Name _____ Role (circle one) programmer/computer/project manager

Name _____ Role (circle one) programmer/computer/project manager

Name _____ Role (circle one) programmer/computer/project manager

Name _____ Role (circle one) quality control

Mine Hunter

Your Tasks (Mark these off as you go)

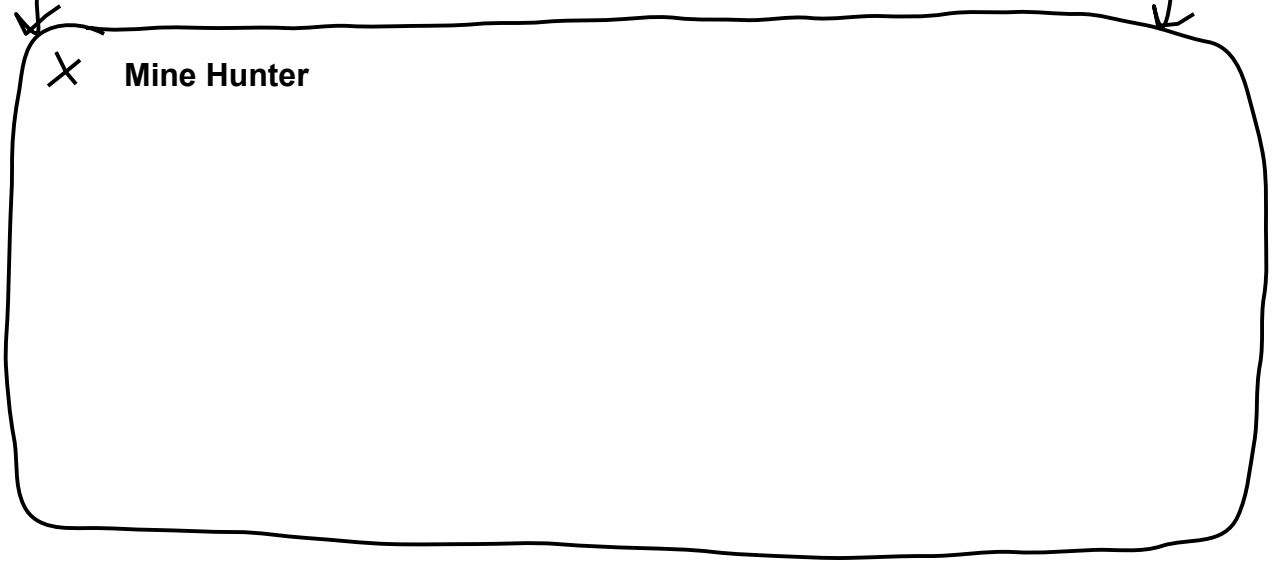
- ☐ Interpret the relationship between the MineHunter classes and the MineHunterInterface
- ☐ Initialize the mines 2D array
- ☐ Write the MakeMines method
- ☐ Have Ms. Pluska check off your initialized mines 2D array and your MakeMines method
- ☐ Complete challenges 1 thru 4
- ☐ Have Ms. Pluska check off your challenges 1 thru 4 before you continue
- ☐ Receive credit for the group portion of this lab
- ☐ Receive credit for the individual portion of this lab

☐ Interpret the relationship between the MineHunter classes and the MineHunterInterface

In this lab you will build upon the grid you created in an earlier lab to create a version of the popular game Mine Sweeper. Before we get started we will review the architecture of this program.

Locate the MineHunter.java class. The job of this class is to create the "Frame" in which are game will be displayed. In this frame we create a new instance of a MineHunterPane()

```
public static void main(String[] args) {
            JFrame frame = new JFrame("Mine Hunter");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.getContentPane().add(new MineHunterPane());
            frame.pack();
            frame.setVisible(true);
}
```



Now locate the MineHunterPane.java

In this class we create new a new Pane. We also create instances of the MineHunterGrid and the MineHunterControls and add them to this pane.

