7. Predikáty a vzory, interpolace a aproximace

Některé užitečné predikáty:

Predikát	Význam
`EvenQ[n]`	Je `n` sudé?
`OddQ[n]`	Je `n` liché?
`PrimeQ[n]`	Je `n` prvočíslo?
`Positive[x]`	Je `x` kladné?
`Negative[x]`	Je `x` záporné?
`NonNegative[x]`	Je `x` nezáporné?
`NonPositive[x]`	Je `x` nekladné?
`MemberQ[seznam,x]`	Je `x` prvkem seznamu?

Predikáty jsou také `<`, `>`, `<=`, `>=`, `==`, `!=`.

Výběr prvků seznamu, pro které predikát dává `True`: `Select[seznam,predikát]`

Logické spojky: `p || q` (`p` nebo `q`) `p && q` (`p` a zároveň `q`) `!p` (negace `p`)

Konstrukce	Význam
`identifikátor_`	Libovolný výraz
`identifikátor_Head`	Výraz s hlavou `Head`
`identifikátor___`	Nula nebo více výrazů oddělených čárkami
`vzor?predikát`	Výraz, který vyhovuje vzoru a predikát pro něj dává `True`
`vzor/;podmínka`	Výraz, který vyhovuje vzoru a splňuje podmínku

Počet prvků seznamu vyhovujících vzoru: `Count[seznam,vzor]`
Prvky seznamu, které (ne)vyhovují vzoru: `Cases[seznam,vzor]` `DeleteCases[seznam,vzor]`

Lagrangeův interpolační polynom: `InterpolatingPolynomial[data,x]`
`data` mají tvar {a_1,a_2,...} nebo {{x_1,y_1},{x_2,y_2},...}.

Po částech polynomiální interpolace: `Interpolation[data]`
Výsledkem je tzv. interpolační funkce (`InterpolatingFunction`).
Stupeň interpolačních polynomů lze nastavit volbou `InterpolationOrder->n` (implicitně 3).

Numerické řešení diferenciální rovnice nebo soustavy rovnic:
`NDSolveValue[{rovnice,počáteční podmínky},x[t],{t,min,max}]`
`NDSolveValue[{`rovnice$_1$,rovnice$_2$`,...,počáteční podmínky},{x[t],y[t],...},{t,min,max}]`
Příkaz vrací řešení ve tvaru interpolační funkce.

Aproximace metodou nejmenších čtverců:
`Fit[data,{`f$_1$`[x],`f$_2$`[x],...},x]` hledá lineární kombinaci zadaných funkcí.
`FindFit[data,f[x,a,b,...],{a,b,...},x]` hledá hodnoty parametrů v obecném modelu.

Cvičení

1. Přirozené číslo n se nazývá dokonalé, jestliže je rovno součtu všech svých dělitelů kromě n, tj. např. $6 = 1+2+3$ je dokonalé. Definujte predikát `perfectQ[n]`, který vrátí `True`, právě když n je dokonalé (použijte funkci `Divisors[n]`, která vrací seznam všech dělitelů n). Pomocí `Select` najděte všechna dokonalá čísla menší než 10 000 (jsou čtyři).

2. Prvočíselnými dvojčaty nazveme každou dvojici prvočísel tvaru $(p, p+2)$ (např. 11 a 13). Pomocí `Select` nebo `Cases` najděte všechna prvočíselná dvojčata menší než 1000 (je jich 35).

Návod: Místo práce s dvojicemi čísel je jednodušší najít všechna čísla p taková, že p a $p+2$ jsou prvočísla. Ze seznamu takto získaných čísel p pak zkonstruujte seznam dvojic $(p, p+2)$, můžete využít např. `Transpose`.

3. Najděte polynom, který má v bodech 1, 2, 3, 4 hodnoty 3, 5, 8, 13. Přesvědčete se o správnosti výsledku pomocí `ListPlot` a `Plot`.

4. Diferenciální rovnice $x''(t) = -\sin x(t)$ popisuje pohyb kyvadla; $x(t)$ je úhel, o který je kyvadlo vychýleno z rovnovážné polohy v čase t (tj. $x(t) = 0$ znamená, že kyvadlo je v čase t v nejnižší poloze). Najděte numerické řešení této diferenciální rovnice s počátečními podmínkami a) $x(0) = 3.1$, $x'(0) = 0$, b) $x(0) = 3.2$, $x'(0) = 0$. Zobrazte grafy obou nalezených funkcí.

5. Najděte numerické řešení tzv. Lorenzovy soustavy diferenciálních rovnic

$$x'(t) = 10(y(t) - x(t)), \quad y'(t) = 28x(t) - y(t) - z(t)x(t), \quad z'(t) = -\frac{8}{3}z(t) + x(t)y(t)$$

na intervalu $t \in [0, 50]$ s počátečními podmínkami $x(0) = y(0) = z(0) = 5$. Zobrazte nalezenou trojici funkcí x, y, z jako prostorovou křivku (použijte `ParametricPlot3D`).

6. Fibonacciho čísla jsou definována hodnotami $F_1 = F_2 = 1$ a rekurentním vzorcem $F_{n+2} = F_{n+1} + F_n$. Číslo F_n lze v Mathematice získat pomocí `Fibonacci[n]`. Naprogramujte funkci, která pro zadané $n \in \mathbf{N}$ najde všechna $k \in \{1, \ldots, n\}$ taková, že F_k je prvočíslo. Lze dokázat, že je-li $k \geq 5$ a F_k je prvočíslo, pak také k je prvočíslo. Pokuste se toto tvrzení experimentálně ověřit.

7. Zjistěte, co dělá následující funkce, pokud na vstupu zadáme seznam čísel:

```
maxima[s_List]:=s //.{a___,b_,c_,d___}/; c<=b -> {a,b,d}
```

8. Funkce $f : \mathbf{N} \to \mathbf{N}$ je definována počátečními hodnotami $f(1) = 34$, $f(2) = 5$ a rekurentními vzorci $f(3k) = 10f(k)+76$, $f(3k+1) = 10f(k)-2$, $f(3k+2) = 10f(k)+8$, kde $k \in \mathbf{N}$. Naprogramujte funkci pro výpočet $f(n)$. Sestrojte graf hodnot $f(n)$ pro $n \in \{1, \ldots, 100\}$. Pro jaká čísla $n \in \{1, \ldots, 100\}$ platí, že $f(n) > \max\{f(1), \ldots, f(n-1)\}$? Zkuste stejnou otázku zodpovědět pro čísla z intervalu $\{1, \ldots, 10\,000\}$.

Návod: Sestrojte seznam dvojic $(n, f(n))$ pro $n \in \{1, \ldots, 10\,000\}$. Poté použijte vhodnou modifikaci funkce `maxima` z předchozího cvičení k nalezení seznamu všech dvojic $(n, f(n))$ takových, že $f(n) > \max\{f(1), \ldots, f(n-1)\}$.

9. Naprogramujte funkci `komprese` pro „komprimaci“ seznamů s opakujícími se prvky. Výstupem bude seznam dvojic tvaru `{prvek,počet opakování}`, tj. např. `komprese[{a,a,a,b,b,c,a,a,a}]` vrátí seznam `{{a,3},{b,2},{c,1},{a,3}}`.

Návod: Nejprve převeďte zadaný seznam `{x,y,z,...}` na tvar `{{x,1},{y,1},{z,1},...}`, poté na něj aplikujte vhodné lokální přepisovací pravidlo.