This is CS50

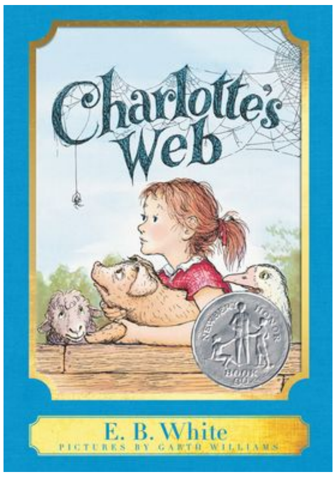
CS50's Introduction to Computer Science

OpenCourseWare

Donate (https://cs50.harvard.edu/donate)

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan) (https://www.instagram.com/davidjmalan/) (https://www.linkedin.com/in/malan/) (https://www.reddit.com/user/davidjmalan) (https://www.threads.net/@davidjmalan) (https://twitter.com/davidjmalan)

Readability



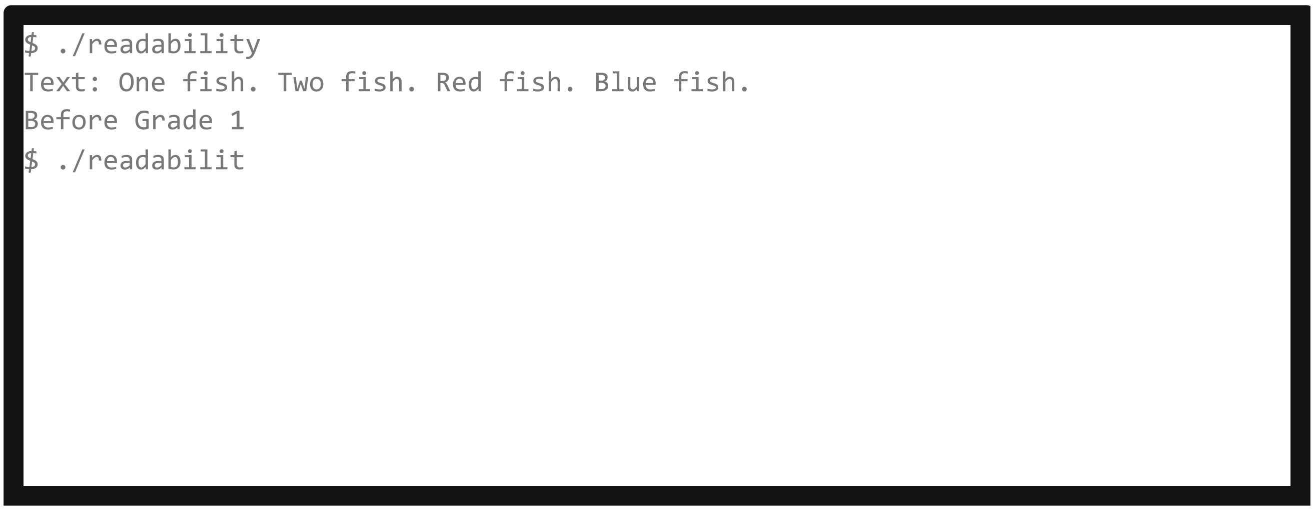
Problem to Solve

According to Scholastic (https://www.scholastic.com/teachers/teaching-tools/collections/guided-reading-book-lists-for-every-level.html), E.B. White's *Charlotte's Web* is between a second- and fourth-grade reading level, and Lois Lowry's *The Giver* is between an eighth- and twelfth-grade reading level. What does it mean, though, for a book to be at a particular reading level?

Well, in many cases, a human expert might read a book and make a decision on the grade (i.e., year in school) for which they think the book is most appropriate. But an algorithm could likely figure that out too!

In a file called `readability.c` in a folder called `readability`, you'll implement a program that calculates the approximate grade level needed to comprehend some text. Your program should print as output "Grade X" where "X" is the grade level computed, rounded to the nearest integer. If the grade level is 16 or higher (equivalent to or greater than a senior undergraduate reading level), your program should output "Grade 16+" instead of giving the exact index number. If the grade level is less than 1, your program should output "Before Grade 1".

