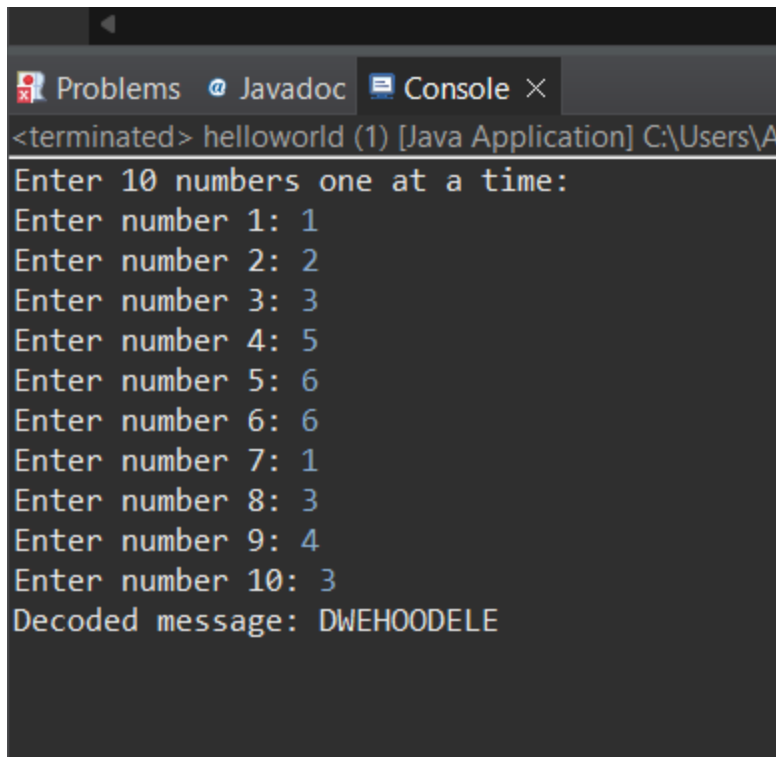


## Java 5\_2

1. Consider you are asked to decode a secret message. The coded message is in numbers and each number stands for a specific letter. You discover enough of the secret code to decode the current message. So far, you know: • 1 represents "D" • 2 represents "W" • 3 represents "E" • 4 represents "L" • 5 represents "H" • 6 represents "O" • 7 represents "R" Write a program that prompts the user for 10 numbers, one at a time, and prints out the decoded message. If the user enters a number that is not one of those already deciphered, prompt him/her for a new number. Test your code with the following input: 5 3 4 4 6 2 6 7 4 1

```
2. package helloworld;
3. import java.util.Scanner;
4. public class helloworld {
5.     public static void main(String[] args) {
6.         Scanner scanner = new Scanner(System.in);
7.         char[] decodedMessage = new char[10];
8.         int[] validNumbers = {1, 2, 3, 4, 5, 6, 7};
9.         char[] correspondingLetters = {'D', 'W', 'E', 'L', 'H', 'O', 'R'};
10.
11.         System.out.println("Enter 10 numbers one at a time:");
12.
13.         for (int i = 0; i < 10; i++) {
14.             int number;
15.             while (true) {
16.                 System.out.print("Enter number " + (i + 1) + ": ");
17.                 number = scanner.nextInt();
18.                 if (number >= 1 && number <= 7) {
19.                     break;
20.                 } else {
21.                     System.out.println("Invalid number! Please enter a number
between 1 and 7.");
22.                 }
23.             }
24.             decodedMessage[i] = correspondingLetters[number - 1];
25.         }
26.
27.         System.out.print("Decoded message: ");
28.         for (char letter : decodedMessage) {
29.             System.out.print(letter);
30.         }
31.         System.out.println();
32.     }
33. }
```



```
<terminated> helloworld (1) [Java Application] C:\Users\A
Enter 10 numbers one at a time:
Enter number 1: 1
Enter number 2: 2
Enter number 3: 3
Enter number 4: 5
Enter number 5: 6
Enter number 6: 6
Enter number 7: 1
Enter number 8: 3
Enter number 9: 4
Enter number 10: 3
Decoded message: DWEHOODELE
```

