CS1007 lecture #10 notes	arrays (1).
tue 8 oct 2002	• used to associate multiple instances of the same type of variable
 NEWS error quiz # 1, question 1d if you got the question WRONG, bring the quiz to class on thu 10 oct for a one-time regrade arrays (one-dimensional) finding array minimum and maximum sorting big-Oh 2-dimensional arrays (preview) reading: <i>ch 5.1-5.7</i> 	 the "[]" indicates it's an array we can have arrays of anything (i.e., other data types) one example we've already used is String[], which is an array of String visualize an array as a sequence of boxes, contiguous in the computer's memory, where each box stores one instance of the type of data associated with that array: i i i i i i i i i i i i i i i i i i i
cs1007-fall2002-sklar-lect10 1	cs1007-fall2002-sklar-lect10 2
arrays (2).	arrays (3).
 to use an array, first you must declare it: int[] A; then you must instantiate it: A = new int[10]; or you can do both of these in one step: int[] A = new int[10]; then you can access its elements: A[4] (index=4, which is the 5th item in the array) you can use this accessed item just like any single data element of that type, in this case an int the number of items in the array is the variable A.length 	<pre>• here's an example that stores in an array 5 random numbers between 0 and 100: public class ex10a { public static void main(String[] args) { int[] A = new int[5]; for (int i=0; i</pre>

arrays — finding the minimum.

```
public class ex10b {
   public static void main( String[] args ) {
      int[] A = new int[5];
      for ( int i=0; i<A.length; i++ ) {
            A[i] = (int)(Math.random()*100);
      }
      int min = A[0];
      for ( int i=1; i<A.length; i++ ) {
            if ( A[i] < min ) {
                min = A[i];
            }
            } // end for i
      System.out.println( "the minimum is: " + min );
      } // end of main()
      } // end of class ex10b
</pre>
```

arrays — finding the maximum.

```
public class ex10c {
   public static void main( String[] args ) {
      int[] A = new int[5];
      for ( int i=0; i<A.length; i++ ) {
            A[i] = (int)(Math.random()*100);
      }
      int max = A[0];
      for ( int i=1; i<A.length; i++ ) {
            if ( A[i] > max ) {
                max = A[i];
            }
            // end for i
            System.out.println( "the maximum is: " + max );
      } // end of main()
} // end of class ex10c
```

cs1007-fall2002-sklar-lect10

sorting (1).

- sorting is one of the classic tasks done in computer programming
- the basic idea with sorting is to rearrange the elements in an array so that they are in a specific order usually ascending or descending, in numeric or alphabetic order
- we will discuss 4 sorting algorithms (i.e., methods for sorting):
 - blort sort

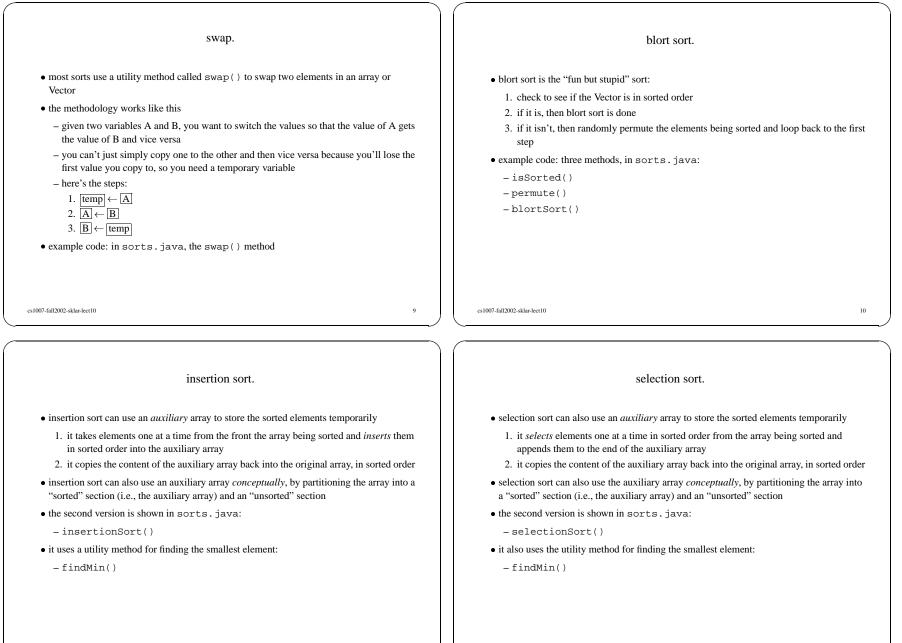
cs1007-fall2002-sklar-lect10

- insertion sort
- selection sort
- bubble sort

sorting (2).

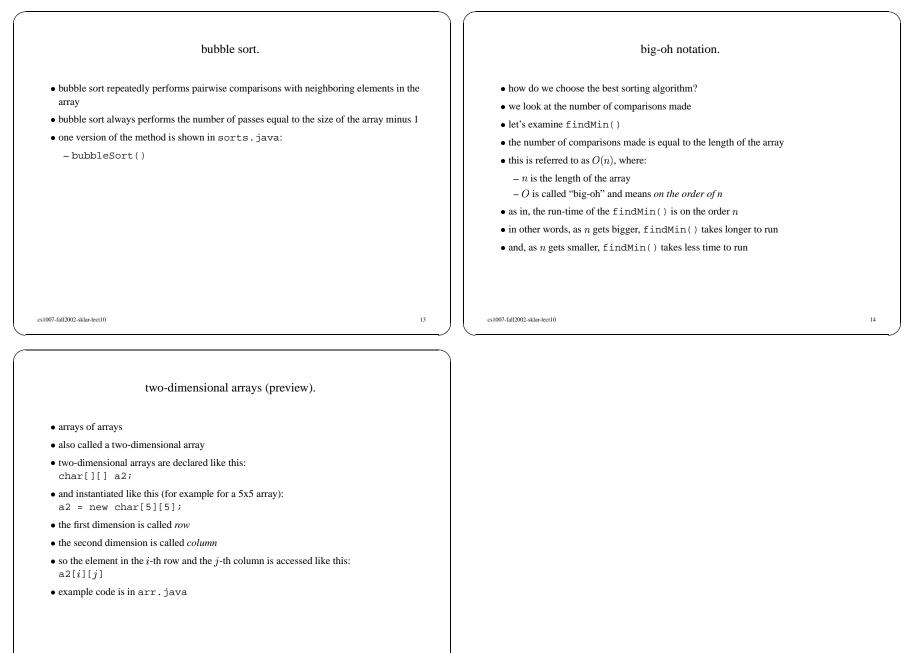
- some sorts require an extra "auxiliary" array during sorting
 - the elements are moved from the original array into the auxiliary array, one at a time
 - at the end of the sort, the auxiliary array contains all the elements in sorted order
 - the final step is to copy the elements from the auxiliary array back into the original array
 - insertion and selection sorts can be done this way
- some sorts do not use an auxiliary array during sorting, but just move the elements around within the original array
 - these sorts involve the use of a swap() function, to switch the locations of two entries in the array
 - insertion and selection sorts can be done this way too
 - blort and bubble sorts are always done this way

5



cs1007-fall2002-sklar-lect10

11



cs1007-fall2002-sklar-lect10

15