String manipulation functions in awk

`awk` comes with many built-in string manipulation functions. Let's have a look at a few of them:

- `length(string)`: This returns the string length.
- `index(string, search_string)`: This returns the position at which `search_string` is found in the string.
- `split(string, array, delimiter)`: This stores the list of strings generated by using the delimiter in the array.
- `substr(string, start-position, end-position)`: This returns the substring created from the string by using the start and end character offsets.
- `sub(regex, replacement_str, string)`: This replaces the first occurring regular expression match from the string with `replacment_str`.
- `gsub(regex, replacment_str, string)`: This is similar to `sub()`, but it replaces every regular expression match.
- `match(regex, string)`: This returns the result of whether a regular expression (regex) match is found in the string or not. It returns a non-zero output if a match is found, otherwise it returns zero. Two special variables are associated with `match()`. They are `RSTART` and `RLENGTH`. The `RSTART` variable contains the position at which the regular expression match starts. The `RLENGTH` variable contains the length of the string matched by the regular expression.

Finding the frequency of words used in a given file

Finding the frequency of words used in a file is an interesting exercise to apply the text-processing skills. It can be done in many different ways. Let's see how to do it.

Getting ready

We can use associative arrays, `awk`, `sed`, `grep`, and so on, to solve this problem in different ways. **Words** are alphabetic characters, delimited by space or a period. First, we should parse all the words in a given file and then the count of each word needs to be found. Words can be parsed by using regex with any of the tools, such as `sed`, `awk`, or `grep`.

How to do it...

We just saw the logic and ideas about the solution; now let's create the shell script as follows:

```
#!/bin/bash
#Name: word_freq.sh
#Desc: Find out frequency of words in a file

if [ $# -ne 1 ];
then
  echo "Usage: $0 filename";
  exit -1
fi

filename=$1

egrep -o "\b[[:alpha:]]+\b" $filename | \

awk '{ count[$0]++ }
END{ printf("%-14s%s\n","Word","Count") ;
for(ind in count)
{  printf("%-14s%d\n",ind,count[ind]);  }

}'
```

A sample output is as follows:

```
$ ./word_freq.sh words.txt
Word          Count
used           1
this             2
counting   1
```

How it works...

`egrep -o "\b[[:alpha:]]+\b" $filename` is used to output only words. The `-o` option will print the matching character sequence, delimited by a newline character. Hence, we receive words in each line.

`\b` is the word boundary character. `[:alpha:]` is a character class for alphabets. The `awk` command is used to avoid the iteration through each word. Since `awk`, by default, executes the statements in the `{ }` block for each row, we don't need a specific loop for doing that. Hence, the count is incremented as `count[$0]++` by using the associative array. Finally, in the `END{}` block, we print the words and their count by iterating through the words.