**5025211098 Timothy Hosia Budianto**
**5025211249 Daffa Saskara**

First all of all, thank you Sir for the assignment, this is our report that we would like to inform that we have been done your assignments and alhamdulillah we finished all the numbers, Sir. First, we made a template code because it seems almost similar like *wordtoSignature* template,command line program, etc.

In Number 1, the dictionary here actually doesn't get stored as the number or question's perquisites but it has some benefits that we got. Here, we take plenty of time to seek the desired output. After we run, program takes 0.05s. This number probably could easily increase/decrease because it has O(n * word[ ].size( )) times.

In number 2, The dictionary get stored on a list, so it will be more quick and efficient than other no storing methods. After we run the file, program took 0.04 s, it has O(word[].size()+n log n+log2(n)) time.

In Number 3, the dictionary get stored by using Map and Tree, they tend to have the same technique, but of course the two of the methods have different implementations. Using Map, we store all the words in map with its signature as key. When we run the file, it took the program 0.04s. It has O(word[].size()+log2(n)) times, bit more quicker than using the List. When using the Tree method, we store one by one character in the node so we can take less time than the previous method. It has O(word[].size()+s.length()) times.

The following is time compilation data :

No Storing
Time start 20:33:10,73
Time stop 20:33:10,78
Map
Time start 20:34:33,38
Time stop 20:34:33,42
List
Time start 20:36:31,98
Time stop 20:36:32,02
Tree
Time start 20:37:21,01
Time stop 20:37:21,03

In Number 4, We use java AWT and Swing to make the display

```java
// Keypad constructor class
public GUI(Container pane) {
 count = 0;
 // sets the size of the Keypad display
 pane.setPreferredSize(new Dimension(width: 320, height: 335));

 // initialize display to hold displayContent
 display = new JLabel(displayContent);
 display.setPreferredSize(new Dimension(width: 320, height: 25));
 // create lowered bevel border around the display
 display.setBorder(BorderFactory.createLoweredBevelBorder());
 // add the display to the panel
 pane.add(display, BorderLayout.PAGE_START);

 // initialize the buttonList
 buttonList = new ArrayList<JButton>(initialCapacity: 12);
 JPanel numberPanel = new JPanel();
 // set the numberPanel to have a 4row by 3col grid layout
 numberPanel.setLayout(new GridLayout(rows: 4,cols: 3,hgap: 0,vgap: 0));
 // set the size of the numberPanel
 numberPanel.setPreferredSize(new Dimension(width: 320,height: 260));
 // create the buttons and add them to the buttonList, properly displaying the numbers
  numButton = new JButton(numPadContent1);
  buttonList.add(numButton);
  numButton = new JButton(numPadContent2);
  buttonList.add(numButton);
  numButton = new JButton(numPadContent3);
  buttonList.add(numButton);
  numButton = new JButton(numPadContent4);
  buttonList.add(numButton);
  numButton = new JButton(numPadContent5);
  buttonList.add(numButton);
  numButton = new JButton(numPadContent6);
  buttonList.add(numButton);
  numButton = new JButton(numPadContent7);
```

SetPreferredSize to make new box with custom dimension and JLabel to make display content. Afterthat we make button with JButton with pre-set numpadcontent. Andthen add Buttonlist to the number panel.

```java
// update the display depending on clicked button(s)
public void actionPerformed(ActionEvent e) {
 String textThere = display.getText();
 String additionalText = "";
 // add clicked number button text to display
 for (int a = 0; a < buttonList.size(); a++) {
  if (e.getSource().equals(buttonList.get(a))) {
 if(a == 9){
   if(count == 0){
     savetext = removeWord(textThere, savetext);
     String converttext = savetext;
     numtotext = predictive.PredictivePrototype.signaturetoWords(converttext);
     String[] result = numtotext.toArray(new String[numtotext.size()]);
     textThere = "";
     additionalText = fulltext + result[count];
     fulltext = textThere;
   }else{
     String[] hasil = numtotext.toArray(new String[numtotext.size()]);
     int endingIndex = textThere.length() - hasil[0].length();
     String tmp = textThere.substring(beginIndex: 0, endingIndex);
     textThere = "";
     // String temptext = removeWord(tmp, hasil[(count)%(numtotext.size()-1)]);
     additionalText = tmp + hasil[count%(numtotext.size())];
     // fulltext = display.getText();
   }
   count++;
 }
 else if(a == 10){
   savetext = textThere;
   additionalText = numPadContent[a];
   fulltext = textThere + space;
   count = 0;
   numtotext.clear();
 }
```

In actionPerformed, the code detects the number button being clicked

User click number 9(*), and count == 0
Use removeword to save text before and send string to signatureTowords to generate text in dictionary. After that, save string in set<string> to arrayof string and print array string with index count and count++.
Else count != 0 change string display with substring print until string display - length of string in array of string. And print array string with index count and count++.

```
else if(a == 10){
  savetext = textThere;
  additionalText = numPadContent[a];
  fulltext = textThere + space;
  count = 0;
  numtotext.clear();
}
else if(a == 11){
  String tmp = textThere.substring(beginIndex: 0, textThere.length()-1);
  textThere = "";
  additionalText = tmp;
}
else{
  additionalText = numPadContent[a];
  count = 0;

}
 }
}
```

User click number 10(0) it sets count to 0 , and save display text.
User click number 11(#) change string display with substring print until string display - 1 and print it. And else add number clicked to display

```
Run | Debug
public static void main(String[] args) {

  //create and set up the window.
  JFrame frame = new JFrame(title: "Numeric Keyboard");

  frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

  //set up the content pane.
  frame.getContentPane().add(new GUI(frame));

  frame.pack();
  frame.setVisible(b: true);
}
```

In void main
Create and set up the window with Jframe and setup content pane with getcontentpane.

Timothy's and Daffa's contribution is equal with 50%-50% parts, because we work together, no one works alone and we understand the responsibility. The tools we are using is Visual Studio Code with Live Share Extension so we can edit the code together. The method we use to solve every problem is first create is a pseudocode which then we elaborate and actually write the code.