Course 'Imperative Programming' (IPC031)

Bonus Assignment 11: *Pattern-matching with wild-cards on strings*

1. Assignment

In this assignment you develop the recursive function:

```
bool match_pattern (string pattern, string source)
```

which decides whether `pattern` matches `source` *entirely*. The string `pattern` consists of 'normal' characters that have to match precisely. Additionally, it contains two special *wildcard* symbols:

- '`.`', which represents *exactly one* arbitrary character. So "`hu.t`" matches "`hurt`" and "`hunt`" but it does not match "`hut`" (which is too short) and "`hunts`" (which is too long).
- '`*`', which represents an arbitrary number (0 or more) of any characters. So "`Cu*e`" matches "`Cue`", "`Cure`", "`Curve`", and so on. It does not match "`A Cue`", "`A Cure`", "`A Curve`", and so on because the first characters do not match.

Every other character is interpreted literally. Wildcard patterns can be simplified by repeatedly applying the following two rules until neither one is applicable:

- "`**`" → "`*`"
- "`*.`" → "`.*`"

Hence, the search term "`*.*.*.*`" is equal to "`...*`" after simplification. It matches any source text consisting of at least 3 characters.

The '`*`' wildcard can usually be matched in more than one way. For instance, in "`Dream Theater`" the characters '`r`', '`e`', '`a`' occur several times, just as the combination "`ea`". The wildcards in the pattern string "`*ea*`" can match in two ways, viz. ("`Dr`", "`m Theater`") and ("`Dream Th`", "`ter`").

Design and implement the *recursive* function `match_pattern`, in a way that enables it to handle wildcard symbols in its first argument. The function yields true only if a match can be found with the second argument. The algorithm is multiply recursive. It is up to you to decide whether you assume that the pattern is already simplified according to the rules mentioned above. If you decide not to assume this, you need to implement and apply the simplification rules as well.

Note that you can readily test the implementation by making the function `match` in *assignment 8: Music database and queries* call your function `match_pattern`.

2. Products

As product-to-deliver you only need to upload to Brightspace *"main.cpp"* that you have created with solutions for each part of the assignment.

⇒ **Deadline:** *Tuesday, 4th of December 2018, 8:30h*