Sort Animator

This program animates various sorting algorithms on the screen using bars. It uses a slightly different type of buttons to allow the applet to draw on the screen and to have buttons at the same time.

The Starter Code is at the end of the file.

Sort Animator #1 – Selection sort

This code will make the selection sort button work.

Add in the selection sort method. It is cut and paste-able and is on the right.

Then, call the method as shown below. It cannot be cut and pasted, I used a picture.

Run your code to see if it sorts.

```
public void selectionSort (int a[]){
    for (int left = a.length - 1 ; left > 0 ; left--){
        int max = 0;
        for (int i = 1 ; i <= left ; i++){
            if (a [max] < a [i])
                max = i;
        }
        int temp = a [max];
        a [max] = a [left];
        a [left] = temp;
        printArray ();
    }
}
```

```
public boolean action (Event e, Object o)
{
    if (e.target == select)
    {
        selectionSort (a);
        printArray ();
    }
    else if (e.target == bubble)
    {
    }
```

Sort Animator #2 – Unsort

Code a method to unsort your data so that you can see it sorted again. Follow the comments to complete it.

```
public void unsort ()
{
    //randomly swap two elements 60 times to un-order the array
    for (int i = 0 ; i < 60 ; i++)
    { //TO DO: get two random numbers between 0 and a.length
        //int b = random number between 0 and a.length
        //int c = random number between 0 and a.length

        //TO DO: swap the things in a[b] and a[c]
        //there will be three lines in your swap

    }
}
```

When you have finished coding the unsort algorithm, call it in the appropriate button.

```
public boolean action (Event e, Object o)
{
    if (e.target == select)
    {
        selectionSort (a);
        printArray ();
    }
    else if (e.target == bubble)
    {
    }
    else if (e.target == quick)
    {
    }
    else if (e.target == shaker)
    {
    }
    else if (e.target == reset)
    {
        unsort ();
        printArray ();
    }
    return true;
}
```

Sort Animator #3 – All the other sorts

Paste in the methods for the other sorts below. Then, call the methods in each button. Note that quicksort is called differently. It's call is highlighted below.

```
else if (e.target == bubble)
{
    bubbleSort (a);
    printArray ();
}
else if (e.target == quick)
{
    quicksort (a, 0, a.length - 1);
    printArray ();
}
else if (e.target == shaker)
{
    shakerSort (a);
    printArray ();
}
```

Sorting Methods:

```
public void shakerSort (int[] array)
{
    for (int i = 0 ; i < array.length / 2 ; i++)
    {
        boolean swapped = false;
        for (int j = i ; j < array.length - i - 1 ; j++)
        {
            if (array [j] > array [j + 1])
            {
                int tmp = array [j];
                array [j] = array [j + 1];
                array [j + 1] = tmp;
                swapped = true;
                printArray ();
            }
        }
        for (int j = array.length - 2 - i ; j > i ; j--)
        {
            if (array [j] < array [j - 1])
            {
                int tmp = array [j];
                array [j] = array [j - 1];
                array [j - 1] = tmp;
                swapped = true;
                printArray ();
            }
        }
        if (!swapped)
            break;
    }
}

public void bubbleSort (int a[])
{
    int n = a.length;
    for (int i = 0 ; i < n - 1 ; i++)
    {
        for (int j = 0 ; j < n - 1 - i ; j++)
        {
            if (a [j + 1] < a [j])
            {
                int temp = a [j];
                a [j] = a [j + 1];
                a [j + 1] = temp;
                printArray ();
            }
        }
    }
}
```

