1Panel.add(ComboBox);
s1Panel.add(fJLabel1);
s1Panel.add(JTextCount);
```

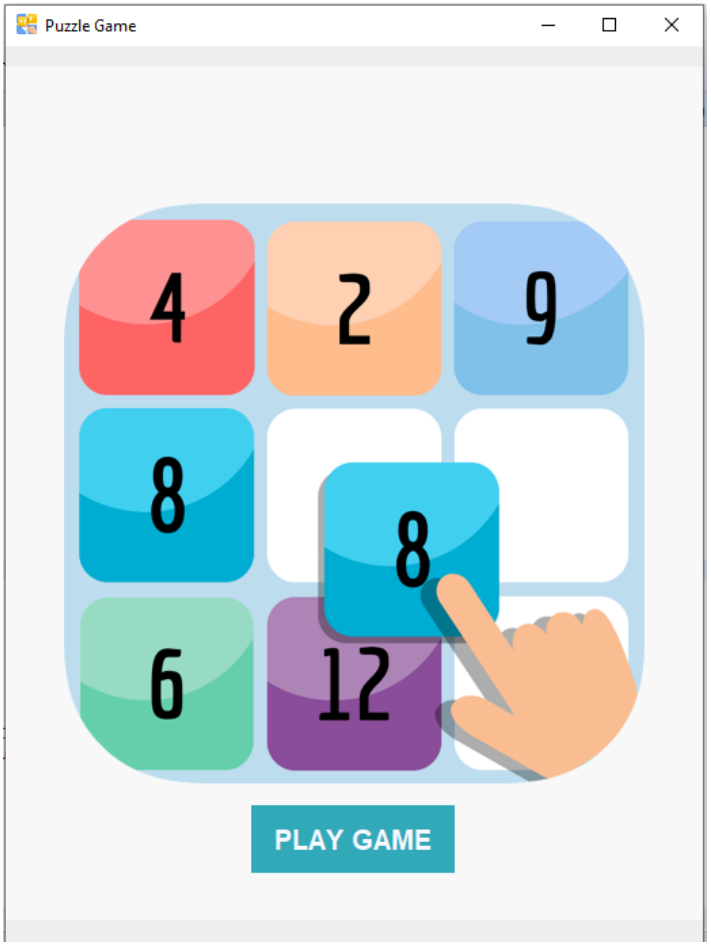
Thêm biểu tượng của game bằng lệnh(this chính là giao diện màn hình) :

```
this.setIconImage(new ImageIcon("Pic/logo.png").getImage());
```

Sau khi tạo và thêm các đối tượng đó ta sẽ có bộ khung của game như sau (màu sắc và phông chữ tuỳ ý thích)



Ảnh 5: màn hình giao diện chơi game



Ảnh 6: màn hình ngoài khi khi vào game

Ngoài ra ta có thể làm thêm 1 màn hình khác khi vào game (hình 6) để ki bắt đầu giao diện sẽ đẹp hơn. Và giao diện này (hình 6) ta sẽ tạo 1 class mới là ***main*** để lưu chúng. Tương tự như làm giao diện chơi game tại class main ta sẽ tạo ra 1 label để đặt ảnh và 1 button để chứ văn bản (StartGame). Về phần định dạng ta có thể dùng các định dạng phông chữ như cũ(hoặc bổ sung thêm), để thêm ảnh vô giao diện ta có thể dùng lệnh như :

```
this.setIconImage(new ImageIcon("Pic/logo.png").getImage());
```

2. Các thao tác điều khiển

Sau khi tạo giao diện tiếp theo ta đến bước thực hiện các giao tiếp tạo nên lõi hoạt động của game. Để tạo các lệnh điều khiển này ta sẽ tạo 1 class mới có tên là ***control.***

Để có thể kết nối giao diện và hệ thống điều khiển ta sẽ cần gọi giao diện ra bằng:

