Institut für Informatik — TUM
Scientific **C**omputing in **C**omputer **S**cience
R. Wittmann, G. Buse

WT 2013

Tutorial
(Advanced Programming)
Worksheet 4:

Assignment 1: Linear Interpolation

Consider an example where you have an unknown function $f(x)$, and you are told that $f(x_a) = y_a$ and $f(x_b) = y_b$. Now you are asked to provide the result of the function evaluation at x_r where $x_a \leq x_r \leq x_b$. Write a program which uses equation (1) to program a function which gets the known values and computes the function value at the point x_r.

$$y_r = y_a + (y_b - y_a)\frac{(x_r - x_a)}{(x_b - x_a)} \tag{1}$$

Assignment 2: Difference Quotient of Arbitrary Functions

Consider an example where you have a known function $f(x)$. You are asked to provide an approximation to the derivative of this function at point x_0. Write a function differenceQuotient with the signature below, which uses equation (2) to calculate and return the difference quotient of function f.

$$f'(x_0) := \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h} \tag{2}$$

```
double
differenceQuotient(double (*f)(double), double x_0, double h = 1e-8);
```

Hint: In your tests, try to pass "real" as well as anonymous functions to differenceQuotient .

Assignment 3: Greatest Common Divisor

In this task we consider the greatest common divisor algorithm which is defined as follows:

$$\begin{aligned} gcd(a, 0) &= a \\ gcd(a, b) &= gcd(b, a \texttt{ mod } b) \end{aligned}$$

In the following you will implement this rather simple algorithm in three different ways:

1. Recursively. You are allowed to use temporary variables, but no static or global ones.

2. Iteratively. Same constraints as in the recursive approach.
   Hint: Apply your knowledge about "tail-recursion".

3. Iteratively, but this time without temporary variables.
   Hint: You can swap two variables using simple arithmetic (or logical) operations.

However, the function signature should remain the same, with the following properties: The function shall take two integer arguments as input and should return one integer. The latter should contain the greatest common divisor of the provided input arguments.

```
int gcd(int a, int b);
```

Home Assignment 1: Fibonacci

Now, we will cover a very famous sequence, namely the Fibonacci sequence, which is defined by the following recurrence relation:

$$\begin{aligned} F(n) &= F(n-1) + F(n-2) \\ F(0) &= 0 \\ F(1) &= 1 \end{aligned}$$

Again, we want to implement different versions.

1. Implement a recursive version directly following the definition. Think about the complexity.
2. Implement a second recursive version requiring only one recursive call.
   Hint: In addition to the number n, use two formal parameters to cache already computed results. Think about different types of formal parameters you can use.

Questions:

Answer the following questions:

- In Assignment 1, What would happen if $x_a = x_b$? How could you avoid that?
- The second algorithm for the recursive calculation of the Fibonacci numbers requires only one recursive call. Can you reformulate the algorithm such that it uses tail-recursion?