1. Index value 2.
2. `int[] quantities = new int[20];`
3. `int[] heights = {1.65, 2.15, 4.95};`
4.
```
for (int element : grades)
    {
        System.out.println(element);
    }
```

6. The main difference between passing an entire array vs. just passing an element is the way the data changes. When you pass an array into a method, the data in the array can be changed by the value of the method parameter (which is an array) it was passed to. This is because when you pass an array, the array parameter of the method is referencing the array you pass, so it can modify its data outside of the method. On the other hand, passing an element of an array into a method does not let the parameter of the method reference the original element and modify it. The element that the method receives is a copy of the element that was passed.

7. Offset array indexes are required in some cases because they allow you to create an array within a range of numbers without it being an unnecessarily long array. For example, in the skillbuilder CountLetters, we use an offset array index because our array ranges from 65 - 90 (A - Z). If we didn't offset the index, it would start at 0, and values 0 - 64 would never be used. An offset array index allows us to downsize arrays.

8. It will output the character 'i'.

10. An example of when a dynamic array might be a better choice is when you have a list of people in a class. The number of people in a class can change throughout the semester. People might move and leave the school, or someone might switch from one class to another, or drop a class. It is important that your program that includes a class list has a dynamic array to handle the growing/shrinking of the class size.

11. It uses the equals() method from the object's class to determine equality between the object passed and the array element.

12. They can be compared using the compareTo() method (works for both integer and double variables), which returns either 0, a positive integer, or a negative integer depending on the result of the comparison.