

Part 1:

```
Keyboard.java x
package lab3_1_package;
import java.util.ArrayList;

public class Keyboard{

    protected ArrayList<String> buttonCodeList;
    protected ArrayList<String> buttonValueList;
    protected String displayedText = "";

    public Keyboard() {}
    protected enum SpecialButtons{Space, Backspace, Enter}

    protected void showKeyboard(int row, int col) {
        TableView<String> table = new TableView<String>(getAllButtonCode(),getAllButtonValue());
        table.viewTable(row, col);
    }

    protected int getButtonCodeIndex(String code) { return buttonCodeList.indexOf(code); }
    public void buttonPressed(String button) {
        int index = getButtonCodeIndex(button);
        if(index != -1) {
            String value = getAllButtonValue().get(index);
            if("Space".equals(value)) { space(); }
            }else if ("Backspace".equals(getAllButtonValue().get(index))) { backspace(); }
            }else if ("Enter".equals(getAllButtonValue().get(index))) { newLine(); }
            }else { addToDisplayedText(value); }
        }

        System.out.println("Text entered: \n" + "-----" );
        System.out.println(getDisplayedText());
        System.out.println("-----" );
    }

    protected void backspace() { addToDisplayedText("\b"); }
    protected void space() { addToDisplayedText(" "); }
    protected void newLine() { addToDisplayedText("\n"); }

    protected ArrayList<String> getAllButtonCode() { return buttonCodeList; }
    protected ArrayList<String> getAllButtonValue() { return buttonValueList; }

    protected String getDisplayedText() { return displayedText; }
    protected void addToDisplayedText(String text) { displayedText += text; }
```

```
QWERTY.java x

import java.util.ArrayList;

public class QWERTY extends Keyboard {
    public QWERTY() {
        buttonCodeList = new ArrayList<>() {{
            for (int i = 1; i <= 39; i++) {
                add("" + i);
            }
        }};
        buttonValueList = new ArrayList<>() {{
            add(""+SpecialButtons.Space);add(""+SpecialButtons.Backspace);add(""+SpecialButtons.Enter);
            add("a");add("b");add("c");add("d");add("e");add("f");add("g");add("h");add("i");
            add("j");add("k");add("l");add("m");add("n");add("o");add("p");add("q");add("r");
            add("s");add("t");add("u");add("v");add("w");add("x");add("y");add("z");

            for (int i = 0; i <= 9; i++) {
                add("" + i);
            }
        }};
        super.showKeyboard( row: 10, col: 10);
    }
}
```

```
Calculator.java ×  
  
package lab3_1_package;  
  
import java.util.ArrayList;  
  
public class Calculator extends Keyboard {  
    public Calculator(){  
        buttonCodeList = new ArrayList<>(){  
            for(int i = 0; i <= 19; i++) { add("" + i); }  
        };  
        buttonValueList = new ArrayList<>(){  
            for(int i = 0; i <= 9; i++) { add("" + i); }  
  
            add("+");add("-");add("/");add("*");add("=");add("(");add(")");  
            add(""+SpecialButtons.Space);add(""+SpecialButtons.Backspace);add(""+SpecialButtons.Enter);  
        };  
        super.showKeyboard( row: 10, col: 10);  
    }  
  
    public void buttonPressed(String button){  
        int index = getButtonCodeIndex(button);  
        if(index != -1){  
            String value = getAllButtonValue().get(index);  
            if("=".equals(value)){  
                String[] text = getDisplayedText().split( regex: " ");  
                int ans = EvaluateString.evaluate(text[text.length - 1]);  
                addToDisplayedText(" = " + ans + "\n");  
                System.out.println("Text entered: \n" + "-----" );  
                System.out.println(super.getDisplayedText());  
                System.out.println("-----" );  
            }  
            else{ super.buttonPressed(button); }  
        }  
    }  
}
```

```
OperatingSystem x
"C:\Program Files\Java\jdk-15.0.2\bin\java.exe" -Didea.launcher
1-QWERTY
2-Calculator
Please choose one of the two keyboards to run:
1
1->Space      |11->h |21->r |31->1 |
2->Backspace  |12->i |22->s |32->2 |
3->Enter      |13->j |23->t |33->3 |
4->a          |14->k |24->u |34->4 |
5->b          |15->l |25->v |35->5 |
6->c          |16->m |26->w |36->6 |
7->d          |17->n |27->x |37->7 |
8->e          |18->o |28->y |38->8 |
9->f          |19->p |29->z |39->9 |
10->g         |20->q |30->0 |
Please enter the code of the button or -1 to exit: -1
1-QWERTY
2-Calculator
Please choose one of the two keyboards to run:
2
0->0 |10->+      |
1->1 |11->-      |
2->2 |12->/      |
3->3 |13->*      |
4->4 |14->=      |
5->5 |15->(      |
6->6 |16->)      |
7->7 |17->Space  |
8->8 |18->Backspace |
9->9 |19->Enter  |
Please enter the code of the button or -1 to exit:
```

