

**Proč ne  $x_1$ ?** Proč navrhujeme volit za pivota prvek  $x_{\lfloor n/2 \rfloor}$ , a ne třeba  $x_1$  nebo  $x_n$ ?

**Průměr.** Jak by dopadlo, kdybychom na vstupu dostali posloupnost reálných čísel a jako pivota používali aritmetický průměr?

**LinearSelect s konstantní pamětí.** Upravte funkci LINEARSELECT tak, aby si vystačila s konstantně velkou pomocnou pamětí. Prvky ve vstupním poli můžete libovolně přeskupovat.

**Proč pětice?** Proč při vybírání  $k$ -tého nejmenšího prvku používáme zrovna pětice? Fungoval by algoritmus s trojicemi? Nebo se sedmicemi? Byl by pak stále lineární?

**$\epsilon$ -sít.** Pro  $n$ -prvkovou množinu prvků  $X$  a číslo  $\epsilon$  ( $0 < \epsilon < 1$ ) definujeme  $\epsilon$ -sít jako posloupnost  $\min X = x_0 < x_1 < \dots < x_{1/\epsilon} = \max X$  prvků vybraných z  $X$  tak, aby se mezi  $x_i$  a  $x_{i+1}$  vždy nacházelo nejvýše  $\epsilon n$  prvků z  $X$ . Pro  $\epsilon = 1/2$  tedy počítáme minimum, medián a maximum, pro  $\epsilon = 1/4$  přidáme prvky ve čtvrtinách,  $\dots$ , a při  $\epsilon = 1/n$  už třídíme. Složitost hledání  $\epsilon$ -sítě se tedy v závislosti na hodnotě  $\epsilon$  bude pohybovat mezi  $\mathcal{O}(n)$  a  $\mathcal{O}(n \log n)$ . Najděte algoritmus s časovou složitostí  $\mathcal{O}(n \log(1/\epsilon))$ .

---

**Kopec.** *Kopcem* nazveme podposloupnost, která nejprve roste a pak klesá. Vymyslete algoritmus, který v zadané posloupnosti nalezne nejdelší kopec (jako podposloupnost).

**Knihovna 1.** Mějme posloupnost  $n$  knih. Každá kniha má nějakou šířku  $s_i$  a výšku  $v_i$ . Knihy chceme naskládat do knihovny s nějakým počtem polic tak, abychom dodrželi abecední pořadí. Prvních několik knih tedy půjde na první polici, další část na druhou polici, a tak dále. Máme zadanou šířku knihovny  $S$  a chceme rozmístit police tak, aby se do nich vešly všechny knihy a celkově byla knihovna co nejnižší. Tloušťku polic a horní a spodní desky přitom zanedbáváme.

**Knihovna 2.** Podobně jako v předchozím cvičení chceme navrhnout knihovnu, jež pojme dané knihy. Tentokrát ovšem máme zadanou maximální výšku knihovny a chceme najít minimální možnou šířku. Pokud vám to pomůže, předpokládejte, že všechny knihy mají jednotkovou šířku.

**Depeše.** Dešifrovali jsme tajnou depeši, ale chybí v ní mezery. Známe však slovník všech slov, která se v depeši mohou vyskytnout. Chceme tedy rozdělit depeši na co nejméně slov ze slovníku.