# Homework: Collection Data Structures and Libraries

This document defines the **homework assignments** for the ["Data Structures" course @ Software University](https://softuni.bg/trainings/1147/Data-Structures-June-2015). Please submit a single zip / rar / 7z archive holding the solutions (source code) of all below described problems.

## Products in Price Range

Write a program to read a **large collection of products** (name + price) and efficiently **find the first 20 products** in the **price range [a…b]** ordered by price. Test for **500 000 products** and **10 000 price searches**.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 7  apples 2.50  bananas 1.20  milk 1.33  water 1.30  beer 0.95  cheese 8.5  muffin 0.5  0.95 2 | 0.95 beer  1.20 bananas  1.30 water  1.33 milk |

##### Hints (Click on the arrow to show)

You may use OrderedBag<Product> or OrderedMultiDictionary<int, Product> and **sub-ranges**.

## String Editor

You have to implement a string editor that starts from empty string and executes sequence of commands:

* **APPEND** some\_string – appends given string at the end of the text. Print "OK" on success.
* INSERT some\_string position – inserts given string at given position. Print "OK" on success. Print "ERROR" in case of invalid position.
* DELETE start\_index count – deletes the specified substring. Print "OK" on success**.** Print "ERROR" in case of invalid substring.
* REPLACE start\_index count some\_string – replaces the specified substring with the specified string. Print "OK" on success. Print "ERROR" in case of invalid substring.
* PRINT – prints the string in the editor.
* END – stops the program execution. Passed as last command in the input. Does not print anything.

Ensure your programs runs **efficiently** for tens of thousands of commands.

|  |  |  |
| --- | --- | --- |
| **Input** | **Editor State** | **Output** |
| APPEND pesho  APPEND 123  INSERT 0 456  DELETE 1 2  DELETE 100 200  PRINT  REPLACE 1 5 kiro  REPLACE 700 800 hi  APPEND Hello C#  PRINT  END | pesho  pesho123  456pesho123  4pesho123  4pesho123  4pesho123  4kiro123  4kiro123  4kiro123Hello C#  4kiro123Hello C# | OK  OK  OK  OK  ERROR  4pesho123  OK  ERROR  OK  4kiro123Hello C# |

##### Hints (Click on the arrow to show)

Use **rope of chars**, e.g. Wintellect.PowerCollections.BigList<char>. Try also using a StringBuilder and compare the difference performance for large enough sequence of commands.

## \* Fast Search for Strings in a Text File

Write a program that finds a **set of strings** (e.g. 1000 strings) in a **large** **text** (e.g. 100 MB text file). Print how many times **each string** occurs in the text **as substring**. Ensure your program works fast enough!

The input comes from the console in the following format:

* The first line holds the **number of strings** for searching s.
* The next s lines hold the **strings** to be found in the text – one string per line.
* The next line holds an integer l – the **number of input lines**.
* The next l lines hold the **input text**.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  C#  at  UNI  a  6  Hello, I am studying C# at SoftUni.  C# is my favorite language. It is unique!  I like C# and Java.  Atmosphere at SoftUni is great.  SoftUni has very mature and practical learning system.  "Ratamahatta" is Sepultura's ninth official single. | C# -> 3  at -> 7  UNI -> 4  a -> 24 |

##### Hints (Click on the arrow to show)

* Note that the string matching is **case-insensitive**.
* Match the strings as **substrings** (part of word), not as words.
* A correct, but **slow** solution is to use String.IndexOf() to find the occurences of each input string in each input line of the text.
* A **faster** solution is to **scan the input text char by char**, append the chars in a buffer and check after each char added if the buffer ends by some of the strings.
* A really **fast solution** is to use [**Aho-Corasick's algorithm**](https://en.wikipedia.org/wiki/Aho%E2%80%93Corasick_algorithm) and the [**trie**](https://en.wikipedia.org/wiki/Trie) data structure.
* A detailed **analysis** and **solution** with explaination of this problem can be found in the "C# Fundamentals" book: <http://www.introprogramming.info/english-intro-csharp-book/read-online/chapter-26-sample-programming-exam-topic-3/>.