1. Write a Perl program `word_frequency.pl` which prints a count of all the words found in its input. Your program should ignore case. It should treat any sequence of alphabetic characters([a-z]) as a word. It should treat any non-alphabetic character as a space. It should print words and their counts sorted in increasing order of frequency in format shown in this example:

```
$ ./word_frequency.pl
Peter Piper picked a peck of pickled peppers;
A peck of pickled peppers Peter Piper picked;
If Peter Piper picked a peck of pickled peppers,
Where's the peck of pickled peppers Peter Piper picked?
Ctrl-d
1 the
1 s
1 where
1 if
3 a
4 pickled
4 piper
4 peppers
4 of
4 peter
4 peck
4 picked
```

2. Write a Perl program `missing_words.pl` which given two files as arguments prints, in sorted order, all the words found in file1 but not file2.
   You can assume words occur one per line in each file.

3. Write a Perl program that given the road distances between a number of towns (on standard input) calculates the shortest journey between two towns specified as arguments. Here is an example of how your program should behave.

```
$ ./shortest_path.pl Parkes Gilgandra
Bourke Broken-Hill   217
Bourke Dubbo          23
Bourke Gilgandra      62
Bourke Parkes         71
Canowindra Dubbo      35
Canowindra Gilgandra  13
Canowindra Parkes    112
Dubbo Gilgandra       91
Dubbo Parkes          57
Ctrl-d
Shortest route is length = 105: Parkes Dubbo Canowindra Gilgandra.
```

4. Write a Perl function `printHash()` that displays the contents of a Perl associative array (hash) in the format below (its the format used by the PHP function `print_r`) e.g.he hash table ...

```
colours = ("John"=>"blue", "Anne"=>"red", "Andrew"=>"green");
```

   and the function call ...

```
printHash(%colours);
```

   should produce the output ...

```
[Andrew] => green
[Anne] => red
[John] => blue
```

   Since the function achieves its effect via `print` , it doesn't really need to return any value, but since Perl functions typically return *something*, `printHash` should return a count of the number of items displayed (i.e. the number of keys in the hash table). Note that the hash should be displayed in ascending alphabetical order on key values.

5. A bigram is two words occuring consecutively in a piece of text. Some pairs of words tend to occur more commonly than others as bigrams, e.g `cold beer` or `programming language`.
   Your task is to write a Perl program `bigrams.pl` which reads a piece of text, and prints the words which occur in the text in sorted order, one per line, Each word should be accompanied by the word which most frequently follows it in the text - if several words occur equally often after the word, any of them can be printed. The number of times the word occurs in the text should be indicated as should the number of times the second word follows it. Case should be ignored. For example given this text:

```
$ ./bigrams.pl
Peter Piper picked a peck of pickled peppers;
A peck of pickled peppers Peter Piper picked;
If Peter Piper picked a peck of pickled peppers,
Where's the peck of pickled peppers Peter Piper picked?
Ctrl-d
a(3) peck(3)
if(1) peter(1)
of(4) pickled(4)
peck(4) of(4)
peppers(4) peter(2)
peter(4) piper(4)
picked(3) a(2)
pickled(4) peppers(4)
piper(4) picked(4)
s(1) the(1)
the(1) peck(1)
where(1) s(1)
```