Numerické metody řešení diferenciálních rovnic

Numerical methods for solving differential equations

Bc. Zdeněk Blata

Diplomová práce
2009

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2008/2009

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Zdeněk BLATA**
Studijní program:	**N 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Numerické metody řešení diferenciálních rovnic.**

Zásady pro vypracování:

1. **Zpracujte literární rešerši na dané téma.**
2. **Nastudujte možnosti řešení diferenciálních rovnic pomocí numerických metod. Vybrané metody popište.**
3. **Navrhněte a vytvořte online systém (ve formě dynamických www stránek) pro numerické řešení obyčejných diferenciálních rovnic. Uživatel si sám zadá tvar diferenciální rovnice včetně počátečních podmínek, dále bude moci volit mezi různými metodami výpočtu. Při návrhu využijte prostředí WebMathematica.**
4. **Zvolte vhodné grafické prostředí pro zobrazení výsledků řešení. Systém bude umožňovat srovnání jednotlivých metod včetně srovnání s analytickým řešením.**

Rozsah práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **RALSTON, Anthony. Základy numerické matematiky. Praha : Academia, 1978. 635 s.**
2. **VITÁSEK, Emil. Numerické metody. Praha : SNTL, 1978. 516 s.**
3. **MARČUK, G.I. Metody numerické matematiky. Praha : Academia, 1987. 528 s.**
4. **VICHER, Miroslav. Numerická matematika. Ústí nad Labem, 2003. 85 s. Skripta. ISBN 80-7044-516-5.**

Vedoucí diplomové práce: **Ing. Bronislav Chramcov, Ph.D.**
Ústav aplikované informatiky

Datum zadání diplomové práce: **20. února 2009**

Termín odevzdání diplomové práce: **27. května 2009**

Ve Zlíně dne 13. února 2009

prof. Ing. Vladimír Vašek, CSc.
děkan

L.S.

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Tato práce se zabývá numerickými metodami řešení obyčejných diferenciálních rovnic. Teoretická část objasňuje, co jsou to diferenciální rovnice, jaké jsou způsoby jejich řešení a nastiňuje jejich využití. Jsou uvedeny možnosti řešení diferenciálních rovnic pomocí numerických metod a vybrané numerické metody jsou podrobně popsány. Dále je představen pojem eLearning a jsou popsány technologie tvorby www stránek.

Praktická část se věnuje vytvořenému online systému, který je zpracovaný formou dynamických www stránek, který byl vytvořen jako elearningová pomůcka dané problematiky. Na systém je nahlíženo z pohledu uživatele i programátora.

Klíčová slova: diferenciální rovnice, numerické metody, webMathematica, eLearning

ABSTRACT

This work deals with numerical methods of solving ordinary differential equations. The theoretical part explains what are the differential equations, what are the ways to address them, and outlines their use. They are given the possibility of solving differential equations using numerical methods and some numerical methods are described in detail. It is introduced the concept of eLearning technology and described the creation of web pages.

The practical part is devoted created an online system, which is processed by means of dynamic web pages, which was created as an eLearning tool that issue. The system is viewed from the perspective of both users and programmer.

Keywords: differential equations, numerical methods, webMathematica, eLearning

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce panu Ing. Bc. Bronislavu Chramcovovi, Ph.D. za jeho cenné rady a věnovaný čas.

Dále bych rád poděkoval svým rodičům za jejich finanční trpělivost, díky které mi bylo umožněno studovat na Univerzitě Tomáše Bati ve Zlíně.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval.

V případě publikace výsledků budu uveden jako spoluautor.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD

Numerické metody řešení diferenciálních rovnic jsou součástí numerické matematiky. Tato matematická disciplína doznala velkého rozvoje teprve s rozvojem výpočetní techniky. Pro člověka totiž není únosné provádět tak velkou spoustu aritmetických operací, které numerická matematika vyžaduje. Stereotypní výpočty, které by člověku trvaly spousty a spousty hodin a někdy dokonce i dní, zvládne počítač provést během několika málo vteřin. Díky výpočetní technice je tedy umožněna aplikace nejrůznějších numerických metod v inženýrské praxi.

