

PROJECT **REPORT**

TEAM MEMBERS:

VIVEK SONAWANE

MAHESH POTULLWAR

SHUBHAM KOTHAWADE

INTRODUCTION **ABOUT 2048 GAME**

2048 game is most addicting puzzle game. It is kind of logical game and easy to play. The game has 4*4 grid. In this game, tile of 2 or 4 is generated. The tile of similar numbers is added and pile up to 2048 tile. Today more than 10 million have become users of this game. About 200 versions of this game have been created yet.

RULES AND OBJECTIVES

1. 2048 game is played on 4*4 grid
2. Player can move the tiles smoothly using 4 arrow keys.
3. After every move, a random tile is generated on the empty spot of the board with either value 2 or 4.

4. In chosen direction tile can slide through empty spaces and is added if similar tile is present beside the sliding tile.

5. The resulting tile cannot merge with another tile in the same move.

6. The player wins the game when the tile with a value of 2048 appears on the board.

WORK DISTRIBUTION

1. Vivek Sonawane (team leader) made some program for upward and downward function. Took the task for generation of random tile of 2 or 4 in a game using random number generator function. Try to solve the errors found while execution of the program. Prepare the SRS document . User manual required for playing the game.
2. Mahesh Pottulwar also wrote some program for upward and downward function. Gave an idea to make right function code by making specific changes in downward function code. I wrote the code to run the game continuously taking continuous inputs. I assist

vivek for writing program for random function. Try to find errors found while execution of the program.

3. Shubham Kothawade helped the team members in writing program for upward and downward function. Made particular changes in upward function code and made code for left function. Helped sometimes in finding errors in the program when it failed to execute. With the helped of simplecpp graphics video Lecture, understand the functions of simplecpp Graphics. Copy the code of puzzle game given in video lecture and made slight changes in the code according to requirement of 2048 game. But failed to write code for addition of similar tiles and generation of random tile of 2 or 4 in the game. Graphics remain incomplete for the game. Wrote final stage report and group diary.

Final Code for Ultimate 2048:-

```
#include <iostream>

#include<cstdlib>

#include<ctime>

using namespace std;

void upwards(int grid[][4]);

void downwards(int grid[][4]);

void left(int grid[][4]);

void right(int grid[][4]);

void assign(int old[][4],int temp[][4]);

bool equal(int a1[][4],int a2[][4]);

int main()

{

    int

grid[4][4],temp[4][4],gridu[4][4],gridd[4][4],gridl[4][4],gridr[4][4],temp1[4][4],temp2[4][4],temp3[4][4],temp4[4][4];

    int i,j,k;

    char input;

    srand(time(NULL));

    cout<<"\t \t ULTIMATE 2048"<<endl;

    cout<<"Controls:"<<endl;

    cout<<"\t W: For moving tiles up"<<endl;

    cout<<"\t S: For moving tiles down"<<endl;

    cout<<"\t A: For moving tiles left"<<endl;

    cout<<"\t D: For moving tiles right"<<endl;

    //all blocks are blank initially

    for(i=0;i<4;i++)

        {
```

```
    for(j=0;j<4;j++)  
        grid[i][j]=0;  
}
```

```
i=rand()%4;  
j=rand()%4;  
k=rand()%4;  
    if(k<2) {grid[i][j]=2;}  
    else {grid[i][j]=4;}
```

```
for(i=0;i<4;i++)  
  
    {  
        for(j=0;j<4;j++){  
            cout<<grid[i][j]<<"\t";  
        }cout<<endl;} cout<<endl;
```

```
for(;;) {
```

```
    assign(grid,temp);  
    cout<<"Where do you want to slide tiles? (Up:W, Down:S, Left:A, Right:D) ";  
    cin>>input;  
    if(input=='w')  
    {
```

```

        upwards(grid);
    }
    else if(input=='s')
    {
        downwards(grid);
    }
    else if(input=='a')
    {
        left(grid);
    }

    else if(input=='d')
    {
        right(grid);
    }

    else{cout<<"Invalid input!!!"; return -1;}
    assign(temp,temp1);
    upwards(temp);
    assign(temp,gridu);
    assign(temp1,temp2);
    downwards(temp1);
    assign(temp1,gridd);
    assign(temp2,temp3);
    left(temp2);
    assign(temp2,gridl);
    assign(temp3,temp4);
    right(temp3);
    assign(temp3,gridr);

