- Jak poznat, že program 'funguje'?
- Jak porovnat délku běhu algoritmů bez měření?
- Jak rozlišujeme 'efektivní' a 'neefektivní' algoritmy? Znamená to, že 'neefektivní' jsou v praxi pomalejší?

1. **vajíčka** Máme mrakodrap s n patry a několik identických naklonovaných mimozemských vajíček. Chceme najít patro k, t.ž. hodem z k-tého patra se vajíčko ještě nerozbije, ale hodem z $k+1$-ního patra už ano. Kolik hodů potřebujeme pro určení k? Co když máme omezený počet vajíček?
2. **mocniny** Máme číslo n a mocninu k. Kolika operacemi bez umocňování můžeme vypočítat m^k? Co se stane, když můžeme umocňovat na konstantní mocniny?
3. **součet** Máme posloupnost kladných čísel délky n. Hledáme úsek se součtem přesně k. Co se stane, pokud chceme nejkratší takový úsek?
4. **výtah** Máme budovu o n patrech, a několik lidí čekajících na výtah v různých patrech. Kolika přesuny mezi sousedními patry je může výtah všechny dopravit? Co když máme místo výtahu tramvaj?
5. **nulová podmatice** Máme matici velikosti $n \times n$. Chceme najít její největší nulovou podmatici.
6. **porovnejte asymptoticky** - i s předchozími složitostmi
   - $n^{\mathcal{O}(1)}$
   - $\mathcal{O}(f(n) + g(n))$
   - $\mathcal{O}(2^n)$, $O(3^n)$, $2^{\mathcal{O}(log(n))}$
   - $\mathcal{O}(n^c)$ pro malé c

Domácí úkol: Mějme na vstupu číslo n. Chceme najít celočíselnou odmocninu $m = \lfloor \sqrt{n} \rfloor$, přičemž nemáme k dispozici operaci odmocniny (pouze násobení, sčítání, ...). Navrhněte algoritmus, který najde m s využitím binárního vyhledávání. Úkol by měl obsahovat:

- stručný popis algoritmu a jeho pseudokód
- důkaz správnosti a konečnosti
- určení asymptotické časové a prostorové složitosti ($\mathcal{O}$ i Ω)

Bonusová otázka: Definujme k jako $2^{k-1} \leq m < 2^k$. Jak můžeme najít k v čase $\mathcal{O}(loglog(m))$? Můžeme s pomocí tohoto postupu asymptoticky zrychlit náš algoritmus pro nalezení m?