1st tutorial

Introduction

Tutorial principles.

- **You** want to learn something.
- **I** am trying to create space for it.
- That's why the highest priority is what **you** need:
  - ... understand something from the lecture
  - ... understand a task from the previous tutorial
  - ... understand some homework assignment, etc.
- **Don't be afraid** to ask questions
- A few general **tips**:
  - When you feel lost, try to play with examples. (E.g., if you can't solve a task for general n, k, try what it says for small values. Don't know what an algorithm does? Try it on small input.)
  - Try to explain where you got stuck to your classmates. It will help you understand what's really going on, get your thinking clearer.
  - If you still don't know, reach out, I will try to help.

Logistical details.

- Email me at `koutecky+ads1@iuuk.mff.cuni.cz`.
- The tutorial website is at `http://research.koutecky.name/db/teaching:ads12324_tutorial`
- You need to get 100 points to gain credit. Points are awarded for:
  - **Homework.** There will be a batch of tasks about every other week. The total amount of points obtainable will be at least 150. You need to solve at least one task from each batch, not necessarily within the deadline.
  - **Quizzes.** At the beginning of each tutorial, there will be a short quizz where I will ask about something basic from the previous lectures. The purpose is to motivate you to come prepared – the tutorial is a space to apply what you have learned in the lecture, not to repeat the lecture ;-)
  - **Grading** each other's homework. If you are confident about your solution, you can email me and say you want to be a student grader; if your solution is indeed correct, I will let you correct the other solutions and gain extra points for it.
  - **A small project.** The topic needs to be really good and original.
  - *I don't require attendance. If you feel that you can benefit more from self-study or anything else, go for it.*
- Tasks are handed into the OWL system: `https://kam.mff.cuni.cz/owl/`
- *What's the best space to discuss? Discord? Telegram? ...?*
- If you need further help, try getting it from your classmates. If there's more of you who need help with the same thing, email me and we'll set up a meeting :) Also **DO** use the student guides and mentors, they are here for you!

1st tutorial

Algorithms. A newspaper tasks a reporter to write an article on the working conditions of a certain company. Thus, he has to try as many positions in this company as possible. However, he would like his salary to keep increasing. The company posts advertisements for various positions at certain times. Mathematically speaking, we are given a sequence $p_1, \ldots, p_n$ of positive real numbers, and we are looking for a longest strictly increasing subsequence.
How can we solve this problem:

a) According to the definition
b) Recursively
c) Recursively with memoization
d) As a graph problem
e) With a clever data structure

Complexity.

- What do we neglect and why?
- What does it mean **in words**: $O(n), \Omega(n), o(n), \omega(n)$? How to write it **formally** (with quantifiers, as a formula / logical expression)?

Asymptotics. Find as many as possible asymptotic relationships between the following functions: n, $\log n$, $\log\log n$, $\sqrt{n}$, $n^{\log n}$, 2^n, 3^n, $n^{3/2}$, $n!$, n^n.

$\mathcal{O}$-sum. Let f_1, f_2 be functions s.t. $f_1 \in \mathcal{O}(f_2)$. Prove that $f_1 + f_2 \in \mathcal{O}(f_2)$.

$\mathcal{O}$-max. Prove that $\mathcal{O}(f + g) = \mathcal{O}(\max\{f, g\})$.

Recursion. Guess what each function does, prove it, and analyze time and space complexity.

```
g(x,y):
if y==0 => return 0
else if even(y) => return 2*g(x, y/2)
else => return 2*g(x, y/2) + x

h(x,y):
if x<y => return (0,x)
else:
(a,b) <- 2*h(x/2, y)
if odd(x) => b <- b+1
if b>=y => a <- a+1, b <- b-y
return (a,b)

d(x,y):
if x==y => return x
if even(x) and even(y): return 2*d(x/2, y/2)
if even(x): return d(x/2, y)
if even(y): return d(x, y/2)
if x>y: return d(x-y, y)
else: return d(x, y-x)
```