# Exercises: Ropes and Tries

This document defines the **exercise assignments** for the ["Data Structures" course @ Software University](https://softuni.bg/opencourses/data-structures). You can submit your code in the SoftUni Judge System - <https://judge.softuni.bg/Contests/639/Ropes-and-Tries-CSharp-Exercise>.

## Text Editor

Your task is to implement a **text editor**. The program will be used by **multiple users** at a time. Each user will have its own **text**, which he should be able to **edit**. Also, we should be able to see **all of the users** that are using our application. You will receive commands, which you need to execute:

* login {username} – the given user can start executing commands
* logout {username} – the given user can no longer edit his string
* {username} {command} {parameters} – this is the pattern we will be using for the following **user commands**:
  + insert {index} {string} – inserts the given string in the given position
  + prepend {string} – inserts the given string in the beginning of the string
  + substring {start index} {length} – replaces the user string with a substring from it
  + delete {start index} {length} – removes part of the user string
  + clear – deletes the user string
  + length – returns the length of the user string
  + print – returns the user string
  + undo – reverts the last operations on the user string. Can be used multiple times
* users – returns all users currently logged in
* users {prefix} – returns all users starting with the given prefix
* end – stops the program

You will be given **skeleton** containing **interface** with the desired functionality. User commands should be **ignored** if the user is **not logged** in. If the same user tries to login twice, delete his old string. Choose the appropriate data structures, which will handle big input size. Command parameters will be split with exactly 1 **whitespace**. All commands that lead to Exceptions, should be ignored.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| login pesho  pesho prepend "hello"  pesho print  pesho length  end | hello  5 |
| **Input** | **Output** |
| login pesho  pesho prepend "hello"  pesho insert 0 "H"  pesho print  pesho delete 1 1  pesho print  pesho insert 5 ", my name is pesho"  pesho print  pesho substring 0 5  pesho print  end | Hhello  Hello  Hello, my name is pesho  Hello |

|  |  |
| --- | --- |
| **Input** | **Output** |
| ivan prepend "hi"  pesho substring 0 4  pesho print  login pesho  pesho prepend "hi, again"  login pesho  pesho print  pesho prepend "HI!"  pesho length  pesho clear  pesho print  pesho undo  pesho print  end | 3  HI! |
| **Input** | **Output** |
| login pesho  login penka  login ivan  login mitko  login petyo  login petrov  login ivo  login stamat  login pencho  users  users pe  logout petrov  users pet  end | pesho  penka  ivan  mitko  petyo  petrov  ivo  stamat  pencho  pesho  penka  petyo  petrov  pencho  petyo |