Scrabble Score

Given a word, compute the Scrabble score for that word.

Letter Values

You'll need these:

Letter	Value
A, E, I, O, U, L, N, R, S, T	1
D, G	2
B, C, M, P	3
F, H, V, W, Y	4
K	5
J, X	8
Q, Z	10

Examples

"cabbage" should be scored as worth 14 points:

- 3 points for C
- 1 point for A, twice
- 3 points for B, twice
- 2 points for G
- 1 point for E

And to total:

- `3 + 2*1 + 2*3 + 2 + 1`
- = `3 + 2 + 6 + 3`
- = `5 + 9`
- = 14

Hamming Distance

Calculate the Hamming Distance between two DNA strands.

Your body is made up of cells that contain DNA. Those cells regularly wear out and need replacing, which they achieve by dividing into daughter cells. In fact, the average human body experiences about 10 quadrillion cell divisions in a lifetime!

When cells divide, their DNA replicates too. Sometimes during this process mistakes happen and single pieces of DNA get encoded with the incorrect information. If we compare two strands of DNA and count the differences between them we can see how many mistakes occurred. This is known as the "Hamming Distance".

We read DNA using the letters C,A,G and T. Two strands might look like this:

```
GAGCCTACTAACGGGAT
CATCGTAATGACGGCCT
^ ^ ^  ^ ^    ^^
```

They have 7 differences, and therefore the Hamming Distance is 7.

The Hamming Distance is useful for lots of things in science, not just biology, so it's a nice phrase to be familiar with :)

The Hamming distance is only defined for sequences of equal length, so you should test with equal sized strings only.

D&D Character

For a game of Dungeons & Dragons, each player starts by generating a character they can play with. This character has, among other things, six abilities; strength, dexterity, constitution, intelligence, wisdom and charisma. These six abilities have scores that are determined randomly. You do this by rolling four 6-sided dice and record the sum of the largest three dice. You do this six times, once for each ability.

Your character's initial hitpoints are 10 + your character's constitution modifier. You find your character's constitution modifier by subtracting 10 from your character's constitution, divide by 2 and round down.

Write a random character generator that follows the rules above.

For example, the six throws of four dice may look like:

- 5, 3, 1, 6: You discard the 1 and sum 5 + 3 + 6 = 14, which you assign to strength.
- 3, 2, 5, 3: You discard the 2 and sum 3 + 5 + 3 = 11, which you assign to dexterity.
- 1, 1, 1, 1: You discard the 1 and sum 1 + 1 + 1 = 3, which you assign to constitution.
- 2, 1, 6, 6: You discard the 1 and sum 2 + 6 + 6 = 14, which you assign to intelligence.
- 3, 5, 3, 4: You discard the 3 and sum 5 + 3 + 4 = 12, which you assign to wisdom.
- 6, 6, 6, 6: You discard the 6 and sum 6 + 6 + 6 = 18, which you assign to charisma.

Because constitution is 3, the constitution modifier is -4 and the hitpoints are 6.

Difference of Squares

Find the difference between the square of the sum and the sum of the squares of the first N natural numbers.

The square of the sum of the first ten natural numbers is $(1 + 2 + ... + 10)^2 = 55^2 = 3025$.

The sum of the squares of the first ten natural numbers is $1^2 + 2^2 + ... + 10^2 = 385$.

Hence the difference between the square of the sum of the first ten natural numbers and the sum of the squares of the first ten natural numbers is 3025 - 385 = 2640.

You are not expected to discover an efficient solution to this yourself from first principles; research is allowed, indeed, encouraged. Finding the best algorithm for the problem is a key skill in software engineering.

Word Count

Given a phrase, count the occurrences of each *word* in that phrase.

For the purposes of this exercise you can expect that a *word* will always be one of:

1. A *number* composed of one or more ASCII digits (ie "0" or "1234") OR
2. A *simple word* composed of one or more ASCII letters (ie "a" or "they") OR
3. A *contraction* of two *simple words* joined by a single apostrophe (ie "it's" or "they're")

When counting words you can assume the following rules:

1. The count is *case insensitive* (ie "You", "you", and "YOU" are 3 uses of the same word)
2. The count is *unordered*; the tests will ignore how words and counts are ordered
3. Other than the apostrophe in a *contraction* all forms of *punctuation* are ignored
4. The words can be separated by *any* form of whitespace (ie "\t", "\n", " ")

For example, for the phrase `"That's the password: 'PASSWORD 123'!", cried the Special Agent.\nSo I fled.` the count would be:

```
that's: 1
the: 2
password: 2
123: 1
cried: 1
special: 1
agent: 1
so: 1
i: 1
fled: 1
```

Rotational Cipher

Create an implementation of the rotational cipher, also sometimes called the Caesar cipher.

