Doodlepad Slide Show Create Task

Victoria Ruan

2b)

Before I started, I divided the whole slide into several different parts and created one java file for each part. Each java file is responsible to create one slide and can be run individually. Every slide follows a pattern:

1. Clear the slide.
2. Create the title.
3. Import images.
4. Get a 2D array from the corresponding data files (for the 3 visualization pages).
5. Get the average of data of interest (for the 3 visualization pages).
6. Visualize the data through various means (for the 3 visualization pages).
7. Add explanation.

When I incorporated these files together, I met a problem about throwing exceptions. Since the reading file method would throw fileNotFound errors during compiling, the upper level code block that called the method will also throw fileNotFound errors. I tried to simply add throws Exception after the constructor's and every methods' name, but the code could not be compiled due to the incompatibility. After checking on StackOverflow and other websites, I chose to use try{} catch{} to avoid the incompatibility. All the work was done individually with the help about the constructor from Mr.Wong.

2c)

```
public double[][] getData(String fileNameRead, int start, int end){
    int row = 0;
    int number = end - start;
    String myLine = "";
    File readFile = new File(fileNameRead);
    try{
        Scanner sc = new Scanner(readFile);
        myLine = sc.nextLine();
        while(sc.hasNextLine()){
            myLine = sc.nextLine();
```

```
                row += 1;
            }
            sc.close();
        }catch(Exception ex){
        }

        double[][] dataArray = new double[number][row];
        try{
            Scanner sc2= new Scanner(readFile);
            myLine = sc2.nextLine();
            int j=0;
            while(sc2.hasNextLine()){
                myLine = sc2.nextLine();
                String[] lineArray = myLine.split(",");
                for(int i=0; i<number; i++){
                    dataArray[i][j] = Double.parseDouble(lineArray[start+i]);
                }
                j++;
            }
        }catch(Exception ex){
        }
        return dataArray;
}

public double[] getAverage(double[][] dataArray){
        double [] average = new double[dataArray.length];
        for(int i=0; i<dataArray.length; i++){
            double total = 0;
            for(int j=0; j<dataArray[0].length; j++){
```

```
                total += dataArray[i][j];
            }
            average[i] = total/dataArray[0].length;
        }
        return average;
}

public void fifthSlide() {
        p.clear();
        double [][] ageArray = getData("age.txt",4,8);
        double [] ageAverage = getAverage(ageArray);
}
```

The above codes show a combination of algorithms that get the data from an input file and get the average of each parameter of interest. This action is divided into three methods:

1) getData

2) getAverage

3) fifthSlide

The getData method basically reads a txt file (converted from csv) and returns a 2D array that represents the data table. The two integer parameters indicate which columns of the data table should be extracted. For the return array, the first index represents column, and the second index represents row. Since arrays' sizes cannot be changed after defined, the program first reads through the file to see how many lines are there in the file. After creating the 2D array based on the row and column size, the program reads the file again to fulfill the data through nested loops. The scanner is created to get the content on each line and the line String will be split into individual String data. According to the start and end bounds, the String data corresponding to the selected part will be parsed into double data type in the 2D array for further use.

After geting the data array, the fifthSlide method then call the getAverage method to get the average. The getAverage method just adds up all the data and divides the number of rows for each column to calculate the average result. The returned array contains the averages for each column.

2d)

```
public void drawLine(double x1, double y1, double x2, double y2,int color){
        Line x = new Line(x1, y1, x2, y2);
        x.setStrokeWidth(4);
        if(color==0){
                x.setStrokeColor(228,64,64);
        }else if(color==1){
                x.setStrokeColor(48,147,154);
        }else if(color==2){
                x.setStrokeColor(239,158,53);
        }else if(color==3){
                x.setStrokeColor(173,100,190);
        }else if(color==4){
                x.setStrokeColor(70,111,174);
        }else if(color==5){
                x.setStrokeColor(57,151,67);
        }else{
                x.setStrokeColor(255,255,255);
        }
        Oval circle = new Oval(x1-3,y1-3,6,6);
        circle.setStroked(false);
        circle.setFillColor(0,0,0);
}
```

The method drawLine abstracts the action of drawing a line for the data visualization. The method has four parameters for starting point x, y position and end point x, y position respectively. The last parameter indicates the color of the line to be set, and two circles will be created to represent the starting and end points. The drawLine method removes all the specific values of creating line visualization and only keeps the abstract variables that can be replaced by all possible values.