Universitetet i Oslo
Institutt for Informatikk

I. Yu, D. Karabeg

INF2220: algorithms and data structures

Mandatory assignment 3

Text-algorithm & BM

Issued: 26. 10. 2016

Due: 09. 11. 2016

The task here is to implement a (rather mild) generalization of a *pattern matching* (string matching) algorithm as presented in the lecture on text algorithms. The generalization is to offer the user the possibility of adding *"wild-cards"* to the needle. A wild-card is a special "symbol" which is meant to *match any letter*. You will use the *underscore* character ("_") as wildcard. For simplicity, we assume the text (the haystack) does not contain the "_"-symbol.

For illustration, the "*needle*" `c_g` can be found in the following "haystack" `cogwrgaccag` two times (at indexes 0 and 8).

What you need to implement is:

1. You must implement a pattern matching algorithm using the *bad character shift* optimization discussed in connection with the Boyer-Moore-Horspool algorithm.

2. Your implementation must be able to handle patterns/`needles` with multiple wild-cards (i.e., more than one "_").

3. Your algorithm should find *all* positions in the haystack where the needle is found.

4. Your program should output all the positions in an easily understandable manner.

5. Because the content of matches can vary with haystacks your program should output the text in the haystack that matched the needle.

6. Your program should take two filenames as an argument. The first containing the needle and the second containing the haystack (i.e., `java AssignmentThree <needle> <haystack>`)

1 How to deliver

The assignment should be carried out individually and delivered through `https://devilry.ifi.uio.no/`.

- The implementation language is Java.

- Your implementation must compile on the LINUX machines at the University.
- Your delivery should contain
  - Compilable (and afterwards runnable) source file(s) of your implementation.
  - A few *test cases* that thoroughly tests your algorithm and the result it produces. Test different positions and numbers of wildcards in the needle. Remember testing edge-cases such as empty needles/haystack. Describe (in the `README`-file) what the correct output for your test cases should be.
  - A `README`-file which contains:
    1. An *explanation* of your algorithm and why it works. Use common sense there, for instance, a lengthy repetition of the principles of Boyer-Moore-Horspool etc is not needed/wished. Concentrate on *your* solution and highlight, if it helps understanding, special points of the code.
    2. How to compile your program (ie. javac *.java)
    3. Which file includes the main-method
    4. Any assumptions you have made when implementing the assignment
    5. Any peculiarities about your implementation
    6. The status of your delivery (what works and what does not)
    7. Give credit if your code is heavily influenced by some source (ie. teaching material)

Good luck!