Demo



```
$ ./readability
Text: One fish. Two fish. Red fish. Blue fish.
Before Grade 1
$ ./readabilit
```

Recorded with **asciinema**

Background

So what sorts of traits are characteristic of higher reading levels? Well, longer words probably correlate with higher reading levels. Likewise, longer sentences probably correlate with higher reading levels, too.

A number of "readability tests" have been developed over the years that define formulas for computing the reading level of a text. One such readability test is the *Coleman-Liau index*. The

Coleman-Liau index of a text is designed to output that (U.S.) grade level that is needed to understand some text. The formula is

```
index = 0.0588 * L - 0.296 * S - 15.8
```

where `L` is the average number of letters per 100 words in the text, and `S` is the average number of sentences per 100 words in the text.

Advice

Click the below toggles to read some advice!

▼ Write some code that you know will compile

```
#include <ctype.h>
#include <cs50.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

int main(void)
{

}
```

Notice that you've now included a few header files that will give you access to functions which might help you solve this problem.

▼ Write some pseudocode before writing more code

If unsure how to solve the problem itself, break it down into smaller problems that you can probably solve first. For instance, this problem is really only a handful of problems:

1. Prompt the user for some text
2. Count the number of letters, words, and sentences in the text
3. Compute the Coleman-Liau index
4. Print the grade level

Let's write some pseudcode as comments to remind you to do just that:

```
#include <ctype.h>
#include <cs50.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

int main(void)
{
    // Prompt the user for some text
    // Count the number of letters, words, and sentences in the text

    // Compute the Coleman-Liau index

    // Print the grade level
}
```

▼ Convert the pseudocode to code

First, consider how you might prompt the user for some text. Recall that `get_string`, a function in the CS50 library, can prompt the user for a string.

```
#include <ctype.h>
#include <cs50.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

int main(void)
{
    // Prompt the user for some text
    string text = get_string("Text: ");

    // Count the number of letters, words, and sentences in the text

    // Compute the Coleman-Liau index

    // Print the grade level
}
```

Now that you’ve collected input from the user, you can begin to analyze that input. First, try to count the number of letters in the text. Consider letters to be uppercase or lowercase alphabetical character, not punctuation, digits, or other symbols.

One way to approach this problem is to create a function called `count_letters` that takes a string, `text`, as input, and then returns the number of letters in that text as an `int`.

```
int count_letters(string text)
{
    // Return the number of letters in text
}
```

You’ll need to write your own code to count the number of letters in the text. But someone more experienced than you may have already written a function to determine if a character is alphabetical. This is a good opportunity to use the CS50 manual (https://manual.cs50.io/), a collection of explanations of common functions in the C Standard Library.

You can integrate `count_letters` into the code you’ve already written, as follows.

```
#include <ctype.h>
#include <cs50.h>
```

```
#include <math.h>
#include <stdio.h>
#include <string.h>

int count_letters(string text);

int main(void)
{
    // Prompt the user for some text
    string text = get_string("Text: ");

    // Count the number of letters, words, and sentences in the text
    int letters = count_letters(text);

    // Compute the Coleman-Liau index

    // Print the grade level
}

int count_letters(string text)
{
    // Return the number of letters in text
}
```

Next, write a function to count words.

```
int count_words(string text)
{
    // Return the number of words in text
}
```

For the purpose of this problem, we'll consider any sequence of characters separated by a space to be a word (so a hyphenated word like "sister-in-law" should be considered one word, not three). You may assume that a sentence:

- will contain at least one word;
- will not start or end with a space; and
- will not have multiple spaces in a row.

Under those assumptions, you might be able to find a relationship between the number words and the number of spaces. You are, of course, welcome to attempt a solution that will tolerate multiple spaces between words or indeed, no words!