Part 2:

Keyboard Class

```
package lab3_2_polymorph;
import static lab3_2_polymorph_OS.OperatingSystem.*;
import java.util.ArrayList;

public class Keyboard {
    protected ArrayList<KeyboardButton> buttons = new ArrayList<>();
    protected String displayedText = "";

    public Keyboard(int firstCode, String[] keyboardButtonsValues, String[] generalButtonsValues,
        String[] generalButtonsActions, SpecialButtons[] SpecialButtonsList) {

        int length1 = keyboardButtonsValues.length;
        int length2 = generalButtonsValues.length;
        int length3 = SpecialButtonsList.length;

        // add button objects to list
        if(firstCode == 1) { //QWERTY
            for(int i = firstCode; i <= length1; i++) {
                buttons.add(new KeyboardButton(keyboardButtonsValues[i-1], i));
            }
            for(int j = (firstCode + length1); j <= (length2 + length1); j++) {
                buttons.add(new NamedButtons(generalButtonsValues[j-1-length1], j,
                    generalButtonsActions[j-1-length1]));
            }
            int length = length1 + length2;
            for(int k = (firstCode + length); k <= (length3 + length); k++) {
                buttons.add(new SpecialActionButton(SpecialButtonsList[k-1-length], k));
            }
        } else { //Calculator
            for(int i = firstCode; i < (length1); i++) {
                buttons.add(new KeyboardButton(keyboardButtonsValues[i], i));
            }
            for(int j = (firstCode + length1); j < (length2 + length1); j++) {
                buttons.add(new NamedButtons(generalButtonsValues[j-length1], j,
                    generalButtonsActions[j-length1]));
            }
            int length = length1 + length2;
            for(int k = (firstCode + length); k < (length3 + length); k++) {
                buttons.add(new SpecialActionButton(SpecialButtonsList[k-length], k));
            }
        }
    }
}
```

```
protected void showKeyboard(int row, int col) {
    TableView table = new TableView(getAllButtonCode(),getAllButtonValue());
    table.viewTable(row, col);
}

protected String getButtonValue(String code) {
    for(KeyboardButton button : buttons) {
        if(code.equals(Integer.toString(button.getCode())) {
            return button.getName();
        }
    }
    return null;
}

public void buttonPressed(String press) {
    String val = getButtonValue(press);
    if(val != null) {
        int index = Integer.parseInt(press);
        KeyboardButton button = buttons.get(index-1);
        if (button instanceof NamedButtons) {
            this.displayedText += ((NamedButtons) button).getAction();
        } else if (button instanceof SpecialActionButton) {
            this.displayedText += ((SpecialActionButton) button).act(val);
        } else {
            this.displayedText += button.getName();
        }
    } else
        System.out.println("Button pressed isn't on keyboard");
    this.showDisplayedText();
}
```

```
protected ArrayList<String> getAllButtonCode() {  
    return new ArrayList<>() {{  
        for(KeyboardButton button : buttons) {  
            int code = button.getCode();  
            add("" + Integer.toString(code));  
        }  
    }};  
}  
  
protected ArrayList<String> getAllButtonValue() {  
    return new ArrayList<>() {{  
        for(KeyboardButton button : buttons) {  
            String name = button.getName();  
            add("" + name);  
        }  
    }};  
}  
  
protected String getDisplayedText() { return displayedText; }  
protected void showDisplayedText() {  
    System.out.println("Text entered: \n" + "-----" );  
    System.out.println(getDisplayedText());  
    System.out.println("-----" ); }  
}
```

QWERTY class

```

Calculator.java × QWERTY.java × Keyboard.java × SpecialActionButton.java × KeyboardButton.java ×
package lab3_2_polymorph;

import static lab3_2_polymorph_OS.OperatingSystem.*;

public class QWERTY extends Keyboard {
    public QWERTY(int firstCode, String[] keyboardButtonsValues, String[] generalButtonsValues,
        String[] generalButtonsActions, SpecialButtons[] SpecialButtonsList) {
        super(firstCode, keyboardButtonsValues, generalButtonsValues,
            generalButtonsActions, SpecialButtonsList);
        super.showKeyboard( row: 10, col: 10);
    }
}