```

```

    if((equal(gridu,temp4)) && (equal(gridd,temp4)) && (equal(gridl,temp4)) &&
(equal(gridr,temp4)))

```

```

    {cout<<"Game over!!!"; break;}

```

```

    else { while(true) {i=rand()%4; j=rand()%4;

```

```

        if(grid[i][j]==0) {k=rand()%4; if(k<2) {grid[i][j]=2;break;} else
{grid[i][j]=4;break;}}

```

```

    }

```

```

    for(i=0;i<4;i++){

```

```

        for(j=0;j<4;j++){

```

```

            cout<<grid[i][j]<<"\t";

```

```

        }cout<<endl;}

```

```

    for(i=0;i<4;i++) {

```

```

        for(j=0;j<4;j++) if(grid[i][j]==2048) {cout<<"Congratulations!!! You won this
game";cout<<"You may continue the game.";break;}}

```

```

    }}

```

```

    return 0; }

```

```

//check equality

```

```

bool equal(int a1[4][4],int a2[4][4])

```

```

{ int i,j;

```

```

    for(i=0;i<4;i++)

```

```

    {

```

```

        for(j=0;j<4;j++)

```

```

        {

```

```

            if(a1[i][j]!=a2[i][j]) {return 0;}

```

```

    }
    } return 1;
}
//assign function
void assign(int old[4][4],int temp[4][4])
{
    int i,j;
    for(i=0;i<4;i++)
    {
        for(j=0;j<4;j++)
        {
            temp[i][j]=old[i][j];
        }
    } return ;
}

```

//for moving up

```

void upwards(int grid[4][4])
{
    int j,sum;
    for(j=0;j<4;j++)
    {
        if(grid[0][j]!=0)
        {
            if(grid[1][j]==0){
                if(grid[2][j]==0 && grid[3][j]==0) {break;}
                else {
                    if(grid[2][j]==0 || grid[3][j]==0) {
                        if(grid[2][j]==0)
{grid[1][j]=grid[3][j];

```



```

grid[3][j]=0;}

else {grid[1][j]=grid[2][j];
      grid[2][j]=0;}

if(grid[0][j]==grid[1][j])

else {break;}

}

else {grid[1][j]=grid[2][j]; grid[2][j]=grid[3][j];

grid[3][j]=0;

if(grid[1][j]==grid[2][j]) {grid[1][j]=2*grid[2][j];

grid[2][j]=0;}

else

if(grid[0][j]==grid[1][j]){grid[0][j]=2*grid[1][j];grid[1][j]=grid[2][j];grid[2][j]=0;}

else{continue;}

}

}

}

else if(grid[1][j]==grid[0][j]){sum=2*grid[0][j];

grid[0][j]=sum;

grid[1][j]=0;

if((grid[2][j]==0)&&(grid[3][j]==0)){continue;}

else{ if(grid[2][j]==grid[3][j]) {

sum=2*grid[2][j];

grid[1][j]=sum;

grid[2][j]=0;

grid[3][j]=0;

}

else if (grid[2][j]==0) {

grid[1][j]==grid[3][j];grid[3][j]=0;}

else{grid[1][j]=grid[2][j];

grid[2][j]=grid[3][j];