```
public Control(Puzzle pz) {
    this.pz = pz;
    add();
}
```

Trong đó hàm add() là hàm thêm các giá trị của ô và sẽ được trình bày dưới đây.

Trước tiên để game có thể hoạt động bình thường ta sẽ tạo ra các ô chứa các giá trị từ 1 đến n*n-1. Để lấy được kích thước n ta sẽ lấy dữ liệu từ combobox đã tạo từ phần trên tuy nhiên dữ liệu trên combobox ta đang để kiểu string do đó ta cần chuyển về kiểu số nguyên để lấy cỡ của bảng ô. Ta có thể làm như sau:

```
String s = (String) pz.ComboBox.getSelectedItem(); // lay du lieu trong combobox
System.out.println(s);
String[] Out = s.split("x"); // tach chuoi
SIZE = Integer.parseInt(Out[0]);// chuyen tu string sang int
```

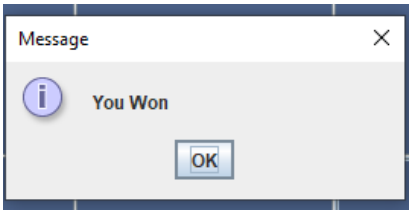
Và như đã nêu ý tưởng ở phần trước để thêm giá trị cho các ô ta sẽ dùng 2 vòng lặp với các giá trị i=j=n cùng với mảng 2 chiều các button có kích cỡ [n][n].

```
pz.jPanel1.removeAll();//xoá các thông tin trước nếu có
pz.jPanel1.setLayout(new GridLayout(SIZE, SIZE, 1, 1));// tạo layout kích ỡ size*size
pz.jPanel1.setPreferredSize(new Dimension(SIZE * 60, SIZE * 60));
```

```
matrix = new JButton[SIZE][SIZE];
for (int i = 0; i < SIZE; i++) {
    for (int j = 0; j < SIZE; j++) {
        JButton btn = new JButton(i * SIZE + j + 1 + "");// gán các giá trị cho từng button
        btn.setFont(new Font("MV Boli", Font.BOLD, 30));//cài phông chữ cho giá trị
        btn.setBackground(ColorButton);//cài màu sắc cho button
        btn.setFocusPainted(false);
        btn.setForeground(new Color(248, 248, 255));// đặt màu cho chữ/số
        matrix[i][j] = btn;//cho vào mảng 2 chiều
        pz.jPanel1.add(btn);// gọi đến giao diện ô để thêm các ô vào panel
```

Khi lấy được kích cỡ ô thì công việc tiếp theo ta sẽ di chuyển các ô(moveButton) và kiểm tra các ô khi di chuyển (checkMove) nếu chiến thắng (checkWin) thì ta sẽ in 1 thông báo (message).

```
btn.addActionListener(new ActionListener() { //thêm sự kiện cho các ô

    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        if (isStartGame) {
            if (isStartGame1 != 0) {
                if (checkMove(btn)) {
                    moveButton(btn);
                    if (checkWin()) {
                        // started = flase;
                        stop();
                        JOptionPane.showMessageDialog(null, "You Won");
                        isStartGame = false;

                        System.out.println(pz.timeLabel.getText());//in thời gian trên console
                        System.out.println(pz.JTextCount.getText());//in số bước trên console
```



Khi ta nhấn ok ta sẽ quay trở lại màn hình chơi game lúc đầu (các ô xếp theo kiểu lợp ngói) và tại đó ta lại tiếp tục chơi game như lúc đầu (newGame)

Tất cả những thao tác trên ta sẽ cho vào 1 hàm gọi là add().

Việc kiểm tra checkWin hay moveButton,… chỉ được thực hiện khi đã bắt đầu và theo lần lượt các bước checkMove → moveButton → checkWin.

Với ý tưởng như trên ta cần xây dựng các hàm như di chuyển các ô (moveButton), kiểm tra ô có di chuyển được không (checkmove), kiểm tra chiến thắng(checkWin).

- Hàm kiểm tra ô có di chuyển được hay không (checkMove)

Để kiểm tra 1 ô có di chuyển được hay không thì ta phải kiểm tra vị trí ô trống ở đâu do đó ta cần một hàm phụ trợ để tìm vị trí (toạ độ) ô trống (getEmptyPos())

Vì thuộc tính ô ta cần toạ độ (vị trí) nên ta có thể lấy vị trí ô theo điểm (Point). Với đối tượng điểm (object) ta có thể dùng bằng Point của java.awt hoặc tạo hàm Point để lấy vị trí điểm. Tại lúc này nhóm sẽ lựa chọn tạo lớp Point2D (lớp tự tạo các bạn có thể tham khảo ở cuối bản báo cáo) để lấy toạ độ i,j của điểm. Kết quả trả về sẽ là vị trí i,j của điểm