```

Calculator class

```

Calculator.java × Keyboard.java × SpecialActionButton.java × KeyboardButton.java ×
1 package lab3_2_polymorph;
2
3 import lab3_2_polymorph_OS.OperatingSystem;
4
5 public class Calculator extends Keyboard {
6     public Calculator(int firstCode, String[] keyboardButtonsValues, String[] generalButtonsValues,
7         String[] generalButtonsActions, OperatingSystem.SpecialButtons[] SpecialButtonsList) {
8         super(firstCode, keyboardButtonsValues, generalButtonsValues,
9             generalButtonsActions, SpecialButtonsList);
10        super.showKeyboard( row: 10, col: 10);
11    }
12    public void buttonPressed(String button) {
13        int index = Integer.parseInt(button);
14        if(index != -1) {
15            String value = super.getAllButtonValue().get(index);
16            if("=".equals(value)) {
17                String[] text = getDisplayedText().split( regex: " ");
18                int ans = EvaluateString.evaluate(text[text.length - 1]);
19                super.displayedText += (" = " + ans + "\n");
20                super.showDisplayedText();
21            } else {
22                super.buttonPressed(Integer.toString( i: index + 1));
23            }
24        }
25    }
26 }

```


KeyboardButton classThe image shows a screenshot of an IDE with four tabs: Calculator.java, Keyboard.java, SpecialActionButton.java, and KeyboardButton.java. The KeyboardButton.java tab is active, displaying the following Java code:

```
1 package lab3_2_polymorph;
2
3 public class KeyboardButton {
4
5     protected String name;
6     protected int code;
7     protected String impliesSymbol = "->";
8
9     public KeyboardButton(String button, int code) {
10         setName(button);
11         setCode(code);
12     }
13     public String checkCode(int code) {
14         return Integer.toString(getCode()) + impliesSymbol + getName();
15     }
16     public String getName() { return name; }
17     public void setName(String name) { this.name = name; }
18
19     public int getCode() { return code; }
20     public void setCode(int code) { this.code = code; }
21
22     @Override
23     public String toString() {
24         return "KeyboardButton{" +
25             "name='" + name + '\'' +
26             ", code=" + code +
27             ", impliesSymbol='" + impliesSymbol + '\'' +
28             '}';
29     }
30 }
```

NamedButton class

```
package lab3_2_polymorph;

public class NamedButtons extends KeyboardButton {

    private String action;

    public NamedButtons(String button, int code, String action) {
        super(button, code);
        setAction(action);
    }

    public String getAction() { return action; }

    public void setAction(String action) {
        this.action = action;
    }

    @Override
    public String toString() {
        return "NamedButtons{" +
            "action='" + action + '\'' +
            '}';
    }
}
```

SpecialActionButton class

```
package lab3_2_polymorph;

import static lab3_2_polymorph_OS.OperatingSystem.*;

public class SpecialActionButton extends KeyboardButton{

    private SpecialButtons buttonType;

    public SpecialActionButton(SpecialButtons button, int code) {
        super(button.name(), code);
        buttonType = button;
    }

    public String act(String value) {
        if (buttonType.name().equals(value)) {
            return backSpace(value);
        }
        return "$%&";
    } //if there was another special button it would be here in switch case

    private String backSpace(String value) {
        if (buttonType.name().equals(value)) {
            return "\b";
        }
        return "$%&";
    }
}
```

```
OperatingSystem (1) x
al machno

-----
Please enter the code of the button or -1 to exit: 21
Text entered:
-----
f 9
hayan ee364
al machnou
-----
Please enter the code of the button or -1 to exit: 11
Text entered:
-----
f 9
hayan ee364
al machnouk
-----
Please enter the code of the button or -1 to exit: -1
1-QWERTY
2-Calculator
Please choose one of the two keyboards to run:
2

0->0 | 10->+ |
```

6: TODO Spark monitoring Terminal Zeppelin 0: M

Part 3:

Inheritance aids in the reuse of code. The parent class's code can be reused by the child class without having to rewrite it.

Because the primary code does not need to be written repeatedly, inheritance can save time and effort.

Inheritance gives a straightforward model structure that is simple to comprehend.

We can override the parent class's methods with inheritance so that the child class can create a meaningful implementation of the base class method.

As a result, all of the above will ultimately result in less maintenance work and lower overhead costs.

In addition, the base class can choose to keep some data private so that the derived class cannot access or modify it except if it is of protected type.

As an example of inheritance, consider the relationship between a parent and a kid. The qualities of parents, such as hands, legs, eyes, and nose, as well as behaviors such as walk, talk, eat, and sleep, are inherited in the child, allowing the child to use/access these properties and behaviors whenever necessary.

Both parent and child can have particular or private properties in addition to some common qualities and behavior. For example, parent and child can have specific properties such as blood group, date of birth, and specific behavior such as one playing cricket or another activity while the other does not.

