Informatik für Mathematiker und Physiker HS14

Exercise Sheet 10

Submission deadline: 15:15 - Tuesday 25th November, 2014
Course URL: http://lec.inf.ethz.ch/ifmp/2014/

Assignment 1 (4 points)

Implement a function

```
typedef  std::vector<int>::const_iterator  cIt;

// PRE: [begin, end) is a valid range whose elements are sorted in
//      non-decreasing order.
// POST: returns a const-iterator to an occurrence of val in [begin, end)
//       or returns end if there is no occurrence
cIt binary_search (cIt begin, cIt end, const int val)
```

This function shall recursively scout for an occurrence of val implementing *binary search*. On the lecture website you can find a template binary_search_template.cpp which already contains a couple of test values for your implementation.

Assignment 2 – Skript-Aufgabe 136

The *nesting depth* of an arithmetic expression counts how many pairs of parentheses enclose the innermost number in the expression. For example, the expression 5 has nesting depth 0, (3+4)*(5/6) has nesting depth 1, ((3+4)*(5/6)) has nesting depth 2, and 2*(3*(4*(5*6))) has nesting depth 3 (attained by the numbers 5 and 6).

Formally define the nesting depth of an arithmetic expression given by the BNF of the calculator from the lecture, and write a program nesting_depth.cpp that computes the nesting depth of an expression!

Assignment 3 – Skript-Aufgabe 126 (4 points)

In how many ways can you own CHF 1? Despite its somewhat philosophical appearance, the question is a mathematical one. Given some amount of money, in how many ways can you partition it using the available denominations (bank notes and coins)? Today's denominations in CHF are 1000, 200, 100, 50, 20, 10 (banknotes), 5, 2, 1, 0.50, 0.20, 0.10, 0.05 (coins). The amount of CHF 0.20, for example, can be owned in four ways (to get integers, let's switch to centimes): $(20), (10, 10), (10, 5, 5), (5, 5, 5, 5)$. The amount of CHF 0.04 can be owned in no way, while there is exactly one way to own CHF 0.00 (you cannot have 4 centimes in your wallet, but you *can* have no money at all in your wallet).

a) Solve the problem for a given input amount, by writing the following function (all values to be understood as centimes).

```
// PRE: [begin, end) is a valid nonempty range that describes
//      a sequence of denominations d_1 > d_2 > ... > d_n > 0
// POST: return value is the number of ways to partition amount
//       using denominations from d_1, ..., d_n
unsigned int partitions (unsigned int amount,
                         const unsigned int* begin,
                         const unsigned int* end);
```

On the course webpage you find the program partitions_template.cpp where you can plug-in your code and use it to test your function. Use the program to determine in how many ways you can own CHF 1, and CHF 10.

b) **Challenge.** Modify your program such that it computes the solution *quickly* (say in less than one second) also for large amounts like CHF 50 or CHF 100.