You can now integrate `count_words` into your program as follows:

```
#include <ctype.h>
#include <cs50.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

int count_letters(string text);
int count_words(string text);
int main(void)
{
    // Prompt the user for some text
    string text = get_string("Text: ");

    // Count the number of letters, words, and sentences in the text
    int letters = count_letters(text);
    int words = count_words(text);

    // Compute the Coleman-Liau index

    // Print the grade level
}

int count_letters(string text)
{
    // Return the number of letters in text
}

int count_words(string text)
{
    // Return the number of words in text
}
```

Finally, write a function to count sentences. You can consider any sequence of characters that ends with a `.` or a `!` or a `?` to be a sentence.

```
int count_sentences(string text)
{
    // Return the number of sentences in text
}
```

You can integrate `count_sentences` into your program as follows:

```
#include <ctype.h>
#include <cs50.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

int count_letters(string text);
int count_words(string text);
int count_sentences(string text);

int main(void)
{
    // Prompt the user for some text
    string text = get_string("Text: ");

    // Count the number of letters, words, and sentences in the text
    int letters = count_letters(text);
    int words = count_words(text);
    int sentences = count_sentences(text);

    // Compute the Coleman-Liau index
    // Print the grade level
}

int count_letters(string text)
{
    // Return the number of letters in text
}

int count_words(string text)
{
    // Return the number of words in text
}

int count_sentences(string text)
{
    // Return the number of sentences in text
}
```

Finally, compute the Coleman-Liau index and print the resulting grade level.

- Recall that the Coleman-Liau index is computed using `index = 0.0588 * L - 0.296 * S - 15.8`
- `L` is the average number of letters per 100 words in the text: that is, the number of letters divided by the number of words, all multiplied by 100.
- `S` is the average number of sentences per 100 words in the text: that is, the number of sentences divided by the number of words, all multiplied by 100.
- You'll want to round the result to the nearest whole number, so recall that `round` is declared in `math.h`, per manual.cs50.io (https://manual.cs50.io/).
- Recall that, when dividing values of type `int` in C, the result will also be an `int`, with any remainder (i.e., digits after the decimal point) discarded. Put another way, the result will be "truncated." You might want to cast your one or more values to `float` before performing division when calculating `L` and `S`!

If the resulting index number is 16 or higher (equivalent to or greater than a senior undergraduate reading level), your program should output "Grade 16+" instead of outputting an exact index number. If the index number is less than 1, your program should output "Before Grade 1".

Walkthrough

How to Test

Try running your program on the following texts, to ensure you see the specified grade level. Be sure to copy only the text, no extra spaces.

- `One fish. Two fish. Red fish. Blue fish.` (Before Grade 1)
- `Would you like them here or there? I would not like them here or there. I would not like them anywhere.` (Grade 2)
- `Congratulations! Today is your day. You're off to Great Places! You're off and away!` (Grade 3)
- `Harry Potter was a highly unusual boy in many ways. For one thing, he hated the summer holidays more than any other time of year. For another, he really wanted to do his homework, but was forced to do it in secret, in the dead of the night. And he also happened to be a wizard.` (Grade 5)
- `In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.` (Grade 7)
- `Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, "and what is the use of a book," thought Alice "without pictures or conversation?"` (Grade 8)
- `When he was nearly thirteen, my brother Jem got his arm badly broken at the elbow. When it healed, and Jem's fears of never being able to play football were assuaged, he was seldom self-conscious about his injury. His left arm was somewhat shorter than his right; when he stood or walked, the back of his hand was at right angles to his body, his thumb parallel to his thigh.` (Grade 8)

- `There are more things in Heaven and Earth, Horatio, than are dreamt of in your philosophy.` (Grade 9)
- `It was a bright cold day in April, and the clocks were striking thirteen. Winston Smith, his chin nuzzled into his breast in an effort to escape the vile wind, slipped quickly through the glass doors of Victory Mansions, though not quickly enough to prevent a swirl of gritty dust from entering along with him.` (Grade 10)
- `A large class of computational problems involve the determination of properties of graphs, digraphs, integers, arrays of integers, finite families of finite sets, boolean formulas and elements of other countable domains.` (Grade 16+)

Correctness

In your terminal, execute the below to check your work's correctness.

```
check50 cs50/problems/2024/x/readability
```

Style

Execute the below to evaluate the style of your code using `style50`.

```
style50 readability.c
```

How to Submit

In your terminal, execute the below to submit your work.

```
submit50 cs50/problems/2024/x/readability
```