**Evaluate**

Write a function in C or C++ that evaluates the result of a simple expression.

The function should ignore whitespace between tokens, but stop at the first non-valid character.

Valid tokens are listed in the table below:

|  |  |
| --- | --- |
| <number> | Only signed decimal integers are allowed in expressions  A <number> is a sequence of characters in the range ‘0’..’9’ |
| ( ) | Nested expressions should be evaluated first. |
| +, -, \*, / | Basic operators are addition, subtraction, multiplication and division. |

* The expression MUST be parsed from left to right, evaluating operators in that order (e.g. 1 + 3 \* 4 = 16). If there is an error in the expression, the function should return false.
* The expression should be able to handle negative numbers (e.g. -1 + 3 = 2).
* The expression should be able to handle common error cases.

The prototype for the function should be:

bool evaluate(const char \*expression, int &result)  
{  
 ...  
}

Example test cases (not limited to):

|  |  |  |
| --- | --- | --- |
| Input | Result | Return code |
| 1 + 3 | 4 | true |
| (1 + 3) \* 2 | 8 | true |
| (4 / 2) + 6 | 8 | true |
| 4 + (12 / (1 \* 2)) | 10 | true |
| (1 + (12 \* 2) | N/A | false (missing bracket) |

Comment the code so that the interviewer may understand decisions you have taken when writing the code.

We anticipate this exercise should take you about 60 – 90 minutes.

When returning your solution please confirm that it is all your own work and the length of time taken to complete it. You are likely to be asked to revisit this exercise again at any future interview stage.

Good luck!