```
public Point2D getEmptyPos() {
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            if (matrix[i][j].getText().equals("")) {
                return new Point2D(i, j);
            }
        }
    }
    return null;
}
```

Để kiểm tra xem ô có di chuyển được hay không ta sẽ kiểm tra vị trí của ô rỗng và vị trí của button ta click vào nếu 2 ô liền kề (x1-x2=1 hoặc y1-y2=1) thì ô button đó có thể di chuyển.

```
public boolean checkMove(JButton button) {
    if (button.getText().equals("")) {
        return false;
    }
    Point2D p = getEmptyPos();
    Point2D clickedPoint = null;

    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            if (matrix[i][j].getText().equals(button.getText())) {
                clickedPoint = new Point2D(i, j);
            }
        }
    }

    if (p.getX() == clickedPoint.getX() && Math.abs(p.getY() - clickedPoint.getY()) == 1) {
        return true;
    }

    if (p.getY() == clickedPoint.getY() && Math.abs(p.getX() - clickedPoint.getX()) == 1) {
        return true;
    }
    return false;
}
```

- Hàm di chuyển ô (moveButton)

Sau khi ta kiểm tra nếu ô button đó di chuyển được ta tiến hành đổi giá trị của 2 ô button đó cho nhau.

```
public void moveButton(JButton button) {
    Point2D p = getEmptyPos();
    matrix[p.getX()][p.getY()].setText(button.getText());
    button.setText("");
    move1++;
    pz.JTextCount.setText(move1 + "");}
```

Ở đây có giá trị move1 đây chính là giá trị thể hiện số lần ta di chuyển (click) vào ô button. Giá trị này sẽ được thể hiện ở label theo như phần giao diện đã thiết kế.

- Hàm kiểm tra chiến thắng (checkWin)

Để chiến thắng thì điều kiện đầu tiên là ô button cuối phải là ô rỗng

Điều kiện thứ 2 là vị trí các ô button phải theo kiểu lợp gói.

Muốn kiểm tra ta sẽ dùng hàm equals(“”) để so sánh nếu điều kiện 1 đúng ta sẽ đến điều kiện 2 , nếu điều kiện 2 chỉ cần 1 ô sai thì checkWin sẽ trả về false luôn.

```
public boolean checkWin() {
    if (matrix[SIZE - 1][SIZE - 1].getText().equals("")) {
        for (int i = 0; i < SIZE; i++) {
            for (int j = 0; j < SIZE; j++) {
                if (i == SIZE - 1 && j == SIZE - 1) {
                    return true;
                }
                if (!matrix[i][j].getText().equals(i * 3 + j + 1 + "")) {
                    return false;
                }
            }
        }
        return true;
    }
    return false;
```

Sau khi kết thúc trò chơi để bắt đầu cho chơi mới ta cần phải xoá hết các thông tin (Time,moves0 và tiến hành trộn button lại (mixButton). Đối với hàm trộn lại button (hay nói cách khác là thay đổi giá trị các ô button) ta sẽ tiến hành lựa chọn ngẫu nhiên các trường hợp để trộn (thay đổi vị trí giá trị) theo kiểu trên thay dưới, trái thay phải,… và ta càng nhiều lần lặp trường hợp các giá trị thay đổi càng không trùng nhau.