```

```

        grid[3][j]=0;}

    }

    }

    else {if(grid[1][j]!=grid[2][j] && grid[2][j]!=grid[3][j] && grid[1][j] !=
grid[3][j]) { if(grid[3][j]==0){continue;}

        else if(grid[1][j]!=0 && grid[2][j]!=0 &&
grid[3][j]!=0) {continue;}

        else if(grid[2][j]==0)

        {grid[2][j]=grid[3][j];grid[3][j]=0;}

        else if(grid[1][j]==0)

        {grid[1][j]=grid[2][j];grid[2][j]=grid[3][j];grid[3][j]=0;} }

    else if(grid[1][j]==grid[2][j]) {grid[1][j]=2*grid[2][j];

        grid[2][j]=grid[3][j];

        grid[3][j]=0; }

    else if(grid[2][j]==grid[3][j]) {

        grid[2][j]=2*grid[3][j];

        grid[3][j]=0;

    }

    else if (grid[2][j]==0 && grid[1][j]==grid[3][j]) {

        grid[1][j]=2*grid[3][j];

        grid[3][j]=0;

    }

} }

```

```

else{

    if(grid[1][j]!=grid[2][j] && grid[2][j]!=grid[3][j] && grid[1][j] !=
grid[3][j])

        { if(grid[1][j]!=0 && grid[2][j]!=0 &&
grid[3][j]!=0){grid[0][j]=grid[1][j];

                                                    grid[1][j]=grid[2][j];
                                                    grid[2][j]=grid[3][j];
                                                    grid[3][j]=0;

                                                    }

        else if(grid[1][j]==0 ){grid[0][j]=grid[2][j];
            grid[1][j]=grid[3][j];
            grid[2][j]=0;
            grid[3][j]=0;}

        else if(grid[2][j]==0){grid[0][j]=grid[1][j];
            grid[1][j]=grid[3][j];
            grid[3][j]=0;}

        else if(grid[3][j]==0){grid[0][j]=grid[1][j];
            grid[1][j]=grid[2][j];
            grid[2][j]=0;}

        }

    else if(grid[1][j]==grid[2][j]) {if( grid[1][j]!=0) {

                                                    grid[1][j]=grid[3][j];
                                                    grid[2][j]=0;
                                                    grid[3][j]=0; }

        else {grid[0][j]=grid[3][j];
            grid[3][j]=0;}

    }

}

```

```

else if(grid[2][j]==grid[3][j]) { if(grid[1][j]==0) {
grid[0][j]=2*grid[1][j];

grid[0][j]=2*grid[2][j];
grid[1][j]=0;
grid[2][j]=0;
grid[3][j]=0;
}
else {
grid[0][j]=grid[1][j];
grid[1][j]=2*grid[2][j];
grid[2][j]=0;
grid[3][j]=0;
}
}

else if (grid[2][j]==0 && grid[1][j]==grid[3][j]) {
grid[0][j]=2*grid[1][j];
grid[1][j]=0;
grid[3][j]=0;
}

else if(grid[2][j]!=0 && grid[1][j]==0 && grid[3][j]==0)
{grid[0][j]=grid[2][j];

grid[2][j]=0;}

else if(grid[2][j]!=0 && grid[1][j]==grid[3][j])
{grid[0][j]=grid[1][j];

grid[1][j]=grid[2][j];
grid[2][j]=grid[3][j];
grid[3][j]=0;}

}

```

```

        } return ; }

// for moving down
void downwards(int grid[][4])
{int j,sum;
  for(j=0;j<4;j++)
  {

      if(grid[3][j]!=0)
      {
          if(grid[2][j]==0){
              if(grid[1][j]==0 && grid[0][j]==0) {break;}
              else {
                  if(grid[1][j]==0 || grid[0][j]==0) {
                      if(grid[1][j]==0)
{grid[2][j]=grid[0][j];

                      grid[0][j]=0;}
                      else {grid[2][j]=grid[1][j];
                          grid[1][j]=0;}
                      if(grid[3][j]==grid[2][j])

{grid[3][j]=2*grid[3][j]; grid[2][j]=0;}

                      else {break;}
                  }

                  else {grid[2][j]=grid[1][j]; grid[1][j]=grid[0][j];

                      if(grid[2][j]==grid[1][j]) {grid[2][j]=2*grid[1][j];

                      grid[1][j]=0;}

                      else

if(grid[3][j]==grid[2][j]){grid[3][j]=2*grid[2][j];grid[2][j]=grid[1][j];grid[1][j]=0;}

                      else{continue;}
                  }
              }
          }
      }
  }
}

