Critical Thinking Questions Ch. 9

1. The third element of the array has the value of the index 2 (arrays are zero-indexed, i.e. the first element is at the index 0, the second at 1, and the third at 2).

2.
```
int[] quantities = new int[20];
```

3.
```
double[] heights = {1.65, 2.15, 4.95};
```

4.
```
for (int grade : grades)
{
    System.out.println(grade);
}
```

6. Passing a whole array to a method will pass by reference to the array, and the method will be able to modify the elements of the original array (changes made inside the method will be made to the original array). Passing a single element (if it’s a primitive type) will only pass the value, meaning that changes to the parameter inside the method will not affect the original element in the array.

7. Offset array indexes are needed if the range of values does not start at 0 or to prevent the wastage of space in large ranges. For example, for the values of the form 1900 to 2000 (like years), if direct indices are used, then we need an array of 2001 elements, with a lot of elements unused. Offsets map the range efficiently (i.e. subtract the lowest value from the actual value to get the index).

8.
```
String name = "Elaine";
System.out.println(name.charAt(3));
```

   Output: i

10. A dynamic array (like ArrayList) is better when the number of elements is ot known in advance or when it needs to change at runtime, e.g. collect an unpredictable number of user inputs (e.g. add names to a list until the user stops). Fixed-size array - Requires knowing the size beforehand, and it is not easy to resize the array.

11. The ArrayList indexOf() method determines equality by calling the equals() method on the objects being compared.

12. Wrapper class objects (e.g. Integer) can be compared by means of the compareTo() method, which returns a negative value if the first is less than the second, zero if equal, or positive if greater (e.g. element1.compareTo(element2) > 0). They can also be unwrapped to primitive types for comparison (e.g. elements1.intValue() > element2.intValue()) or tested for equality using equals().

https://gemini.google.com/share/2b723812b1be