1.Illustrate the working mechanism of Checkbox and Radio button by considering all possible constructors.

Checkbox and Radio button are HTML elements that allow users to select options from a list. The difference is that Checkbox allows multiple selections, while Radio button allows only one selection.

To create a Checkbox or a Radio button, you need to use the <input> tag with the type attribute set to “checkbox” or “radio”. You also need to use the <label> tag to associate a text with each option. The <input> and <label> tags should be siblings, not nested. You can use the id and for attributes to link them together.

For example, this is how you can create a Checkbox with two options:

<div class="form-check">

<input class="form-check-input" type="checkbox" value="" id="option1">

<label class="form-check-label" for="option1">

Option 1

</label>

</div>

<div class="form-check">

<input class="form-check-input" type="checkbox" value="" id="option2">

<label class="form-check-label" for="option2">

Option 2

</label>

</div>

And this is how you can create a Radio button with two options:

<div class="form-check">

<input class="form-check-input" type="radio" name="options" id="option1" checked>

<label class="form-check-label" for="option1">

Option 1

</label>

</div>

<div class="form-check">

<input class="form-check-input" type="radio" name="options" id="option2">

<label class="form-check-label" for="option2">

Option 2

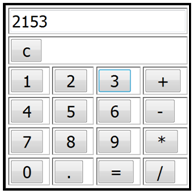
</label>

</div>

Notice that the Radio button options have the same name attribute, which ensures that only one of them can be selected at a time. You can also use the checked attribute to specify which option is selected by default.

To customize the appearance of the Checkbox and Radio button, you can use CSS to style the .form-check-input and .form-check-label classes. You can also use pseudo-elements like :before, :after, :hover and :checked to create custom icons and indicators[1](https://getbootstrap.com/docs/5.0/forms/checks-radios/)[2](https://www.w3schools.com/howto/howto_css_custom_checkbox.asp).

2.Develop a swing application to implement simple calculator as shown in the following figure.



|  |
| --- |
| mport java.awt.event.\*;  import javax.swing.\*;  import java.awt.\*;  class calculator extends JFrame implements ActionListener {      // create a frame      static JFrame f;        // create a textfield      static JTextField l;        // store operator and operands      String s0, s1, s2;        // default constructor      calculator()      {          s0 = s1 = s2 = "";      }        // main function      public static void main(String args[])      {          // create a frame          f = new JFrame("calculator");            try {              // set look and feel              UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());          }          catch (Exception e) {              System.err.println(e.getMessage());          }            // create a object of class          calculator c = new calculator();            // create a textfield          l = new JTextField(16);            // set the textfield to non editable          l.setEditable(false);            // create number buttons and some operators          JButton b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, be, beq, beq1;            // create number buttons          b0 = new JButton("0");          b1 = new JButton("1");          b2 = new JButton("2");          b3 = new JButton("3");          b4 = new JButton("4");          b5 = new JButton("5");          b6 = new JButton("6");          b7 = new JButton("7");          b8 = new JButton("8");          b9 = new JButton("9");            // equals button          beq1 = new JButton("=");            // create operator buttons          ba = new JButton("+");          bs = new JButton("-");          bd = new JButton("/");          bm = new JButton("\*");          beq = new JButton("C");            // create . button          be = new JButton(".");            // create a panel          JPanel p = new JPanel();            // add action listeners          bm.addActionListener(c);          bd.addActionListener(c);          bs.addActionListener(c);          ba.addActionListener(c);          b9.addActionListener(c);          b8.addActionListener(c);          b7.addActionListener(c);          b6.addActionListener(c);          b5.addActionListener(c);          b4.addActionListener(c);          b3.addActionListener(c);          b2.addActionListener(c);          b1.addActionListener(c);          b0.addActionListener(c);          be.addActionListener(c);          beq.addActionListener(c);          beq1.addActionListener(c);            // add elements to panel          p.add(l);          p.add(ba);          p.add(b1);          p.add(b2);          p.add(b3);          p.add(bs);          p.add(b4);          p.add(b5);          p.add(b6);          p.add(bm);          p.add(b7);          p.add(b8);          p.add(b9);          p.add(bd);          p.add(be);          p.add(b0);          p.add(beq);          p.add(beq1);            // set Background of panel          p.setBackground(Color.blue);            // add panel to frame          f.add(p);            f.setSize(200, 220);          f.show();      }      public void actionPerformed(ActionEvent e)      {          String s = e.getActionCommand();            // if the value is a number          if ((s.charAt(0) >= '0' && s.charAt(0) <= '9') || s.charAt(0) == '.') {              // if operand is present then add to second no              if (!s1.equals(""))                  s2 = s2 + s;              else                  s0 = s0 + s;                // set the value of text              l.setText(s0 + s1 + s2);          }          else if (s.charAt(0) == 'C') {              // clear the one letter              s0 = s1 = s2 = "";                // set the value of text              l.setText(s0 + s1 + s2);          }          else if (s.charAt(0) == '=') {                double te;                // store the value in 1st              if (s1.equals("+"))                  te = (Double.parseDouble(s0) + Double.parseDouble(s2));              else if (s1.equals("-"))                  te = (Double.parseDouble(s0) - Double.parseDouble(s2));              else if (s1.equals("/"))                  te = (Double.parseDouble(s0) / Double.parseDouble(s2));              else                  te = (Double.parseDouble(s0) \* Double.parseDouble(s2));                // set the value of text              l.setText(s0 + s1 + s2 + "=" + te);                // convert it to string              s0 = Double.toString(te);                s1 = s2 = "";          }          else {              // if there was no operand              if (s1.equals("") || s2.equals(""))                  s1 = s;              // else evaluate              else {                  double te;                    // store the value in 1st                  if (s1.equals("+"))                      te = (Double.parseDouble(s0) + Double.parseDouble(s2));                  else if (s1.equals("-"))                      te = (Double.parseDouble(s0) - Double.parseDouble(s2));                  else if (s1.equals("/"))                      te = (Double.parseDouble(s0) / Double.parseDouble(s2));                  else                      te = (Double.parseDouble(s0) \* Double.parseDouble(s2));                    // convert it to string                  s0 = Double.toString(te);                    // place the operator                  s1 = s;                    // make the operand blank                  s2 = "";              }                // set the value of text              l.setText(s0 + s1 + s2);          }      }  } |