The Caesar cipher is a simple shift cipher that relies on transposing all the letters in the alphabet using an integer key between `0` and `26`. Using a key of `0` or `26` will always yield the same output due to modular arithmetic. The letter is shifted for as many values as the value of the key.

The general notation for rotational ciphers is `ROT + <key>`. The most commonly used rotational cipher is `ROT13`.

A `ROT13` on the Latin alphabet would be as follows:

```
Plain:  abcdefghijklmnopqrstuvwxyz
Cipher: nopqrstuvwxyzabcdefghijklm
```

It is stronger than the Atbash cipher because it has 27 possible keys, and 25 usable keys.

Ciphertext is written out in the same formatting as the input including spaces and punctuation.

Examples

- ROT5 `omg` gives `trl`
- ROT0 `c` gives `c`
- ROT26 `Cool` gives `Cool`
- ROT13 `The quick brown fox jumps over the lazy dog.` gives `Gur dhvpx oebja sbk whzcf bire gur ynml qbt.`
- ROT13 `Gur dhvpx oebja sbk whzcf bire gur ynml qbt.` gives `The quick brown fox jumps over the lazy dog.`

Linked List

Implement a doubly linked list.

Like an array, a linked list is a simple linear data structure. Several common data types can be implemented using linked lists, like queues, stacks, and associative arrays.

A linked list is a collection of data elements called *nodes*. In a *singly linked list* each node holds a value and a link to the next node. In a *doubly linked list* each node also holds a link to the previous node.

You will write an implementation of a doubly linked list. Implement a Node to hold an Integer value and pointers to the next and previous nodes. Then implement a List which holds references to the first and last node and offers an array-like interface for adding and removing items:

- `push` (*insert value at back*);
- `pop` (*remove value at back*);
- `shift` (*remove value at front*).
- `unshift` (*insert value at front*);

To keep your implementation simple, the tests will not cover error conditions. Specifically: `pop` or `shift` will never be called on an empty list.

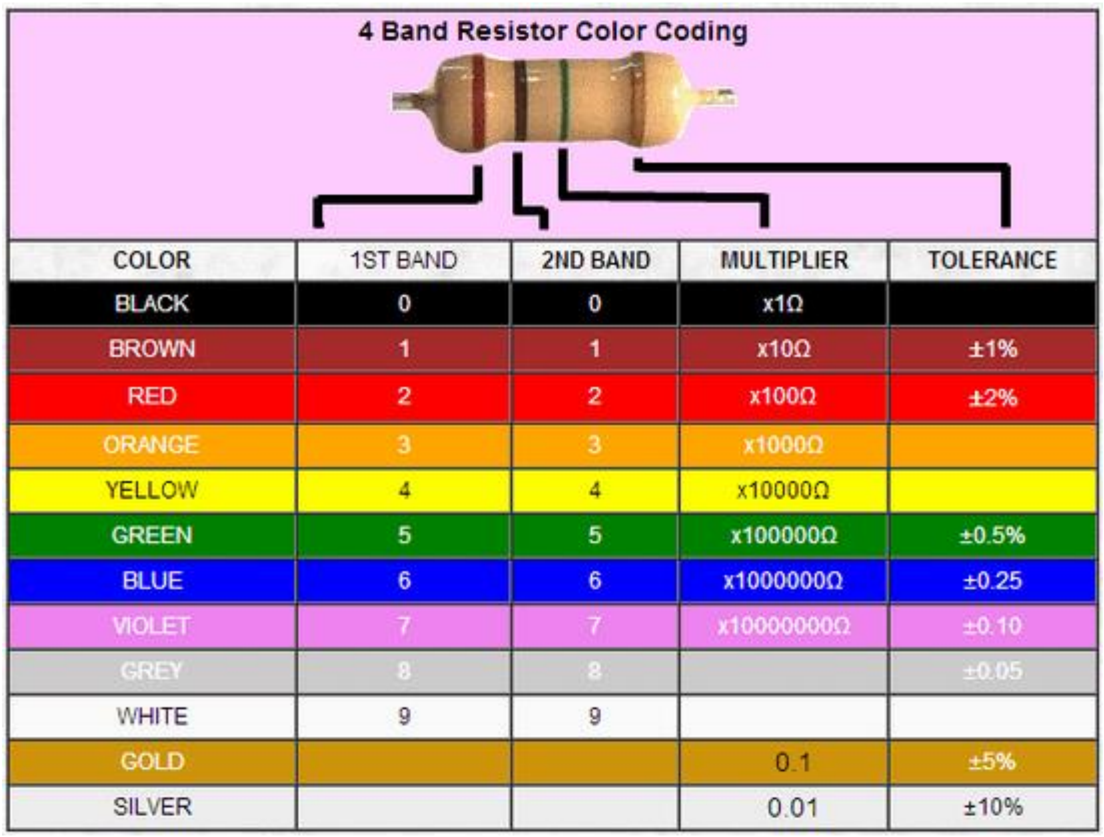
Resistor Colors

Resistors are small - so small in fact that if you printed the resistance value on them, it would be hard to read.

