Báo cáo thu hoạch bài tập lớn

**- Xử lý thắng thua, logic game**

Vi du: private final String easyPuzzle =

"360000000004230800000004200" +

"070460003820000014500013020" +

"001900000007048300000000045";

Ta dùng một mảng gồm 81 phần tử lưu các giá trị trong bàn cờ. Và việc xử lý logic game thông qua việc xử lý mảng đó. Các số khác 0 sẽ được cho hiện lên trên màn hình còn bằng 0 thì không cho xuất hiện trên màn hình, tạo thành một bàn chơi gồm 81 ô. Khi chèn một số vào tại vị trí có giá trị bằng 0 thì sẽ lưu giá trị lại, nếu như tất cả các số được chèn hết , đồng nghĩa với việc không còn giá trị 0 trong mảng nữa thì sẽ xuất hiện một bảng thông báo thắng game.

**public** **void** WinGame()

{

**int** dem=0;

**for**(**int** i:puzzle)

**if**(i==0) dem++;

**if**(dem==0)

{

AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);

//Thiết lập tiêu đề hiển thị

builder.setTitle("Thong bao");

//Thiết lập thông báo hiển thị

builder.setMessage("Chuc mung ban da thang! ");

builder.setCancelable(**false**);

//Tạo nút

builder.setNeutralButton("Xem lai", **null**);

builder.setPositiveButton("Quay lai", **new** DialogInterface. OnClickListener() {

@Override

**public** **void** onClick(DialogInterface dialog, **int** which)

{

Intent intent = **new** Intent(Game.**this**,MainActivity.**class**);

startActivity(intent);

}});

builder.create().show();

}

}

Ta sử dụng thêm một mảng 3 chiều để lưu trữ tất cả các số đã xuất hiện trong bảng game

Viết một vòng lặp để quét tất cả các vị trí trong bảng gồm 81 ô.

**private** **final** **int** used[][][] = **new** **int**[9][9][];

**protected** **int**[] getUsedTiles(**int** x, **int** y) {

**return** used[x][y];

}

**private** **void** calculateUsedTiles() {

**for** (**int** x=0; x<9; x++) {

**for** (**int** y=0; y<9; y++) {

used[x][y] = calculateUsedTiles(x, y);

}

}

}

**private** **int**[] calculateUsedTiles(**int** x, **int** y) {

**int** c[] = **new** **int**[9];

// horizontal

**for** (**int** i=0; i<9; i++) {

**if** (i==y)

**continue**;

**int** t = getTile(x, i);

**if** (t != 0)

c[t-1] = t;

}

// vertical

**for** (**int** i=0; i<9; i++) {

**if** (i==x)

**continue**;

**int** t = getTile(i, y);

**if** (t != 0)

c[t-1] = t;

}

// same cell block

**int** startx = (x/3) \* 3;

**int** starty = (y/3) \* 3;

**for** (**int** i=startx; i<startx+3; i++) {

**for** (**int** j=starty; j<starty+3; j++) {

**if** (i==x && j==y)

**continue**;

**int** t = getTile(i, j);

**if** (t != 0)

c[t-1] = t;

}

}

// compress

**int** nused = 0;

**for** (**int** t: c) {

**if** (t != 0)

nused++;

}

**int** c1[] = **new** **int**[nused];

nused = 0;

**for** (**int** t: c) {

**if** (t!=0)

c1[nused++] = t;

}

**return** c1;

}

}

Tại mỗi một vị trí sẽ lưu tất cả các số đã xuất hiện theo chiều dọc , ngang, và phân vùng 3x3 chứa ô đang được trỏ tới đó gồm các số từ 1 đến 9. Nếu như đã có đủ 9 số thì ta sẽ không thể chèn thêm bất kì số nào khác thêm vào vị trí đó được nữa. Ngược lại sẽ xuất hiện một bàn phím ảo để mình chọn số chèn vào trong ô đó. Cứ lần lượt chèn cho đến khi không còn ô trống nào nữa thì sẽ thắng game.

**protected** **void** showKeypadOrError(**int** x, **int** y) {

**int** tiles[];

**if**(getOriTile(x,y)==0)

{

tiles = getUsedTiles(x, y);

**if** (tiles.length == 9) {

Toast toast = Toast.*makeText*(**this**,"không còn số có thể điền vào", Toast.***LENGTH\_SHORT***);

toast.setGravity(Gravity.***CENTER***, 0, 0);

toast.show();

}

**else** {

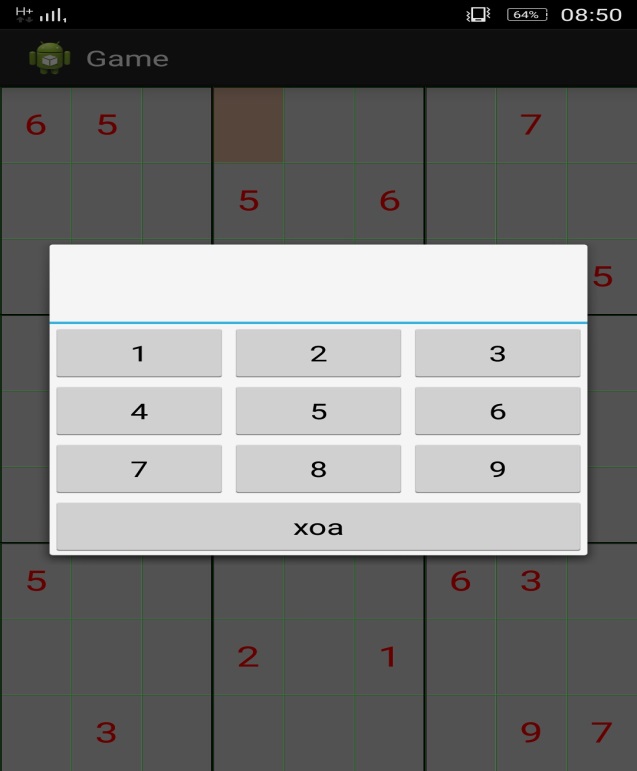
Dialog v= **new** KeyPad(**this**,tiles, puzzleView);

v.show();

}

}

}



Kiểm tra xem số định nhập vào đã xuất hiện chưa, nếu chưa thì sẽ chèn vào mảng, đồng thời kiểm tra điều kiện thắng game, ngược lại xuất thông báo lỗi nhập

**public** **boolean** setTileIfValid(**int** x, **int** y, **int** value) {

**int** tiles[] = getUsedTiles(x, y);

**if** (value != 0) {

**for** (**int** tile : tiles) {

**if** (tile == value)

**return** **false**;

}

}

setTile(x, y, value); // nếu số không bị trùng thì chèn số vào mảng

calculateUsedTiles();

**return** **true**;

}

**public** **void** setSelectedTile(**int** tile) {

**if** (game.setTileIfValid(selectX, selectY, tile)) {

invalidate();

game.WinGame();

}

**else**

{

Toast toast = Toast.*makeText*(game, "số này đã có rồi", Toast.***LENGTH\_LONG***);

toast.setGravity(Gravity.***CENTER***, 0, 0);

toast.show();

}

}