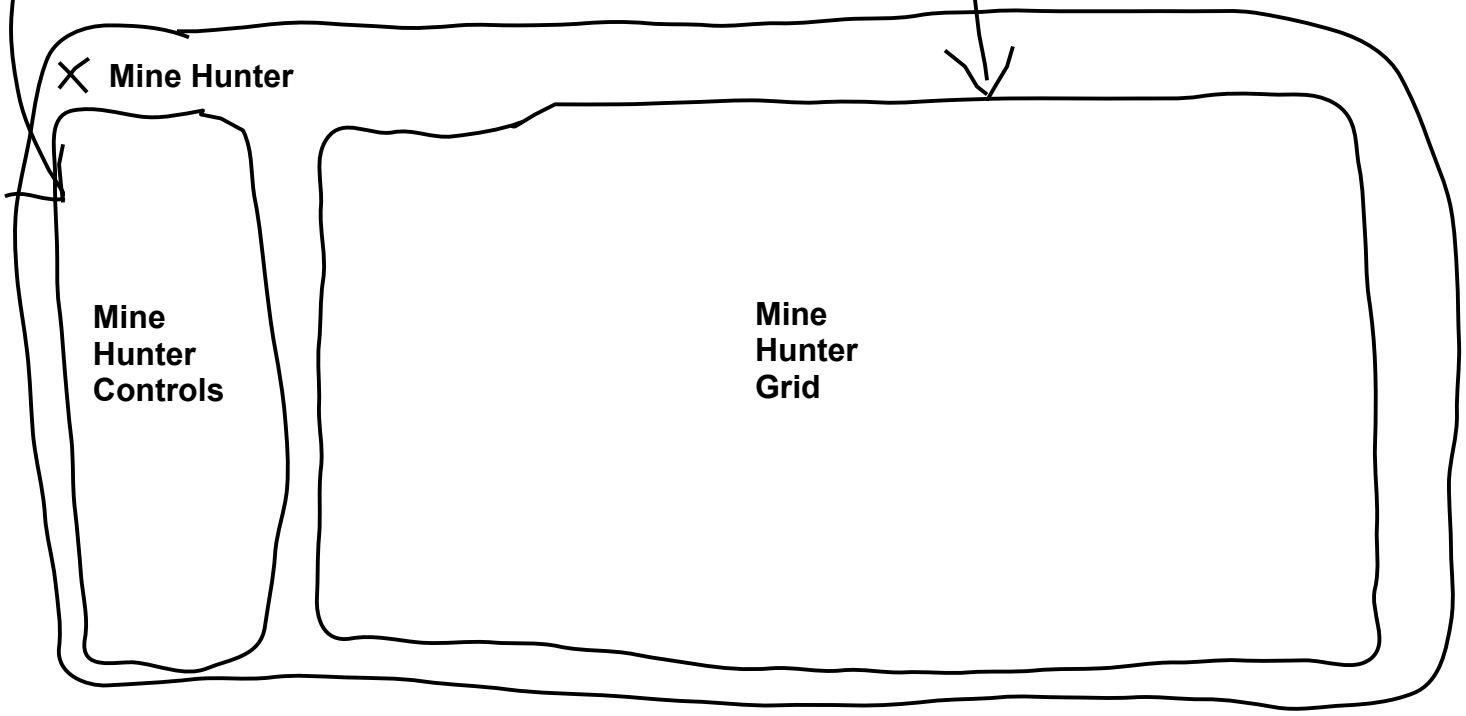
```
public class MineHunterPane extends JPanel {
    private int a;
    private MineHunterGrid lbg = new MineHunterGrid(20, 20);
    private MineHunterControls lbc = new MineHunterControls(new
colorListener());

    public MineHunterPane(){

            setLayout(new BorderLayout());
            setPreferredSize(new Dimension(800, 800));
            add(lbg,BorderLayout.CENTER);
            add(lbc, BorderLayout.WEST);

    }

    private class colorListener implements ActionListener{
            @Override
            public void actionPerformed(ActionEvent e)
            {
                  a = lbc.getButtonAction((JButton) e.getSource());
                                    lbg.doButtonAction(a);
            }
    }
}
```



Now, locate the MineHunterControls.java class.

This class is written for you. The purpose of this class is display the color key, which can be used to determine the number of mines around each button clicked. There is also a reset button, which can be used to start a new game.

Now, locate the interface MineHunterInterface.java.

This interface provides the framework for the game. Whatever class implements this interface must also implement the methods it defines. This will be your job :-)

Finally, locate the MineHunterGrid.java class.