2. Suppose you are implementing a search routine that searches through a String, character by character, until it finds a space character. As soon as you find the first space character, you decide that you do not want to continue searching the string. If you are using a WHILE loop and your loop will continue to execute until you have gone through the entire string, should you use the keyword break or continue when you find the first space character? Why? Why would you not use the other keyword?

```
package helloworld;

import java.util.Scanner;

public class helloworld {

    public static void main(String[] args) {

        String str = "Hello World";

        int i = 0;

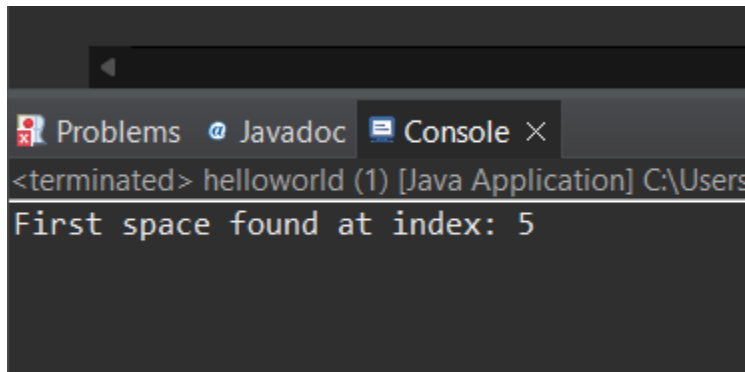
        while (i < str.length()) {
            if (str.charAt(i) == ' ') {
                break; // Exit the loop when a space is found
            }
            i++;
        }
    }
}
```

```
}
```

```
System.out.println("First space found at index: " + i);
```

```
}
```

```
}
```



3. Imagine you are writing a program that prints out the day of the week (Sunday, Monday, Tuesday, etc.) for each day of the year. Before the program executes, can you tell how many times the loop will execute? Assume the year is not a Leap year. Given your answer, which type of loop would you need to implement? Explain your reasoning.

```
package helloworld;
```

```
import java.util.Scanner;
```

```
public class helloworld {
```

```
    public static void main(String[] args) {
```

```
        String[] days = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",  
"Saturday"};
```

```
        int dayIndex = 0;
```

```
        for (int i = 1; i <= 365; i++) {
```

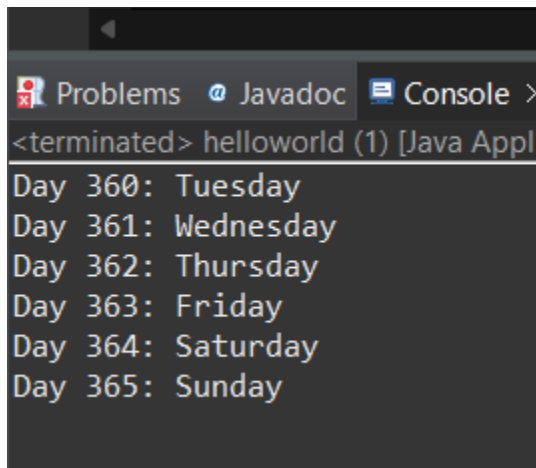
```
            System.out.println("Day " + i + ": " + days[dayIndex]);
```

```
            dayIndex = (dayIndex + 1) % 7; // Cycle through the days of the week
```

```
        }
```

```
    }
```

```
}
```



```
<terminated> helloworld (1) [Java Appl  
Day 360: Tuesday  
Day 361: Wednesday  
Day 362: Thursday  
Day 363: Friday  
Day 364: Saturday  
Day 365: Sunday
```

4. An anagram is a word or a phrase made by transposing the letters of another word or phrase; for example, "parliament" is an anagram of "partial men," and "software" is an anagram of "swear oft." Write a program that figures out whether one string is an anagram of another string. The program should ignore white space and punctuation.

```
package helloworld;
```

```
import java.util.Arrays;
```

```
import java.util.Scanner;
```

```
public class helloworld {
```

```
    public static void main(String[] args) {
```

```
        String str1 = "parliament";
```

```
        String str2 = "partial men";
```

```
        if (areAnagrams(str1, str2)) {
```

```
            System.out.println("The strings are anagrams.");
```

```
        } else {
```

```
            System.out.println("The strings are not anagrams.");
```

```

    }
}

public static boolean areAnagrams(String str1, String str2) {
    // Remove white spaces and punctuation, and convert to lower case
    str1 = str1.replaceAll("[\\s\\p{Punct}]", "").toLowerCase();
    str2 = str2.replaceAll("[\\s\\p{Punct}]", "").toLowerCase();

    // Convert strings to char arrays and sort them
    char[] charArray1 = str1.toCharArray();
    char[] charArray2 = str2.toCharArray();
    Arrays.sort(charArray1);
    Arrays.sort(charArray2);

    // Compare sorted char arrays
    return Arrays.equals(charArray1, charArray2);
}
}

```