```
public void mixButton() {
    for (int k = 0; k < 500; k++) {
        Point2D p = getEmptyPos();
        Random r = new Random();
        int choice = r.nextInt(4);
        switch (choice) {
        case 0:
            if (p.getX() > 0) {
                matrix[p.getX()][p.getY()].setText(matrix[p.getX() - 1][p.getY()].getText());
                matrix[p.getX() - 1][p.getY()].setText("");
            }
            break;
        case 1:
            if (p.getY() < SIZE - 1) {
                matrix[p.getX()][p.getY()].setText(matrix[p.getX()][p.getY() + 1].getText());
                matrix[p.getX()][p.getY() + 1].setText("");
            }
            break;
        case 2:
            if (p.getX() < SIZE - 1) {
                matrix[p.getX()][p.getY()].setText(matrix[p.getX() + 1][p.getY()].getText());
                matrix[p.getX() + 1][p.getY()].setText("");
            }
            break;
        case 3:
            if (p.getY() > 0) {
                matrix[p.getX()][p.getY()].setText(matrix[p.getX()][p.getY() - 1].getText());
                matrix[p.getX()][p.getY() - 1].setText("");
            }
            break;
        }
    }
}
```

3. Các thao tác tính điểm

Như đã trình bày ở giao diện ta có các thông số cần lấy sau khi kết thúc trò chơi như thời gian chiến thắng, số lượt di chuyển nên ta cần phải định dạng kiểu thông tin cho phù hợp. Ví dụ tính thời gian theo 00:00:00.

Trước tiên ta có hàm thời gian chơi. Để tính thời gian ta sẽ dùng thư viện Timer để đếm thời gian trôi qua theo giây (s) đồng thời ta cần khởi tạo các biến ban đầu (giây, phút,..)

```
int elapsedTime = 0; //thời gian trôi qua
int seconds = 0;
int minutes = 0;
int hours = 0;
Timer time;
String seconds_string = String.format("%02d", seconds);
String minutes_string = String.format("%02d", minutes);
String hours_string = String.format("%02d", hours);
```

Tiếp theo đó là hàm ép kiểu số thành chữ (string) để hiện trên giao diện (timelabel)

Vậy thời gian được tính như thế nào ta sẽ tính thời gian khi mà người dùng nhấn bắt đầu và số lượt di chuyển cũng như các trạng tháu của ô về vị trí ban đầu (move=0 và giá trị ô đang ở trạng thái thắng)

```
public void StartGame() {

    time = new javax.swing.Timer(1000, new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            elapsedTime = elapsedTime + 1000;
            hours = (elapsedTime / 3600000);
            minutes = (elapsedTime / 60000) % 60;
            seconds = (elapsedTime / 1000) % 60;
            seconds_string = String.format("%02d", seconds);
            minutes_string = String.format("%02d", minutes);
            hours_string = String.format("%02d", hours);
            pz.timeLabel.setText(hours_string + ":" + minutes_string + ":" + seconds_string);
        }
    });
    if (started == false) {
        started = true;
        pz.ButtonNew.setText("Reset");
        start();
    } else {
        started = false;
        // pz.ButtonNew.setText("Start");
        // reset();

    }
    this.isStartGame1 = 1;
    isStartGame = true;
    move1 = 0;
    pz.JTextCount.setText("0");
}
```

Sau khi chiến thắng một việc cần làm nữa là ta cần phải đặt lại các giá trị và ta có hàm resetgame()(quay về trang thái chiến thắng) và newgame()(game mới với các ô đã được tráo.

```
public void ResetGame() {
    add();
    reset();
    move1 = 0;
    pz.JTextCount.setText("0");
    pz.ButtonNew.setText("NewGame");
    this.isStartGame1 = 0;
    started = false;
}
```

```
//game mới
public void NewGame() {
    add();
    mixButton();
    // move1 = 0;
    // pz.JTextCount.setText("0");
    // pz.ButtonNew.setText("Start");
    pz.ButtonNew.setText("Reset");
    // reset();
    this.isStartGame1 = 0;
}
```

Tại đây ta lại có 1 hàm reset(), hàm này sẽ trả về các giá trị 0 ban đầu cho thời gian hiện trên Label timelabel

```
void reset() {
    stop();
    elapsedTime = 0;
    seconds = 0;
    minutes = 0;
    hours = 0;
    seconds_string = String.format("%02d", seconds);
    minutes_string = String.format("%02d", minutes);
    hours_string = String.format("%02d", hours);
    pz.timeLabel.setText(hours_string + ":" + minutes_string + ":" + seconds_string);
}
```

Ta cũng có hàm stop() và start() dùng để tính xác định thời gian kết thúc và bắt đầu.