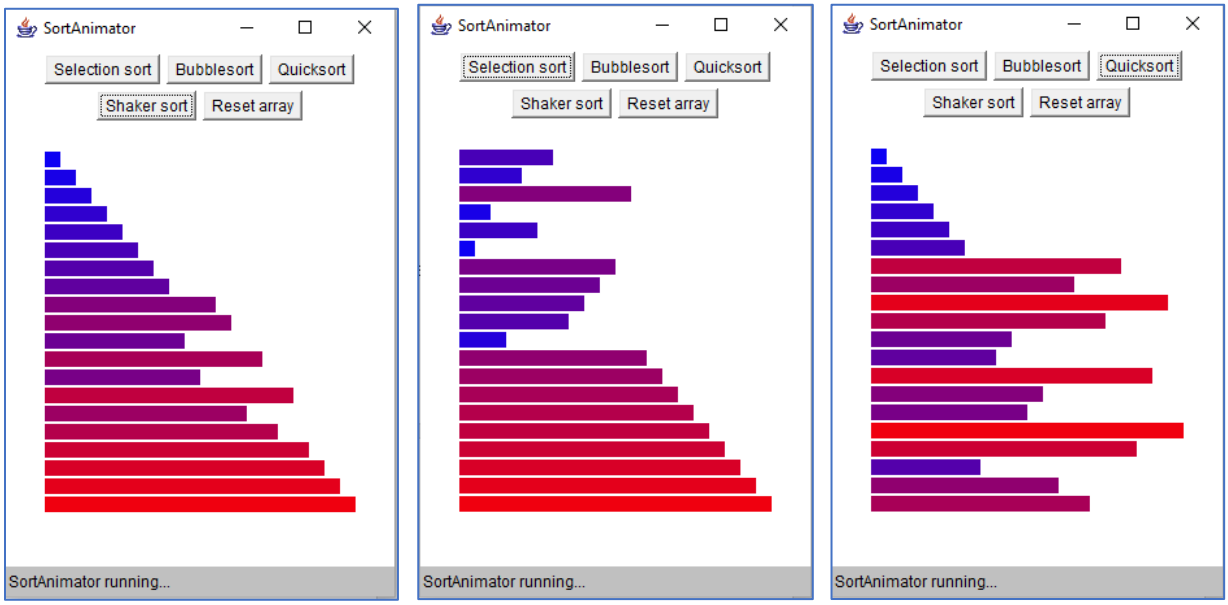
```
public void quicksort (int a[], int lo0, int hi0)
{
    int lo = lo0;
    int hi = hi0;

    if (lo >= hi)
    {
        return;
    }
    int mid = a [(lo + hi) / 2];
    while (lo < hi)
    {
        while (lo < hi && a [lo] < mid)
        {
            lo++;
        }
        while (lo < hi && a [hi] > mid)
        {
            hi--;
        }
        if (lo < hi)
        {
            int T = a [lo];
            a [lo] = a [hi];
            a [hi] = T;
        }
    }
    if (hi < lo)
    {
        int T = hi;
        hi = lo;
        lo = T;
    }
    printArray ();
    quicksort (a, lo0, lo);
    quicksort (a, lo == lo0 ? lo + 1:
    lo, hi0);

}
```

Run your code to verify it works.

These show shaker, selection and quicksort.



Starter Code

```
import java.awt.*;
import java.applet.*;
public class SortAnimator extends Applet
{
    Button select, bubble, quick, shaker;
    int a[] = {17, 2, 7, 19, 12, 1, 16, 11, 5, 18, 14, 6, 20, 8, 9, 13, 4, 10, 15, 3};
    Button reset;
    public void init ()
    {
        select = new Button ("Selection sort");
        add (select);
        bubble = new Button ("Bubblesort");
        add (bubble);
        quick = new Button ("Quicksort");
        add (quick);
        shaker = new Button ("Shaker sort");
        add (shaker);
        reset = new Button ("Reset array");
        add (reset);
        resize (300, 400);
    }

    public boolean action (Event e, Object o)
    {
        if (e.target == select)
        {

        }
        else if (e.target == bubble)
        {

        }
        else if (e.target == quick)
        {

        }
        else if (e.target == shaker)
        {

        }
        else if (e.target == reset)
        {

        }
        return true;
    }



    public void paint (Graphics g)
    {
        drawArray ();
    }


    public void drawArray ()
    {
        Graphics g = getGraphics ();
        //Blank out old array
        g.setColor (Color.white);
        g.fillRect (30, 70, 500, 500);
        //Draw rectangles for array
        int y = 80;
        for (int i = 0 ; i < a.length ; i++)
        {
            //new colour based on length
            g.setColor (new Color (a [i] * 12, 0, 255 - a [i] * 12));
            g.fillRect (30, y, (a [i] * 12), 12);
            y += 14;
        }
    }

    public void printArray ()
    {
        drawArray ();
        //Waste time
        for (int i = 0 ; i < 200000000 ; i++)
            ;
    }
}
```