Programming Project Group 17 Report

Group member:
SHANE GILLIGAN: 20331916
KAJUS KRISCIUNAS: 20332808
ZHONGYUAN LIU: 19336865
MINJUAN LUO: 20313326

Outline of design:

Main Screen:
Bar Chart:
(1)different months of total cases, (2)different bars for daily case, (3)Text widget: months
Line Chart:
(1)The growth of the cases, (2)The growth in different ranges of months
Pie Chart:
(1)Proportion of confirmed cases in different states, (2)Texts that show the exact percentage of different states
USA map:
(1)50 states(50 images) included in the screen, (2)50 different images that can show the total cases and most recent cases amount when mouse click one state
(3)50 pie chart that show the proportion of a state against all cases for a year
Search:
(1)Text widgets that when the user inputed an exact state, area, start date and end date, the program will show you the exact cases in this state from the start date to the end date.

How we split up the work and organised the team:

Main idea:Different class Different people(Mostly the group was splitted into two pairs: Kajus and Shane, Minjuan and Zhongyuan)

(1)Main program: The whole group came together to figure out how to input the dataset. Each one has their own parts of the program implemented in the main program including bar chart, line chart, pie chart and finally, the USA map, and Zhongyuan designed the homepage.

(2)Bars: Shane coded the Bars class to create the bars

(3)Constants: Everyone added to the constants EVENT_BUTTONs. Kajus coded the map class. Minjuan and Zhongyuan spent time testing all the sizes of the images.

(4)Data: The whole group worked on this class including setting a constructor and several getters to reach the data individually.

(5)Functions: Kajus coded the functions class and added all the images into the data set, Zhongyuan and Minjuan put all the states together to draw the US map.

(6)Graph: Shane coded the graph class to display data on a graph

(7)Lines: Minjuan and Zhongyuan worked together on the Lines class

(8)Map:Kajus coded this class in order to put images on the screen

(9)Pie_Chart:Minjuan Luo coded this part to generate a pie chart

(10)Result:Minjuan Luo coded this part to get a figure from the Data class and put those data into a pie chart.

(11)Screen: Shane and Kajus worked in pairs switching the coding every 30 minutes.

(12)TextWidgets: Kajus coded this class, enabling the user input characters in a text box.

(13)Widget: Shane and Kajus coded the Widget class ~ 30 min each.

Organizing the team: Taking a regular meeting every Monday at 2pm. We will discuss our ideas and problems encountered during the meeting and then set goals for each pair to finish in that week until Thursday. The group members will then work in pairs to code and meet up at 3 pm on Thursday just before the lab session in order to work together and fix bugs.

Features implemented:

1. A main home page to tell the user what we are going to present.
2. A bar chart which shows the top 10 states of infected cases in different ranges of months. A text widget which allows the user to type in numbers from 1 to 12 and generate a graph that whatever he wants to know. Three daily buttons to show most daily cases a state got in three different periods.
3. A line chart which shows the user the growth curve of confirmed cases across the United States. Three different buttons which enable the user to choose different lengths of period and get a bigger or smaller graph.
4. A pie chart which shows the user proportions of each state in confirmed cases. And 50 texts listed the exact percentage of different states.
5. A map that presents the USA territory with 50 states specialised. Users can click each state to get a pop out window which shows the state name, total infected cases and most recent cases. What's more, there is a pie chart generated below to show the proportion of cases for this state against all cases in a year
6. A search function which allows users to enter an exact state name, area name, start date and end date to get infected cases in this time range and this area(this state).

Problems encountered:

Minjuan Luo:

1.Can't generate a line graph in the screen:Problem fixed by adding a set axis in the main program.

2. Can't cut the whole line graph into two pieces or four pieces in the same class:
Can't provide users with a different time period of line chart:Problem solved(Kajus helped) by creating a constructor and giving each graph a different value of variables to make dots keep another style of distance in their IDs.

3.Can't get data from Data class:Kajus helped me fix that bug.

4. Don't know how to draw a pie chart:Problem solved by viewing others pie chart code from the internet and getting inspired.(using an arc instruction)

5.Have no idea how to improve the functionality and make my part work more efficiently: Problem solved by spending an enormous amount of time to debug and find a better solution. The program now can load faster than before.

6. Don't know how to generate a gradient in order to make the pie chart look better to specify different states: Problem solved by adding a for loop which contains three integers and they implement 3, 6, 9 separately in fill() instruction to make a gradient for pie chart.

7.Can't make every state's images fit for each other in the USA map we present:Problem solved by using a picture editor to rotate and change values of different variables of the images in order to change the size of them.