```
void start() {
    time.start();
}
```

```
void stop() {
    if (elapsedTime != 0) {
        time.stop();

    }
}
```

4. Một số thao tác khác

Sau khi giành chiến thắng và kết thúc lượt chơi , ta sẽ cần 1 nơi lưu thông tin của lượt chơi đó và để lưu ta có thể lựa chọn lưu trên cơ sở dữ liệu SQL SERVER hoặc xuất ra file txt để lưu trên máy tính.

- Với cơ sở dữ liệu SQL SERVER ta sẽ có lop Util để tạo các thao tác (phương thức). Trước tiên ta sẽ tạo ra kết nối của SQL với JDBC của Java .

```
public class Util {

    public static String DRIVER_PATH = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
    public static final String DB_URL = "jdbc:sqlserver://DESKTOP-UCIH129\\VANH:1433;databaseName=BTLJAVA";
    public Puzzle pz;

    public static Connection dbConn() {

        Connection conn = null;
        // load driver
        try {
            Class.forName(DRIVER_PATH);

            conn = DriverManager.getConnection(DB_URL, "sa", "123");
            System.out.println("OK - Connection is created! " + conn);
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        return conn;
    }
```

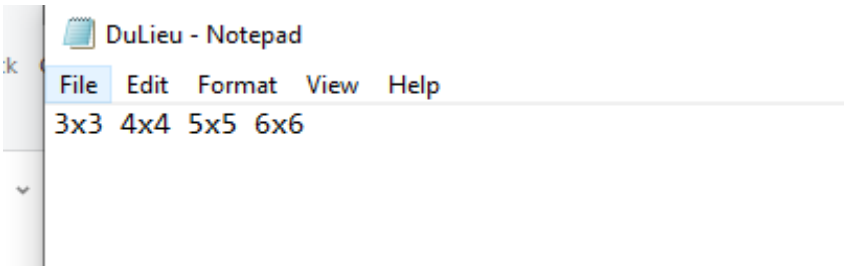
Sau khi kết nối hoàn thành ta tiến hành đưa kết quả lên SQL SERVER. Trong đó giá trị đưa lên sẽ là thời gian chơi (time) và số lượt di chuyển (moves).

```
public static void insert(String time, String Count) {
    Connection c = dbConn();

    int C = Integer.parseInt(Count);
    try {
        String sql = "INSERT INTO result(Tg, Clicks) VALUES(?,?);";
        PreparedStatement pt = c.prepareStatement(sql);
        pt.setString(1, time);
        pt.setInt(2, C);
        pt.executeUpdate();
        pt.close();

    } catch (Exception e) {
        // TODO: handle exception
    }
    try {
        c.close();
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

}
```

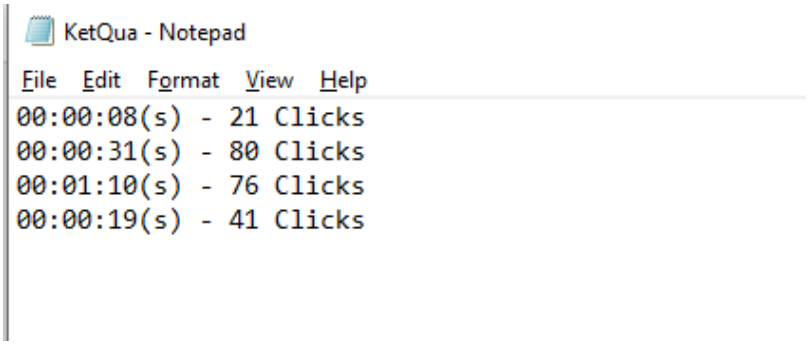
Dưới đây là hàm lấy dữ liệu từ sql vào arraylist để add dữ liệu vào combobox

```java
public static ArrayList<String> loadDataToCombobox() {
    Statement st = null;
    ArrayList<String> List = new ArrayList<>();
    Connection c = dbConn();

    try {
        st = c.createStatement();
        // Step 4: Communication with MYSQL via: SQL.
        String sql = "Select ID from Combobox";
        ResultSet rs = st.executeQuery(sql);

        while (rs.next()) {
            String id = rs.getNString("ID");
            List.add(id);
            System.out.println(id);
        }