```

```

    }
else if(grid[2][j]==grid[3][j]){sum=2*grid[3][j];
    grid[3][j]=sum;
    grid[2][j]=0;
    if((grid[1][j]==0)&&(grid[0][j]==0)){continue;}
    else{ if(grid[1][j]==grid[0][j]) {
        sum=2*grid[1][j];
        grid[2][j]=sum;
        grid[1][j]=0;
        grid[0][j]=0;
        }
    else if (grid[1][j]==0) { grid[2][j]==grid[0][j];
grid[0][j]=0;}

    else{grid[2][j]=grid[1][j];
        grid[1][j]=grid[0][j];
        grid[0][j]=0;}
    }
    }

    else {if(grid[2][j]!=grid[1][j] && grid[1][j]!=grid[0][j] && grid[2][j] !=
grid[0][j]) { if(grid[0][j]==0){continue;}

    else if(grid[0][j]!=0 && grid[2][j]!=0 &&
grid[1][j]!=0) {continue;}

    else if(grid[1][j]==0)

    {grid[1][j]=grid[0][j];grid[0][j]=0;}

    else if(grid[2][j]==0)
    {grid[2][j]=grid[1][j];grid[1][j]=grid[0][j];grid[0][j]=0;} }

    else if(grid[2][j]==grid[1][j]) {grid[2][j]=2*grid[1][j];
    grid[1][j]=grid[0][j];
    grid[0][j]=0; }

```

```
else if(grid[1][j]==grid[0][j]) {
```

```
    grid[1][j]=2*grid[0][j];
```

```
    grid[0][j]=0;
```

```
}
```

```
else if (grid[1][j]==0 && grid[2][j]==grid[0][j]) {
```

```
    grid[2][j]=2*grid[0][j];
```

```
    grid[0][j]=0;
```

```
}
```

```
}}
```

```
else{
```

```
    if(grid[2][j]!=grid[1][j] && grid[1][j]!=grid[0][j] && grid[2][j] !=  
grid[0][j])
```

```
        { if(grid[2][j]!=0 && grid[3][j]!=0 &&  
grid[0][j]!=0){grid[3][j]=grid[2][j];
```

```
            grid[2][j]=grid[1][j];
```

```
            grid[1][j]=grid[0][j];
```

```
            grid[0][j]=0;
```

```
        }
```

```
    else if(grid[2][j]==0 ){grid[3][j]=grid[1][j];
```

```
        grid[2][j]=grid[0][j];
```

```
        grid[1][j]=0;
```

```
        grid[0][j]=0;}
```

```
    else if(grid[1][j]==0){grid[3][j]=grid[2][j];
```

```

        grid[2][j]=grid[0][j];
        grid[0][j]=0;}
    else if(grid[0][j]==0){grid[3][j]=grid[2][j];
        grid[2][j]=grid[1][j];
        grid[1][j]=0;}
    }
else if(grid[2][j]==grid[1][j]) {if( grid[2][j]!=0) {
grid[3][j]=2*grid[2][j];

        grid[2][j]=grid[0][j];
        grid[1][j]=0;
        grid[0][j]=0; }
    else {grid[3][j]=grid[0][j];
        grid[0][j]=0;}

    }
else if(grid[1][j]==grid[0][j]) { if(grid[2][j]==0) {
grid[0][j]=2*grid[1][j];

        grid[3][j]=2*grid[1][j];
        grid[2][j]=0;
        grid[1][j]=0;
        grid[0][j]=0;
        }
    else {
        grid[3][j]=grid[2][j];
        grid[2][j]=2*grid[1][j];
        grid[1][j]=0;
        grid[0][j]=0;

    }
}

