Řešení rekurentních problémů

- **Rekurentní vztah**
    - $Y_{i+1} = F(Y_{i-k} \dots, Y_{i-2}, Y_{i-1}, Y_i)$, kde:
      $Y_{i-k}, Y_{i-k+1}, \dots, Y_i, Y_{i+1}$ – zevšeobecněné proměnné,
      F – funkce na základě které získáme z hodnot
      $Y_{i-k} \dots, Y_{i-2}, Y_{i-1}, Y_i$ hodnotu Y_{i+1}
      $i,k \geq 0$ – počet předchozích kroků řešení
    - **Vlastnosti:**
        - Pro výpočet další hodnoty potřebujeme pouze k+1 posledních hodnot.
        - Musí existovat takové n, že Yn je požadovanou hodnotou, po jejímž získání iterační výpočet končí.
        - Musíme umět rozpoznat požadovanou hodnotu.

- **Posloupnosti**
    - Uvažujme rekurentní vztah:
        - $Y_{i+1} = F(Y_i)$
    - Pro výpočet Y_{i+1} Je potřeba zjistit hodnotu Y_i.
    - Na začátku musí být dané Y0, ze kterého celý výpočet začíná.
    - Postupně dostáváme hodnoty Y1, Y2, …, Yn, pro které platí:
        - 1. $Y_{i+1} = F(Y_i) \quad pro\ i \geq 0$
        - 2. $Y_i \neq Y_j\ pro\ i \neq j$
        - 3. $Y_i\ pro\ i < n$ – nesplňuje podmínky požadované
        - Hodnoty.
        - 4. Y_n – splňuje podmínky požadované hodnoty.

    - **Algoritmické schéma pro posloupnosti**
        - Algoritmus realizující vztah $Y_{i+1} = F(Y_i)$:
            - 1. $Y = y_0$;
            - 2. $while\ (\neg B(Y)) \quad Y = F(Y)$;
        - Všeobecné symboly:
            - Proměnná Y
            - Predikátový symbol B
            - Funkční symbol F
        - Predikát B(Y) – podmínka požadované hodnoty závislá na hodnotě Y
        - Nejde o řešení konkrétního rekurentního vztahu

- **Příklad – Výpočet druhé odmocniny**
  - **Výpočet druhé odmocniny reálného čísla A≥0 lze popsat rekurentním vztahem:**
    - $\boldsymbol{y_{i+1} = \frac{1}{2}\left(\frac{A}{y_i} + y_i\right)}$
  - y_0 – pro jednoduchost lze volit 1.
  - Způsob ukončení algoritmu:
    - Obvykle se výpočet opakuje, dokud $|y_i - y_{i+1}|$ není menší než nějaká zadaná hodnota.
    - Této hodnotě se říká přesnost výpočtu – značíme ji **eps.**
  - Pro výpočet nové hodnoty y potřebujeme jednu předcházející hodnotu – použijeme dvě proměnné:
    - Starey, novey.
  - **Výpočet**
    - $y_0 = 1$
    - $\sqrt{5} = 2.23$
    - $\boldsymbol{y_2 = \frac{1}{2}\left(\frac{5}{1} + 1\right) => y_2 = 3}$
    - $\boldsymbol{y_3 = \frac{1}{2}\left(\frac{5}{3} + 3\right) => y_3 = 2.3}$
    - $\boldsymbol{y_4 = \frac{1}{2}\left(\frac{5}{2.3} + 2.3\right) => y_4 = \cdots}$
    - $\boldsymbol{y_n \approx \sqrt{5}}$
    - **Přestanu počítat když rozdíl $\boldsymbol{y_i}$ A $\boldsymbol{y_{i+1}}$ Bude dostatečně malá**
    - $\mathbf{2.23 \pm E}$
    - $\boldsymbol{while(\neg|y_i - y_{i-1}| \geq \mathrm{E})}$
    - $\mathbf{E}$ **–** Maximální odchylka

- **Řady**
  - Uvažujme řadu vytvořenou z členů:
    - $T_0, t_1, t_2, \ldots$
  - Pro členy řady lze napsat rekurentní vztah:
    - $T_i = f(t_{i-1})$, pro i > 0
  - Nechť částečné součty jsou:
    - $S_0, s_1, s_2, \ldots$
    - $S_i = t_0 + t_1 + \ldots + t_i$
  - Pro částečné součty platí rekurentní vztah:
    - $S_0 = t_0$
    - $S_i = s_{i-1} + t_i$
  - Na základě uvedené analýzy lze sestavit modifikované algoritmické schéma pro řady:

```
1. T = t0;
2. S = T;
3. while (¬B(S, T))
   {
         T = f(T);
         S = S + T;
   }
```

    - 4. zobraz výsledek
  - **Příklad – Aproximace e^x**
    - Exponenciální funkci e^x lze aproximovat řadou
      - $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$
    - Pro částečný součet s_i lze napsat:
      - $s_i = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{i!}$

- Úprava na algoritmus:
  - $e^x = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$
  - $e_n^x = e_{n-1}^x * \frac{x}{n} \quad pro\, n > 0$

    Algoritmus:

    1. zadání x, eps
    2. inicializace: t=1, soucetRady=t, i=0
    3. opakuj pokud abs(t)≥eps
       ```
       {
         inkrementace i
         výpočet dalšího členu t
         soucetRady = soucetRady + t
       }
       ```
    4. zobrazení výsledku
  -
- **Rychlost konvergence**
  - Není stejná pro všechna reálná čísla.
  - Je velká pro malé hodnoty argumentu x (okolo nuly).
- Pro velké hodnoty x se doporučuje rozdělit argument **x** na celou část **c** a desetinnou část **d** a použít vztah:
  - $e^{c+d} = e^c * e^d$
  - Hodnota e^c se vypočítá opakovaným násobením e.

- **Heuristika**
  - Opatření:
    - K snížení náročnosti výpočtu
    - K zvýšení efektivity výpočtu
  - Posun výpočtu do intervalu nejrychlejší konvergence
    - $e^{c+d}=e^c e^d$
    - Využití periodicity u goniometrických funkcí
  - Odstranění zbytečných výpočtů
    - Zejména z těla cyklu – odsunout mimo!