    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } finally {
        // Step 5: close connection
        try {
            st.close();
            c.close();
            System.out.println("OK - Statement and connection is closed.");
        } catch (SQLException | NullPointerException e) {
            // N.A
        }
    }

    return List;
}
```

- Đối với file txt ta sẽ đọc từ file và cũng ghi kết quả vào 1 file khác (ở đây là file ketqua.txt)



File dữ liệu chứa các các kích thước trò chơi 3x3,4x4,..
File ketqua.txt chứa thười gian và số bước đi cho đến khi chiến thắng



```
public static ArrayList<String> ReadFile(){
    ArrayList<String> List = new ArrayList<>();
    File fileIn = new File("DuLieu.txt");

    FileReader fr = null;
    BufferedReader br = null;
    try {
        fr = new FileReader(fileIn);
        br = new BufferedReader(fr);

        String line;
        while ((line = br.readLine()) != null) {

            //System.out.println(line);
            String[] strs = line.split(" ");
            for (String string : strs) {
                List.add(string);
            }
        }
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {

        try {
            br.close();
            fr.close();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    return List;
}
```

```
//========== Ghi dữ liệu lên file txt===============

public static void writeFile(String time, String Count) {

    File fileOut = null;

    try {
        fileOut = new File("KetQua.txt");
        FileWriter fw = new FileWriter(fileOut, true);
        // BufferedWriter writer give better performance
        BufferedWriter bw = new BufferedWriter(fw);
        bw.write(time + "(s) - " + Count + " Clicks \n");
        // Closing BufferedWriter Stream
        bw.close();
    } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public static void main(String[] args) {
    // TODO Auto-generated method stub
    ArrayList<String> a = ReadFile();
    for (String string : a) {
        System.out.println(string);
    }
}
```

Sau khi mà đã hoàn thành trò chơi thì thơi gian và số lần di chuyển sẽ đc đưa lên sql (và file KetQua.txt)

```
                                //extracted();
                                WirteFile();

                            }
                        }
                    }
                }
                // else {
                // JOptionPane.showMessageDialog(null, "Press NewGame to
                // play");
                // }
            }
            /* ====== Đưa dữ liệu chiến thắng lên SQL sever*/
'/          private void extracted() {
'/              Util.insert(pz.timeLabel.getText(), pz.JTextCount.getText());
'/          }
            /* ========== Đưa dữ liệu chiến thắng gồm thời gian và số lần clicks vào file txt*/
            private void WirteFile() {
                wFile.writeFile(pz.timeLabel.getText(), pz.JTextCount.getText());
            }
        });
    }
}
```

Trên sql server ta sẽ cần tạo ra bảng để lưu dữ liệu:

```sql
create database BTLJAVA
go
create table Combobox
(
    ID nvarchar(20) not null
    CONSTRAINT PK_Combobox_ID PRIMARY KEY,
    Name nvarchar(20) not null
)
create table result
(
    Tg nvarchar(20),
    Clicks int
    CONSTRAINT PK_result_Clicks PRIMARY KEY(Tg,Clicks)
)
insert into Combobox
values
(N'3x3',N'Muc de'),
(N'4x4',N'Muc trung bình'),
(N'5x5',N'Muc khó'),
(N'6x6',N'Muc cuc khó')

select * from Combobox
select * from result
```

Tiếp theo để cho game thêm sôi động ta có thể thêm file âm thanh cho game. Tại đây ta sẽ có thêm package sound và lớp soundeffect để thực hiện (tham khảo):

```java
public class SoundEffect {

    Clip clip;

    public void setFile(String soundFile) {
        try {
            File file = new File(soundFile);
            AudioInputStream Au = AudioSystem.getAudioInputStream(file);
            clip = AudioSystem.getClip();
            clip.open(Au);