```



```

        }
        else if (grid[1][j]==0 && grid[2][j]==grid[0][j]) {
            grid[3][j]=2*grid[2][j];
            grid[2][j]=0;
            grid[0][j]=0;
        }
        else if(grid[1][j]!=0 && grid[2][j]==0 && grid[0][j]==0)
{grid[3][j]=grid[1][j];
            grid[1][j]=0;}
        else if(grid[1][j]!=0 && grid[2][j]==grid[0][j])
{grid[3][j]=grid[2][j];
            grid[2][j]=grid[1][j];
            grid[1][j]=grid[0][j];
            grid[0][j]=0;}
    }

```

```

    } return ; }

```

//for moving left

```

void left(int grid[4][4])

```

```

{int i,sum;

```

```

    for(i=0;i<4;i++)

```

```

    {

```

```

        if(grid[i][0]!=0)

```

```

        {

```

```

            if(grid[i][1]==0){

```

```

                if(grid[i][2]==0 && grid[i][3]==0) {break;}
            }
        }
    }
}

```

```

else {
    if(grid[i][2]==0 || grid[i][3]==0) {
        if(grid[i][2]==0)
            grid[i][3]=0;
        else {grid[i][1]=grid[i][2];
            grid[i][2]=0;}
        if(grid[i][0]==grid[i][1])
            {grid[i][0]=2*grid[i][0]; grid[i][1]=0;}
        else {break;}
    }
    else {grid[i][1]=grid[i][2]; grid[i][2]=grid[i][3];
        if(grid[i][1]==grid[i][2]) {grid[i][1]=2*grid[i][2];
            grid[i][2]=0;}
        else
            if(grid[i][0]==grid[i][1]){grid[i][0]=2*grid[i][1];grid[i][1]=grid[i][2];grid[i][2]=0;}
            else{continue;}
    }
}

else if(grid[i][1]==grid[i][0]){sum=2*grid[i][0];
    grid[i][0]=sum;
    grid[i][1]=0;
    if((grid[i][2]==0)&&(grid[i][3]==0)){continue;}
    else{ if(grid[i][2]==grid[i][3]) {
        sum=2*grid[i][2];
        grid[i][1]=sum;
        grid[i][2]=0;
        grid[i][3]=0;
    }
}

```

```

else if (grid[i][2]==0) { grid[i][1]==grid[i][3];
grid[i][3]=0;}

else{grid[i][1]=grid[i][2];
grid[i][2]=grid[i][3];
grid[i][3]=0;}
}
}

else {if(grid[i][1]!=grid[i][2] && grid[i][2]!=grid[i][3] && grid[i][1] !=
grid[i][3]) { if(grid[i][3]==0){continue;}

else if(grid[i][1]!=0 && grid[i][2]!=0 &&
grid[i][3]!=0) {continue;}

else if(grid[i][2]==0)

{grid[i][2]=grid[i][3];grid[i][3]=0;}

else if(grid[i][1]==0)

{grid[i][1]=grid[i][2];grid[i][2]=grid[i][3];grid[i][3]=0;} }

else if(grid[i][1]==grid[i][2]) {grid[i][1]=2*grid[i][2];
grid[i][2]=grid[i][3];
grid[i][3]=0; }

else if(grid[i][2]==grid[i][3]) {

grid[i][2]=2*grid[i][3];
grid[i][3]=0;

}

else if (grid[i][2]==0 && grid[i][1]==grid[i][3]) {
grid[i][1]=2*grid[i][3];
grid[i][3]=0;
}

