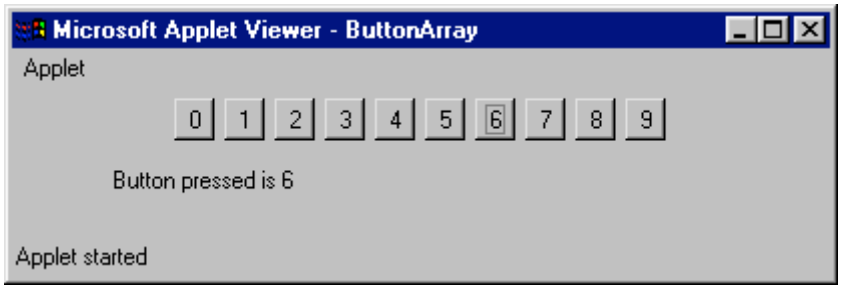
Java - Arrays of Controls

C&G criteria: 2.1.2, 2.2.1, 2.2.2, 2.3.2.

This exercise extends existing knowledge to simulate a calculator in Java. The code to create and reference controls as individual objects is not short. Where a large number of controls are involved it becomes very attractive to set them up as an array.

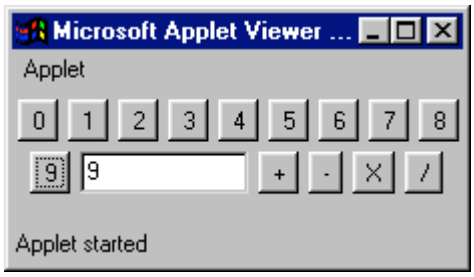
A complete program to create and test an array of Buttons is shown below.

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class ButtonArray extends Applet implements ActionListener
{
	Button []digit = new Button[10];//declare the array
	int butNumber;

	public void init()
	{
		for(int x = 0;x<digit.length;x++)
		{//each Button has to be created separately
			//the text of each button will be 'x'
			digit[x]=new Button(Integer.toString(x));
			add(digit[x]);
			digit[x].addActionListener(this);
		}
	}
	public void paint(Graphics g)
	{//output to check this all works
		g.drawString("Button pressed is " + String.valueOf(butNumber),50,50);
	}
	public void actionPerformed(ActionEvent e)
	{// get the label of the button - changed from String to int
		butNumber = Integer.parseInt(e.getActionCommand());
		repaint();
	}
}
```



Setting up a calculator interface

For the moment ignore effect of a decimal point and any maths operations. More buttons are needed to select the mathematical operators (effects to be coded later) and a TextField will handle output.



The operator Buttons (+, - , X, /) can be set up as an array like the digits but the array number (0 – 3) is not what is required for the Button label. Suitable labels can be assigned during the same loop that adds the operator Buttons through the use of a Switch clause. The syntax is the same as for C programming.

```
for(int x = 0;x<calc.length;x++)
{//maths Buttons are another array called calc[4] – these must be added to the program
        switch(x)
        {//set the labels for maths
                case 0:
                        calc[x].setLabel("+");
                        break;
                case 1:
                        calc[x].setLabel("-");
                        break;
                case 2:
                        calc[x].setLabel("X");
                        break;
                case 3:
                        calc[x].setLabel("/");
                        break;
        }
}
```

Output from the program will be set to the TextField not drawn on the form. This code will test the Buttons work in actionPerformed(ActionEvent e).

```
for(int x = 0;x<digit.length;x++)
{
        if(e.getSource()==digit[x])
                output.setText(e.getActionCommand());
}
for(int x = 0;x<calc.length;x++)
{
        if(e.getSource()==calc[x])
        ; // action code to write
}
```

Operator code

1. The calculator must be able to place a chain of numbers in the TextField. As more buttons are pressed the number in the TextField increases by 10 and the value of the current button is added to the total. The value stored in the TextField can be assigned to an int. If the getText() method of the TextField is used Java falls down when there is no text in the TextField. Use Integer.parseInt() and String.ValueOf() to change between Strings and int.

2. Have the + and – Buttons act like those on a calculator in keeping a constant total without having to use =. So if 1 if pressed followed by + then 2 the TextField shows 3. This will require another switch routine for the operator array in actionPerformed(ActionEvent e). The total can be recorded by another int separate from that used for the number in the TextArea.

3. Add an equals Button to the operator array. Code the \ and X Buttons, the total variable will now have to be a float. Create and code a clear Button, this can be part of the operator array.