        } catch (Exception e) {
            // TODO: handle exception
        }
    }

    public void play() {
        // clip.setFramePosition(0);
        clip.start();
    }

    public void loop() {
        clip.loop(Clip.LOOP_CONTINUOUSLY);
    }

    public void Stopsound() {
        clip.stop();
    }

}
```

Ngoài thêm nhạc nền ở trên ta có thể thêm âm thanh khi di chuyển , khi đó âm thanh sẽ thêm ở hàm movebutton() của lớp control

```
String soundMove = "move.wav";
Sound.setFile(soundMove);
Sound.play();
```

Ta có thêm các message khi chiến thắng hoặc thoát ra tại bởi showMessageDialog của JoptionPane . Trước đó ta phải kế liên kết giao diện với các button để khi thực hiện nhấn button ta sẽ có các thông báo khác nhau. Và ta sẽ có lớp PuzzleAction để lưu thao tác để thoát game.

```
public class PuzzleAction implements ActionListener {

    private Puzzle pz;
    Control c;
    main M = new main();
    JColorChooser colorChoose;

    public PuzzleAction(Puzzle pz) {
        super();
        this.pz = pz;
    }

    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        String cmd = e.getActionCommand();

        switch (cmd) {
        // case "Start":
        case "Reset":
            // pz.getC().StartGame();
            pz.c.ResetGame();
            break;
        case "OutGame":
            int confirmed = JOptionPane.showConfirmDialog(null, "Do you want to Exit", "Exit",
                    JOptionPane.YES_NO_OPTION);
            if (confirmed == JOptionPane.YES_OPTION) {
                System.exit(0);
            }
            break;
        case "NewGame":
            pz.c.NewGame();
            pz.c.StartGame();

            break;
        default:
            break;
        }
    }
```

III. Kết quả

Sau khi ta thực hiện các bước trên ta sẽ có giao diện game (có thể chọn 3x3,4x4..). Việc từ giao diện khởi động đến lúc chơi được thực hiện khi ta click vào button Playgame thì sẽ ẩn Jframe cũ (màn khởi động) và hiện ra Jframe mới (giao diện chơi)

IV. Đóng góp ý kiến

Sau khi thực hiện các thao tác tạo game **N Puzzle** về cơ bản chúng ta đã hoàn toàn có thể chơi và sử dụng được chúng. Tuy nhiên tại bản báo cáo này cần rút ra 1 số vấn đề cần được giải quyết như:

- Chưa thể đưa ngược lại dữ liệu trên SQL SERVER về để tạo sắp xếp và xếp hạng người chơi nhanh nhất.
- Cấu trúc các đối tượng, thuộc tính còn không chặt chẽ, gây lộn xộn hoặc nhầm lẫn trong quá gọi hàm, thuộc tính,…
- Sắp xếp chưa chặt chẽ giữa các phần(hàm) chưa chuẩn, nhiều thuộc tính(biến) còn thừa hoặc dùng không đủ với mục đích của chúng,..
- Giao diện vẫn còn theo kiểu máy móc chưa hoàn toàn tạo sự mềm mại cho người dùng
- …..

Tuy còn một số vấn đề cơ bản nhưng rất cảm ơn các thành viên đã cùng nhau làm và tạo bản báo cáo này. Nếu thầy(các bạn) chưa hoàn toàn có thể hiểu kĩ thông tin cũng như bố cục các phần thì có thể tham khảo trên đường dẫn sau:

https://drive.google.com/drive/folders/1VlL2kaDgiBj72m8dzaDUQWwZ792K_LMw?usp=sharing

https://github.com/bgbghvhv/N-Puzzle.git

Tài liệu tham khảo

1. *https://helpex.vn/question*. (2022, 03 30). Được truy lục từ helpex.vn: https://helpex.vn/question/lam-cach-nao-de-phat-am-thanh-trong-java-5cbbb237ae03f60a1ccdbff3

2. *https://stackoverflow.com/questions*. (2022, 03 30). Retrieved from https://stackoverflow.com: https://stackoverflow.com/questions/19047414/java-program-to-connect-to-sql-server-and-running-the-sample-query-from-eclipse

3. Phúc, N. T. (2022, 03 15). Công nghệ Java. *Tài liệu môn công nghệ Java*. Hà Nội, Hoàn thành, Việt Nam.