```

```
}}
```

```
else{
```

```
if(grid[i][1]!=grid[i][2] && grid[i][2]!=grid[i][3] && grid[i][1] !=  
grid[i][3])
```

```
{ if(grid[i][1]!=0 && grid[i][2]!=0 &&  
grid[i][3]!=0){grid[i][0]=grid[i][1];
```

```
grid[i][1]=grid[i][2];
```

```
grid[i][2]=grid[i][3];
```

```
grid[i][3]=0;
```

```
}
```

```
else if(grid[i][1]==0 ){grid[i][0]=grid[i][2];
```

```
grid[i][1]=grid[i][3];
```

```
grid[i][2]=0;
```

```
grid[i][3]=0;}
```

```
else if(grid[i][2]==0){grid[i][0]=grid[i][1];
```

```
grid[i][1]=grid[i][3];
```

```
grid[i][3]=0;}
```

```
else if(grid[i][3]==0){grid[i][0]=grid[i][1];
```

```
grid[i][1]=grid[i][2];
```

```
grid[i][2]=0;}
```

```
}
```

```
else if(grid[i][1]==grid[i][2]) {if( grid[i][1]!=0) {  
grid[i][0]=2*grid[i][1];
```

```
grid[i][1]=grid[i][3];
```

```
grid[i][2]=0;
```

```
grid[i][3]=0; }
```

```
else {grid[i][0]=grid[i][3];
```

```

        grid[i][3]=0;}

    }

    else if(grid[i][2]==grid[i][3]) { if(grid[i][1]==0) {

grid[i][0]=2*grid[i][1];

        grid[i][0]=2*grid[i][2];
        grid[i][1]=0;
        grid[i][2]=0;
        grid[i][3]=0;
    }

    else {

        grid[i][0]=grid[i][1];
        grid[i][1]=2*grid[i][2];
        grid[i][2]=0;
        grid[i][3]=0;

    }

    }

    else if (grid[i][2]==0 && grid[i][1]==grid[i][3]) {

        grid[i][0]=2*grid[i][1];
        grid[i][1]=0;
        grid[i][3]=0;
    }

    else if(grid[i][2]!=0 && grid[i][1]==0 && grid[i][3]==0)

{grid[i][0]=grid[i][2];

        grid[i][2]=0;}

    else if(grid[i][2]!=0 && grid[i][1]==grid[i][3])

{grid[i][0]=grid[i][1];

        grid[i][1]=grid[i][2];
        grid[i][2]=grid[i][3];

```

```

grid[i][3]=0;}

}

} return ; }

//for moving right
void right(int grid[4][4])
{int i,sum;
for(i=0;i<4;i++)
{

if(grid[i][3]!=0)
{
if(grid[i][2]==0){
if(grid[i][1]==0 && grid[i][0]==0) {break;}
else {
if(grid[i][1]==0 || grid[i][0]==0) {
if(grid[i][1]==0)
grid[i][0]=0;}
else {grid[i][2]=grid[i][1];
grid[i][1]=0;}
if(grid[i][3]==grid[i][2])
{grid[i][3]=2*grid[i][3]; grid[i][2]=0;}
else {break;}
}
else {grid[i][2]=grid[i][1]; grid[i][1]=grid[i][0];
grid[i][3]=3;
if(grid[i][2]==grid[i][1]) {grid[i][2]=2*grid[i][1];
grid[i][1]=0;}

