The operands in the expression are floating-point numbers; the operators are +, -, *, and /. The expression is evaluated from left to right (no operator takes precedence over any other operator).

13. Write a program that calculates the average word length for a sentence:

```
Enter a sentence: It was deja vu all over again.
Average word length: 3.4
```

For simplicity, your program should consider a punctuation mark to be part of the word to which it is attached. Display the average word length to one decimal place.

14. Write a program that uses Newton's method to compute the square root of a positive floating-point number:

```
Enter a positive number: 3
Square root: 1.73205
```

Let x be the number entered by the user. Newton's method requires an initial guess y for the square root of x (we'll use $y = 1$). Successive guesses are found by computing the average of y and x/y. The following table shows how the square root of 3 would be found:

x	y	x/y	Average of y and x/y
3	1	3	2
3	2	1.5	1.75
3	1.75	1.71429	1.73214
3	1.73214	1.73196	1.73205
3	1.73205	1.73205	1.73205

Note that the values of y get progressively closer to the true square root of x. For greater accuracy, your program should use variables of type `double` rather than `float`. Have the program terminate when the absolute value of the difference between the old value of y and the new value of y is less than the product of .00001 and y. *Hint:* Call the `fabs` function to find the absolute value of a `double`. (You'll need to include the `<math.h>` header at the beginning of your program in order to use `fabs`.)

15. Write a program that computes the factorial of a positive integer:

```
Enter a positive integer: 6
Factorial of 6: 720
```

(a) Use a `short` variable to store the value of the factorial. What is the largest value of n for which the program correctly prints the factorial of n?
(b) Repeat part (a), using an `int` variable instead.
(c) Repeat part (a), using a `long` variable instead.
(d) Repeat part (a), using a `long long` variable instead (if your compiler supports the `long long` type).
(e) Repeat part (a), using a `float` variable instead.
(f) Repeat part (a), using a `double` variable instead.
(g) Repeat part (a), using a `long double` variable instead.

In cases (e)–(g), the program will display a close approximation of the factorial, not necessarily the exact value.