To get around this problem, manufacturers print color-coded bands onto the resistors to denote their resistance values. Each band has a position and a numeric value.

In this exercise you are going to create a helpful program so that you don't have to remember the values of the bands.

These colors are encoded as follows:



4 Band Resistor Color Coding

COLOR	1ST BAND	2ND BAND	MULTIPLIER	TOLERANCE
BLACK	0	0	x1Ω	
BROWN	1	1	x10Ω	±1%
RED	2	2	x100Ω	±2%
ORANGE	3	3	x1000Ω	
YELLOW	4	4	x10000Ω	
GREEN	5	5	x100000Ω	±0.5%
BLUE	6	6	x1000000Ω	±0.25
VIOLET	7	7	x10000000Ω	±0.10
GREY	8	8		±0.05
WHITE	9	9		
GOLD			0.1	±5%
SILVER			0.01	±10%

The goal of this exercise is to write a program that given a string composed by the name of four colors, it gives you the resulting resistance. Example:

“Brown Black Yellow Gold” = > 1000000 Ohms => 100 KOhms

You can ignore the last color for the calculation, just validate that the color in the fourth position has a valid tolerance value. Example:

“Brown Black Yellow Black” = > Error

Diamond

The diamond kata takes as its input a letter, and outputs it in a diamond shape. Given a letter, it prints a diamond starting with 'A', with the supplied letter at the widest point.

Requirements

- The first row contains one 'A'.
- The last row contains one 'A'.
- All rows, except the first and last, have exactly two identical letters.
- All rows have as many trailing spaces as leading spaces. (This might be 0).
- The diamond is horizontally symmetric.
- The diamond is vertically symmetric.
- The diamond has a square shape (width equals height).
- The letters form a diamond shape.
- The top half has the letters in ascending order.
- The bottom half has the letters in descending order.
- The four corners (containing the spaces) are triangles.

Examples

In the following examples, spaces are indicated by `·` characters.

Diamond for letter 'A':

```
A
```

Diamond for letter 'C':

```
··A··
·B·B·
C···C
·B·B·
··A··
```

Diamond for letter 'E':

```
····A····
···B·B···
··C···C··
·D·····D·
E·······E
·D·····D·
··C···C··
···B·B···
····A····
```

Acronyms

Convert a phrase to its acronym.

Techies love their TLA (Three Letter Acronyms)!

Help generate some jargon by writing a program that converts a long name like Portable Network Graphics to its acronym (PNG).

Roman Numerals

Write a function to convert from normal numbers to Roman Numerals.

The Romans were a clever bunch. They conquered most of Europe and ruled it for hundreds of years. They invented concrete and straight roads and even bikinis. One thing they never discovered though was the number zero. This made writing and dating extensive histories of their exploits slightly more challenging, but the system of numbers they came up with is still in use today. For example the BBC uses Roman numerals to date their programmes.

The Romans wrote numbers using letters - I, V, X, L, C, D, M. (notice these letters have lots of straight lines and are hence easy to hack into stone tablets).

```
 1  => I
10  => X
 7  => VII
```

There is no need to be able to convert numbers larger than about 3000. (The Romans themselves didn't tend to go any higher)

Wikipedia says: Modern Roman numerals ... are written by expressing each digit separately starting with the left most digit and skipping any digit with a value of zero.

To see this in practice, consider the example of 1990.

In Roman numerals 1990 is MCMXC:

1000=M 900=CM 90=XC

2008 is written as MMVIII:

2000=MM 8=VIII

Minesweeper

Add the mine counts to a completed Minesweeper board.

Minesweeper is a popular game where the user has to find the mines using numeric hints that indicate how many mines are directly adjacent (horizontally, vertically, diagonally) to a square.

In this exercise you have to create some code that counts the number of mines adjacent to a given empty square and replaces that square with the count.

The board is a rectangle composed of blank space (' ') characters. A mine is represented by an asterisk ('*') character.

If a given space has no adjacent mines at all, leave that square blank.

Examples

For example, you may receive a 5 x 4 board like this (empty spaces are represented here with the '·' character for display on screen):

```
·*·*·
··*··
··*··
·····
```

And your code will transform it into this:

```
1*3*1
13*31
·2*2·
·111·
```