```

```

else
if(grid[i][3]==grid[i][2]){grid[i][3]=2*grid[i][2];grid[i][2]=grid[i][1];grid[i][1]=0;}
else{continue;}
}
}
}
else if(grid[i][2]==grid[i][3]){sum=2*grid[i][3];
grid[i][3]=sum;
grid[i][2]=0;
if((grid[i][1]==0)&&(grid[i][0]==0)){continue;}
else{ if(grid[i][1]==grid[i][0]) {
sum=2*grid[i][1];
grid[i][2]=sum;
grid[i][1]=0;
grid[i][0]=0;
}
else if (grid[i][1]==0) {
grid[i][2]==grid[i][0];grid[i][0]=0;}

else{grid[i][2]=grid[i][1];
grid[i][1]=grid[i][0];
grid[i][0]=0;}
}
}
else {if(grid[i][2]!=grid[i][1] && grid[i][1]!=grid[i][0] && grid[i][2] !=
grid[i][0]) { if(grid[i][0]==0){continue;}

else if(grid[i][2]!=0 && grid[i][1]!=0 &&
grid[i][0]!=0) {continue;}

else if(grid[i][1]==0)

{grid[i][1]=grid[i][0];grid[i][0]=0;}

else if(grid[i][2]==0)
{grid[i][2]=grid[i][1];grid[i][1]=grid[i][0];grid[i][0]=0;} }

```

```
else if(grid[i][2]==grid[i][1]) {grid[i][2]=2*grid[i][1];
```

```
    grid[i][1]=grid[i][0];
```

```
    grid[i][0]=0; }
```

```
else if(grid[i][1]==grid[i][0]) {
```

```
    grid[i][1]=2*grid[i][0];
```

```
    grid[i][0]=0;
```

```
}
```

```
else if (grid[i][1]==0 && grid[i][2]==grid[i][0]) {
```

```
    grid[i][2]=2*grid[i][0];
```

```
    grid[i][0]=0;
```

```
}
```

```
}}
```

```
else{
```

```
    if(grid[i][2]!=grid[i][1] && grid[i][1]!=grid[i][0] && grid[i][2] !=  
    grid[i][0])
```

```
        { if(grid[i][2]!=0 && grid[i][1]!=0 &&  
        grid[i][0]!=0){grid[i][3]=grid[i][2];
```

```
            grid[i][2]=grid[i][1];
```

```
            grid[i][1]=grid[i][0];
```

```
            grid[i][0]=0;
```

```
}
```

```
else if(grid[i][2]==0 ){grid[i][3]=grid[i][1];
```



```

        grid[i][2]=grid[i][0];
        grid[i][1]=0;
        grid[i][0]=0;}
    else if(grid[i][1]==0){grid[i][3]=grid[i][2];
        grid[i][2]=grid[i][0];
        grid[i][0]=0;}
    else if(grid[i][0]==0){grid[i][3]=grid[i][2];
        grid[i][2]=grid[i][1];
        grid[i][1]=0;}
    }
else if(grid[i][2]==grid[i][1]) {if( grid[i][2]!=0) {
grid[i][3]=2*grid[i][2];

        grid[i][2]=grid[i][0];
        grid[i][1]=0;
        grid[i][0]=0; }
    else {grid[i][3]=grid[i][0];
        grid[i][0]=0;}

    }
else if(grid[i][1]==grid[i][0]) { if(grid[i][2]==0) {
grid[i][3]=2*grid[i][2];

        grid[i][3]=2*grid[i][1];
        grid[i][2]=0;
        grid[i][1]=0;
        grid[i][0]=0;
        }
    else {
        grid[i][3]=grid[i][2];
        grid[i][2]=2*grid[i][1];

```

```

        grid[i][1]=0;
        grid[i][0]=0;

    }

}

else if (grid[i][1]==0 && grid[i][2]==grid[i][0]) {
    grid[i][3]=2*grid[i][2];
    grid[i][2]=0;
    grid[i][0]=0;
}

else if(grid[i][1]!=0 && grid[i][2]==0 && grid[i][0]==0)
{grid[i][3]=grid[i][1];

    grid[i][1]=0;}

else if(grid[i][1]!=0 && grid[i][2]==grid[i][0])
{grid[i][3]=grid[i][2];

    grid[i][2]=grid[i][1];
    grid[i][1]=grid[i][0];
    grid[i][0]=0;}

}

} return ; }

```

