Using String format to create neat columns

% - start of the format	- left justify	s - String output	d - integer output
f - decimal output	b - boolean output	c - single char output	\t – tab
\n - new line			

StringFormatting.java

```
7
8     public class StringFormatting {
9
10        public static void main(String[] args) {
11            String[] stArr1 = {"Joe", "Hendrick", "Di", "Frederick"};
12            String[] stArr2 = {"Smit", "van der Walt", "Ding", "van der Merve"};
13
14            for (int i = 0; i < 4; i++){
15                String format1 = String.format("%-13s", stArr1[i]);
16                String format2 = String.format("%-20s", stArr2[i]);
17                System.out.print(format1);
18                System.out.println(format2);
19            }
20
21            }
22
23    }
24
```

Output - Temp2021 (run)

```
run:
Joe           Smit
Hendrick      van der Walt
Di            Ding
Frederick     van der Merve
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
22
23      // Using String format to define columns of a specific width and data type
24      public String toString() {
25          String col10s = "%-10s"; // Column of 10, left aligned, string
26          String col20s = "%-20s";  // Column of 20, left aligned, string
27          String col8d = "%8d";  // Column of 8, right aligned, number
28          String col8b = "%8b";  // Column of 8, right aligned, boolean
29          String newLine = "\n";  // New line
30
31          String all = String.format(col10s + col20s + col10s + col8d + col8b + newLine,name,country,colour,number,isProtected);
32          return all;
33      }
```

Output - Grade12_2021 (run)

```
run:
Antelope manager
impala     South Africa        brown         5000   false
impala     Botswana            brown         5000   false
impala     Mozambique          brown         7000   false

Print main
BUILD SUCCESSFUL (total time: 0 seconds)
```