T-79.4202 **Autumn 2013**
Principles of Algorithmic Techniques
Programming Tutorial 1, October 8–9
Problems

For both of the problem sets described below, design, implement and test a program for *one* of the alternatives. You may use any programming language of your choice, but please observe that your programs must meet the input/output interface requirements given on the opposite side. Valid input/output pairs for testing your programs are available in Noppa.

The **deadline** for submitting your solutions is Monday October 7, 23:59 p.m. In order to earn credit, you must also attend either one of the tutorial sessions on 8–9 October, and be ready to explain and demonstrate your solutions. For further details and instructions, see `https://noppa.aalto.fi/noppa/kurssi/t-79.4202/harjoitustyot`.

1. **Algorithms with numbers:**

   (a) Write a program that computes modular division using extended Euclid's algorithm. Given a prime p and two integers $a \in [0, p-1]$ and $b \in [1, p-1]$, the program computes and returns $c \in [0, p-1]$ such that $a \equiv bc \pmod{p}$.

   (b) Write a program that finds multiplicative inverse modulo a prime p for an integer $a \in [1, p-1]$ with modular exponentiation. Do not use extended Euclid's algorithm! Hint: You might find Fermat's little theorem ($a^{p-1} \equiv 1 \pmod{p}$) useful.

2. **Divide and conquer:**

   (a) Let $A[1, \ldots, n]$ be a sorted array of distinct integers. Write a program that finds out whether there is an index i for which $A[i] = i$. If there is more than one such index i, it suffices to return only one of them. Use a divide and conquer algorithm that runs in time $O(\log n)$. [1]

   (b) Write a program that gives the nth smallest element in the union of two sorted arrays of size n, $A[1, \ldots, n]$ and $B[1, \ldots, n]$. Use a divide and conquer algorithm that runs in time $O(\log n)$. [2]

See the reverse side for implementation details.

[1] Exercise 2.17 in Dasgupta et al. "Algorithms"
[2] Modification of Exercise 2.22 in Dasgupta et al. "Algorithms"

Input/output Interface

README of the package in Noppa:

1. The programs must take the required values (32-bit unsigned integers) as command-line arguments or from standard input (`stdin`) and print the result to standard output (`stdout`). The programs should operate, for example, as follows:

```
user@computer:$ ./asgn01_1a 12 7 17
12 / 7 mod 17 = 9

user@computer:$ ./asgn01_1b 9 11
1 / 9 mod 11 = 5
```

2. The programs must take the file name(s) of the array file(s) as command-line arguments or read the arrays from `stdin`. Positions are numbered from 1 to n. The programs must print the result to `stdout`. The programs should operate, for example, as follows:

```
user@computer:$ ./asgn01_2a 01.array
No index i for which A[i]=i
user@computer:$ ./asgn01_2a 02.array
A[2] = 2

user@computer:$ ./asgn01_2b 01.array
Element at position 7 in sorted union is 7
```

   The format of array text files is the following: $n\ A[1]\ A[2] \ldots A[n]$. For example, the text file for $\{1,3,7,10,14\}$ contains `5 1 3 7 10 14`. The values $A[i]$ are 32-bit signed integers. In assignment 2(b), the arrays are given in one file as follows: $n\ A[1]\ A[2] \ldots A[n]\ B[1]\ B[2] \ldots B[n]$.