Počítačová algebra - cv. 1

Pátek 28. 2. 2020

1. Napište příkaz, který vynásobí dvě čísla v $\mathbb{Z}$ a v $\mathbb{Z}_{2^{32}}$ a vypíše výsledek.

2. Spočtěte NSD dvou polynomů z $\mathbb{Z}_3[x]$ pomocí zabudované funkce (jak se taková funkce asi jmenuje? - anglicky se největší společný dělitel řekne greatest common divisor).

3. Příkaz `list(f)` vrací pro polynom f jedné proměnné seznam jeho koeficientů. Uměli byste s jeho pomocí definovat funkci `deg`, vracící stupeň polynomu a funkci `lc` vracící jeho vedoucí koeficient?

4. Napište program, který počítá Fibonacciho posloupnost F_n a vypíše F_{512} (pozor! naivní rekurzivní definice vede na exponenciální složitost).
   * Jak si možná vzpomínáte z lingebry, zobrazení

   $$(F_i, F_{i+1}) \mapsto (F_{i+1}, F_{i+2})$$

   je lineární a lze jej proto reprezentovat maticí. I bez hledání vlastních čísel a diagonalizace lze (s pomocí binárního mocnění) spočítat vysoké mocniny této matice a tím i číslo F_k podstatně efektivněji než naivním výpočtem jednoho Fibonacciho čísla po druhém. Zkusíte implementovat tuto maticovou metodu?

5. Naprogramujte funkci, která pro $a, b \in \mathbb{Z}$ spočte NSD(a, b).
   * Jak rychle algoritmus běží pro náhodná velká a, b?

6. Vytvořte řešič kvadratických rovnic. Pro zadané koeficienty $a, b, c \in \mathbb{Q}$ vraťte seznam reálných kořenů rovnice $ax^2 + bx + c = 0$. Může se vám hodit příkaz `seznam.append(prvek)`.
   * Namísto v $\mathbb{R}$ pracujte $\mathbb{C}$ (typ `Complex` v Pythonu nebo `CC` v Sagi) a vracejte vždy dvojici kořenů.
   ** Obecná reálná (a komplexní) čísla lze v počítači reprezentovat pouze přibližně. My zde ale nepotřebujeme pracovat s obecnými čísly, pouze se zlomky obsahujícími odmocniny, tj. s nějakým $\mathbb{Q}[\sqrt{D}]$. Zamyslete se nad tím, co by bylo potřeba abychom zbytečně neztráceli přesnost.