Laboratory Work No. 3

Functions in C++

This laboratory work covers the following concepts in C++ programming language:

- function overloading
- default arguments
- reference parameters
- inline functions

⇨ Create a Win32 Console application and an empty C++ source file in Visual Studio IDE to be able to start typing programs.

Task-1: Write a C++ program that gives the definition of two overloaded functions for calculating the distance between two discrete points in the Cartesian coordinate system. First variant takes into consideration of integers only. Second variant extends it double values. Pay attention to the fact that both functions shall be named same but declared with different signatures. The distance formula: $d = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}$.

Remark: You may also use a template function definition to minimize both functions into one instead of having two.

Task-2: Write a C++ program that uses *default arguments* in a function for calculating the volume of a cuboid. All dimensions, namely, the width, the height and the depth shall be defaulted to 1 unit, respectively. Test your program with different function calls!

Remark: Provide both function prototype and definition separately to observe the usage of default values.
Can you repeat default values in the definition?

Task-3: Write a C++ program that uses *reference parameters* in a function for exchanging the values between two characters. Test your program with different set of arguments in the function calls. Provide another version of the function for doing the swapping but this time using pointers to achieve C-style call-by-reference methodology. Observe the difference between the strategies!

Remark: Can reference parameters be defaulted as well?

Task-4: Write a C++ program that gives definitions of *inline functions* for calculating the area of a circle and the volume of a cylinder. The area of a circle can be calculated using πr^2 whereas the volume of a cylinder with $\pi r^2 h$. Pay attention to the fact that the second function's definition might get the benefit of the first one. Test your program with different set of arguments in the function calls.

Remark: You may write a macro to achieve the same effect. Try it!