1. Using the StringTokenizer class

a. Write a **ConsoleProgram** that counts the number of occurrences of a word within a line of text. The program should first ask a line of text from the user. Then, it should ask for a word to search for within the line. The program should split the line of text into tokens (words) using the StringTokenizer class and count the number of times that a token is equal to the word being searched. The output of the program should be as follows.

```
Enter a line of text: To be or not to be
Enter a word: be
The word “be” occurs 2 times in the text
```

b. Write a **ConsoleProgram** that parses a line of text and counts the number of occurrences of words that end with a special character. The program should first ask a line of text from the user. Then, it should ask for a character. The program should split the line of text into tokens (words) using the StringTokenizer class and count the number of times that a token ends with the character specified by the user. The output of the program should be as follows.

```
Enter a line of text: To be or not to be
Enter a character: o
There are 2 words that end with character ‘o’
```

2. Arrays

a. Creating and printing an array with random elements

Write a **ConsoleProgram** named **RandomArray** that creates an integer array of size 10 and assigns a random value to each array element between 0 and 100. You should implement a **printArray** method that displays the contents of the array to the user. The output of the program should look like the following.

```
[11,60,34,82,13,30,71,27,70,89]
```

```java
/* Initial code template */
import acm.program.*;
public class RandomArray extends ConsoleProgram {
    public void run() {
    /* complete the rest of the code here */
    }
    /* prints out the content of an array */
    public void printArray(int[] array) {
        /* complete the rest of the code here */
    }
}
```

b. Shifting elements of an array

Now extend the program so that it shifts the elements of the array from left to right. The first element should be replaced with the last element and every other element should be replaced with the previous element in the array. Implement a method called **shiftArrayRight**, which takes an array of integers as input and performs the shift operation. In the run
method, call this method to shift the contents of the array to the right. Call the `printArray` method before and after the operation to display the contents of the array. The output of the program should be as follows.

```
[11, 60, 34, 82, 13, 30, 71, 27, 70, 89]
[89, 11, 60, 34, 82, 13, 30, 71, 27, 70]
```

c. **Shifting elements in both directions**

Now extend the program so that it can shift the elements of the array in both directions, and for many times. First, implement another method called `shiftArrayLeft`, which takes an array of integers as input and shifts all the elements of the array from right to left. The last element should be replaced with the first element and every other element should be replaced with the next element in the array. Then, modify the run method so that it asks for the number of shifts from the user. If the user specifies a negative number, this means that the shift operation should be performed from right to left. A positive number indicates a shift operation to be performed from left to right. The run method should call either `shiftArrayRight` or `shiftArrayLeft` methods many times as indicated by the number of shifts entered by the user. Again, the previously written `printArray` method will be called to display the contents of the array before and after the shift operations. The output of the program should be as follows.

```
[11, 60, 34, 82, 13, 30, 71, 27, 70, 89]
Enter the number of shifts: 3
[27, 70, 89, 11, 60, 34, 82, 13, 30, 71]
```

In another run of the program, the output could be as follows.

```
[11, 60, 34, 82, 13, 30, 71, 27, 70, 89]
Enter the number of shifts: -2
[34, 82, 13, 30, 71, 27, 70, 89, 11, 60]
```