Reading:
1. Dale, Chapter 11 (Sections 3 and 4)

Objectives:
1. Introduce sorting

Concepts:
1. Sorting
1. Introduce sorting
   - Discuss the purpose of sorting
   - Discuss different sorting algorithms:
     o Bubbla sort
     o Heap sort
     o Quick sort
     o Merge sort
     o Selection sort
   - Demonstrate sorting demos in class
   - Discuss sorting lists as well as sorting arrays
Objective:
1. Practice sorting arrays using bubble sort

Student Activities:
1. In bubble sort the smaller value gradually “bubble” their way to the top of the array like air bubbles rising in water, while the larger values sink to the bottom of the array. The technique is to make several passes through the array. On each pass, successive pair of elements are compared. If a pair is in increasing order (or the values are identical), we leave the values as they are. If a pair is in decreasing order, their values are swapped in the array. Given the following program, fill in the missing code and show what the output would look like:

```java
import java.awt.Graphics;
import java.applet.Applet;

public class BubbleSort extends Applet {
    int a[] = {2, 6, 4, 8, 10, 12, 89, 68, 45, 37};
    int hold; // temporary holding area for swap

    public void paint ( Graphics g )
    {
        print ( g, “Data items in original order”, a, 25, 25);
        sort();
        print ( g, “Data items in ascending order”, a, 25, 55);
    }

    public void sort()
    {
        // FILL in the missing code
    }

    public void print ( Graphics g, String head, int b[], int x, int y)
    {
        g.drawString ( head, x, y);
        x += 15;
        y += 15;

        for (int l = 0; l < b.length; l++) {
            g.drawString ( String.valueOf ( b[l] ), x, y);
            x += 20;
        }
    }
}
```
The missing code is:

```java
public void sort()
{
    for ( int pass = 1; pass < a.length; pass++) // passes
        for ( int j = 0; j < a.length – 1; j++)  // one pass
            if ( a[j] > a[j+1] ) {   // one comparison
                hold = a[j];   // one swap
                a[j] = a[j+1];
                a[j+1] = hold;
            }
}
```

The output is supposed to be:

Data items in original order
2 6 4 8 10 12 89 68 45 37
Data items in ascending order
2 4 6 8 10 12 37 45 68 89