02393 C++ Programming Exercises

Week 2

To be handed in via CodeJudge, before September 12, 5pm
`https://dtu.codejudge.net/02393-e16/`

In the following exercises, you will have to compute a function with an input parameter that you should read from `cin`. The result is to be provided on `cout`.

Another sum Write a program that computes the sum of all even integers between 0 and n. For instance, for $n = 6$, the result is $0 + 2 + 4 + 6 = 12$.

Prime Factorization Write a program that computes the prime factorization of a given positive integer. For instance, the factorization of 60 is `2 * 2 * 3 * 5`.

Hints:

- In C++, the modulus function % gives the remainder of integer division, i.e., x is divisible by y if and only if $x\%y == 0$.
- Given the number n to factorize, iterate through all the numbers $i = 2, 3, 4, 5, \ldots$ and check whether i divides n. If so, print out "i $*$ " and continue to check the factorization of n/i. Stop when n cannot be further factorised.

In order to check with CodeJudge, please ensure that (i) the factors are printed in ascending order, (ii) between two factors print a space, an asterisk ($*$), and another space, (iii) and at the end there is a newline (see example above).

Approximating π Compute an approximation of π using Leibniz' formula:

$$\frac{\pi}{4} = \sum_{i=0}^{\infty} \frac{(-1)^i}{2i+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \ldots$$

To that end write a function with head `double pi(int n)` that computes the first n terms of the infinite summation (and then multiplies by 4). For instance for $n = 1$ we get the bad approximation 4, and with increasing n, the approximation gets better.

Hint: the expression $(-1)^i$ could be computed using the function `pow`, but this is rather inefficient (as this causes $log_2 i$ multiplications for every summand). Can you find a better way, avoiding `pow`?