Shane Gilligan:

Shane worked mainly on displaying information in accordance with the program. Notable examples of this were his work with the various bar charts and his work displaying text boxes on the map of America as well as a pie chart to go with it. Various problems that he encountered with these were

1) The bar chart problems,

Due to the vast difference in cases over time (i.e., cases amount after one month/day is significantly smaller than cases amount after 12 months) a problem encountered was accurately fitting the bars to stay on the screen while still being legible. The way Shane went about solving this was adding a divider as a parameter that would divide the top ten total amounts of cases into a number, still proportionate with each other, that can be graphically displayed onto the set screen size. Different periods of time had different dividers so different case amounts could still be displayed on the same plane.

Another problem with this that was encountered was creating accurate points along the y axis to accurately display the real number of cases. For any given input of months it would have been made redundant to hard code any of the values along the y axis for each value so the way Shane worked around this was he took the top cases over that given period of time and matched that with the top of the bar chart, from there he worked his way down multiplying the top value by a fraction over ten so that the y axis would always have ten correlating and observably accurate markers to accurately depict the bar charts.

The graph class was used to create 10 individual bars on a plane and continuously draw them on a set axis.

2) The USA Map Problems,

The main problem with the map and the text box was ensuring the text box was always on screen regardless of where and which state was clicked. Shane went around this by simply adding if statements if the x and y positions were past certain locations then you shift both the text box and text.

The information that was gained when clicked was displayed by using a Boolean which was made true when the colour that the mouse was over matched the colour of the state. When the state was clicked we had already implemented various getter methods to access the state name, total cases and most recent cases and I was able to display them after the query by declaring these variables globally and changing them accordingly. Displaying this information proved to be challenging.

A pie chart was created by Shane as well separate from the piechartscreen class in order to convey the proportion of the total cases in the state clicked on with the total cases in the USA. The problem was working out the angles to make this pie chart work however only one angle was needed to construct this so Shane in his part wrote a simple equation to get the required angle.

Throughout the rest of the program Shane helped code other bits and pieces such as the widgets class and helped create methods to call the different state cases over both set periods of time and custom as there was a problem with gaining this information and then implementing it into the different classes.

Zhongyuan Liu:

1. When drawing the button, the button cannot switch pages, because I put the case in the wrong area in the switch. Finally, Kajus helped me with that and fixed it.

2. When creating a line class, it is very hard to combine three line charts representing different months into one line class. Minjuan helped me to put the three different line classes together.

3. When drawing a line chart, I had no idea how to change the coordinate axis and proportion of line charts of different months. I found a function that can change the coordinate axis, but there were still a few problems with that, Minjuan helped me with that.

4. When drawing a map of the United States, it is difficult to fit all the states together nicely, in the final, through repeated modifications, they were perfectly stitched together.

5. When making the homepage, the loaded picture and text cannot be displayed.Kajus helped me to display the picture and text in the homepage.

6. After the images are displayed, all the images will turn red when the USAmap button is pressed, Kajus fix this problem.

Kajus Krisciunas:

Kajus mainly worked on the backend of the project. He mainly focused on dataset management and queries. Kajus also designed the search screen that allowed the user to search for information about a state accurately.

1. Kajus worked on the design for reading in the dataset and organising it into separate parts i.e Date, Area, etc. Allowing his group to get any information needed about any state. One of the key problems of this was trying to split the date into 3 parts (day, month and year). This was eventually solved by using parseInt and substring.
2. Kajus also implemented the basics of the mousePressed function which would allow his group to add in their own widgets/text widgets and make them function. One of the problems he found was that when implementing the text widgets, their events were not being found. Shane and Kajus came up with a solution for this. They added in another getEvent function in the widget class but with slightly different parameters. This ended up working fine.
3. Kajus added in the keyPressed function which would allow the user to type into the text box on the screen. However a problem occurred when the user would press a “special” key i.e shift. This had an easy solution of making an if statement to catch any “special” keys that would be pressed in the textWidget class.
4. Kajus also put a lot of work in the Functions class which provided everyone else in the group with relevant information that they needed. E.g An array of total cases/ array of daily cases etc. He designed many methods in this class that would allow for a fast response. Some of the functions were called in setup which would mean that for some of the screens there would be no wait time for the information to appear. Some of the methods involved user input and Kajus tried to make sure that the response time would be as fast as possible. The main issues that he had with this was trying to figure out the math behind getting the right start and end dates so as to not mess up the case numbers. After a while of testing he was able to get all of the functions working.
5. Kajus also got all of the images for the states and cropped them in such a way that they would not have square borders. He also came up with the idea of giving each state a slightly different colour so as to be able to distinguish between which state had been pressed.
6. Lastly Kajus also worked on the search screen. This screen allowed the user to type in a state, area and two dates and the result would be the total cases between those dates for that state/area. The main problem with this was the fact that most users won't type in the state with a capital letter. This meant that he had to make sure that each state/area had a capital letter but also for states like New York where there are two words he also had to make sure that they were capital letters. The fix for this was to convert the string to an array of chars and to go through each character and see if it is meant to be capitalized or not. This ended up working perfectly. Kajus also had to design a function that would search through the whole data set for the result. He ended up creating a function that would give the output almost instantaneously.