You will notice at the top of this class the following declaration and the key work “implements. The MineHunterGrid class implements an ActionListener which will listen for when a button is clicked. It also implements the MineHunterInterface.

```
public class MineHunterGrid extends JPanel implements ActionListener,
MineHunterInterface
```

□ Initialize the mines 2D array

Much of the MineHunterGrid class has been written already. For our game to function however, you will need to complete the methods in this class.

Before we do so, locate the constructor in this class. You will notice that the purpose of this class is to create a grid of buttons. In the constructor we also need to create two boolean arrays. The first boolean array, locatedMines, will store the mines the user has located (or thinks they located). The second array called mines will be used to store the mines on the grid as true/false values.

The mines array has already been declared, you can initialized it using the following line of code,

```
mines = new boolean[gridDimensions][gridDimensions];
```

Locate the constructor in the MineHunterGrid class.

Add this line of code under the section that reads

```
//TODO: initialize the mines 2D array to a new boolean 2D array
//the number of rows in this array is equal to the gridDimensions
//the number of cols in this array is equal to the gridDimenstions
```

☐ Write the MakeMines method

Now that you have initialized your mines array you can write the MakeMines method. You will do this on the paper you have been provided.

Begin by declaring your MakeMines method

```
public void makeMines(int percent){

}
```

Notice this method accepts an int parameter, percent. This parameter represents the percentage of mines we want on our grid. If a mine is present on a location on our grid, this will be represented as a true value, otherwise the location will be represented as a false value.

To complete this method, we will iterate over each location in the 2-dimensional array. At each location we will generate a random number which represents a percent. And, if the number is greater than the 100-percent, we will set the location in the mines array to "true".

The completed makeMines method is shown below. Write this method on the paper provided.

Completed makeMines method

```
public void makeMines(int percent){
        for(int rows = 0; rows < tiles.length; rows++){
            for(int cols = 0; cols < tiles[rows].length;cols++){
                  int percentMines = (int)(Math.random()*100);
                        if(percentMines>(100-percent)){
                              mines[rows][cols] = true;
                        }
            }

        }
}
```

☐ Have Ms. Pluska check off your initialized mines 2D array and your makeMines method

Before you continue have Ms. Pluska check off your initialized 2D array and your makeMines method

Do not continue until you have Ms. Pluska's (or her designated TA's) signature _____

☐ Complete Challenges 1 thru 3

Challenge 1

Write a method called showMines()

showMines() is a void type method. The purpose of this method is to display the mines on the grid. To accomplish this task, you will iterate over the mines array you created. At each "true" location, you will set the background color of the button at that location to red (.setBackground(Color.RED). You will also set the icon on the button to a flag (.setIcon(flag);

Challenge 2

Write a method called checkForMine.

checkForMine is a void type method, which accepts two parameters (int mineX, int mineY) which represent where the user has clicked. The purose of this method is to check whether a mine is set at the mineX, mineY location. If a mine is found at the location, call showMines() to indicate that the player has lost.

Challenge 3

Write the paintTiles method.

paintTiles is a void type method, which accepts two parameters(clickedX, clickedY) which represent where the user has clicked. The purose of this method is to count all mines touching this location. And depending on the number of mines will change the background color of the button clicked to a different color in the array,

```
private Color[] buttonColors = {

Color.WHITE, Color.GRAY, Color.GREEN, Color.YELLOW, Color.PINK, Color.BLUE,
Color.RED, Color.BLACK

};
```

For example, if the user clicked on the button shown below, the background color should change to yellow, because it is touching three mines.

mine		mine
	User clicked	
		mine

Your method should account for boundary conditions. For example, if the user clicked on the button below, your program should not go out of bounds. The color of the clicked button should change to green.

	mine
User clicked	
	mine

Challenge 4

Write the isDone() method

isDone() is a boolean type method. Each time isDone() is called, it checks to see if the user has identified all the mines. If all the mines have been the identified, the method will return true, otherwise it will return false.

Have Ms. Pluska check off challenges 1 thru 4

Before you continue have Ms. Pluska check off challenges 1 thru 4.

Do not continue until you have Ms. Pluska's (or her designated TA's) signature _____

□ Receive Credit for the group portion of this lab

Make sure to indicate the names of all group members, then submit this lab to the needs to be graded folder to receive credit for the group portion of this lab.

□ Receive Credit for the individual portion of this lab

Implement challenges 1 thru 4 on your computer. You should have fully functioning game, once these methods are implemeted. Show Ms. Pluska your completed game to receive credit for this lab.