MATHEMATICAL ALGORITHMS I

1. Homework 3:

(1) Write code to work with an expression tree for functions. It should include the following features:
   - a print function, possibly with many brackets. E.g. output like
   $$(((x_0) + (x_0)) * (x_0))$$
   is okay. It is of course desirable to reduce the number of brackets, but this is not the point.
   - an evaluation function to compute the value of the function
   - a differentation function that returns an expression tree for the derivative

   You need to include the following operations:
   - constant nodes and variable nodes
   - power functions, i.e. x^n, where n is real number. You may regard this as a unary node (meaning n is part of the info-field of the node, rather than a child).
   - addition, subtraction, multiplication and division

(2) write a function for reverse mode automatic differentiation (also known as backward). In particular, include a function that computes the value of the gradient; this is different from symbolic differentation.

(3) perform operator overloading (or use function calls if you work in a language that does not support this) to implement forward automatic differentation.

(4) Consider the function
$$H = x^2 + y^3 - y^2$$
Compute the gradient of H at $(0.0, 1.0)$ using automatic differentiation (backward, also known as reverse mode), and compute the symbolic derivative with your program (so not by hand); Output these values to the screen or to a file.

(5) Implement the simple Taylor method.

(6) Apply the simple Taylor method for the following initial value problem:
$$\begin{pmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \nabla H(x, y)$$
$$\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$
to numerically approximate $x(10)$.

   Guidelines:
   - Take as step size 1/4. You may reduce this, but this is not necessary if you do it correctly, and if fact you can take a larger stepsize.
   - Use Lagrange's formula for the remainder to compute a guess for the error by putting the unknown $\tau \in [0, h]$ to 0; you don't need to obtain a true estimate with interval arithmetic.
   - Take the order of the Taylor method to be sufficiently high such that the guessed error is less than 10^{-20}.
   - output the value of $(x(t), y(t))$ and $H(x(t), y(t)\,)$ to the screen after each integration step.
   - You need to use a `long double` for this.

Remark 1.1. When done correctly, the result does not depend (strongly) on the implementation details.

Remark 1.2. In case you missed the lectures, references for expression trees are Knuth's book, volume I. The paper "A Hitchhiker's Guide to Automatic Differentiation" by Hoffmann contains a survey on automatic differentation

Remark 1.3. In practice, tools can be used for autodiff: the webpage

`http://www.autodiff.org`

contains a lot of information. However, the point of this problem is to learn how the method works, not to see whether you can get some library to work (which is of course also important).

Remark 1.4. The program should output part (4) and (6) to the screen or to a file.

If you want to use a language other than `C`, `C++` or `Java`, then please ask me. Most languages will be okay. Provide compilation instructions when anything non-standard is used (for me standard means gcc, clang) as well as the version of `C++` (e.g. 11, 14).

The due date is June 22nd. If you need more time, then write me in advance (this means well before June 22nd).