S numerickými metodami se setkávájí nejen studenti technicky zaměřených vysokých škol, ale také například studenti vysokých škol ekonomických. Literatury věnované dané problematice je napsáno mnoho, a proto si tahle práce vytyčila za cíl vytvoření systému, který by umožňoval praktické odzkoušení si jednotlivých numerických metod pomocí nichž lze řešit diferenciální rovnice. Systém by se tak měl stát eLearningovou pomůckou výuky.

Jelikož je systém vytvořen formou www stránek, pro přístup k němu uživateli postačuje pouze webový prohlížeč a není nutná přítomnost žádného matematického softwaru na jeho straně. O veškeré výpočty se stará server a uživateli vrací již vypočtená data.

Teoretická část práce se zaměřuje na popis numerických metod a pojednává také o webových technologiích, pomocí kterých byl systém vytvořen. Praktická část popisuje systém jak z pohledu uživatele, tak z pohledu programátora. V příloze je pak možné nalézt části programu, které jsou pro vytvořený systém klíčové.

I. TEORETICKÁ ČÁST

1 DIFERENCIÁLNÍ ROVNICE

Rovnice je jeden ze základních pojmů a také základních kamenů matematiky. Rovnice je zápis rovnosti dvou výrazů, např. $f(x) = g(x)$, kde f a g jsou funkce proměnné x (neznámé). $f(x)$, resp. $g(x)$, se nazývá levá, resp. pravá, strana rovnice. Neznámá může být číslo, *n*-tice čísel, funkce či jiný matematický objekt. Pokud jako proměnné vystupují derivace funkcí, hovoříme o tzv. diferenciální rovnici. [1] [4]

1.1 Typy diferenciálních rovnic

Základní dělení diferenciálních rovnic vychází z typu obsažených derivací.

- **Obyčejné diferenciální rovnice** (ODR) obsahují derivace hledané funkce jen jedné proměnné.
- **Parciální diferenciální rovnice** (PDR) obsahují derivace hledané funkce dvou a více proměnných, tedy parciální derivace. [4]

1.2 Řád diferenciální rovnice

Řád diferenciální rovnice je řád nejvyšší derivace, která je v ní obsažená. [4]

1.3 Řešení diferenciální rovnice

Nalézt řešení diferenciální rovnice znamená nalézt funkci v daném oboru čísel, která má příslušné derivace a vyhovuje dané diferenciální rovnici. Řešení dělíme na:

- Obecné
- Partikulární (částečné)

Obecné řešení je takové řešení diferenciální rovnice, které obsahuje libovolnou integrační konstantu.

Partikulární (částečné) řešení je řešení diferenciální rovnice, které získáme přiřazením určité číselné hodnoty každé integrační konstantě obecného řešení. Partikulární rešení můžeme v případě jednoduchých diferenciálních rovnic vypočítat analyticky. Nicméně ve velkém množství případů je analytické řešení příliš obtížné a diferenciální rovnice se řeší numericky. [4]

1.4 Příklad diferenciální rovnice

Typickým příkladem diferenciální rovnice je

$$\frac{dy(x)}{dx} = y(x) \tag{1.1}$$

nebo také zapsáno

$$y'(x) = y(x) \tag{1.2}$$

jejíž řešením je funkce

$$y(x) = C \cdot e^{x} \tag{1.3}$$

kde C je libovolná integrační konstanta. Tato konstanta se určuje z počátečních podmínek, tedy zadané hodnoty $y(x)$ v jedné hodnotě x (typicky $y(0)$). Tato rovnice je tedy podle výše uvedené klasifikace obyčejná diferenciální rovnice 1. řádu. Nalezené řešení je řešení obecné. [4]

1.5 Historie diferenciálních rovnic

Diferenciální počet vytvořili v 2. polovině 17. století nezávisle na sobě anglický matematik, fyzik a astronom *Isaac Newton* a německý filozof, vědec a matematik *Gottfried Wilhelm Leibniz*. [4]

1.6 Využití diferenciálních rovnic

V podobě diferenciálních rovnic lze formulovat velkou spoustu vědeckých problémů, a tak se diferenciální rovnice objevují snad ve všech vědeckých oborech. Největší zastoupení mají v matematice a fyzice. Lze se však s nimi také setkat např. v chemii, sociologii, ekologii atd.

Ve fyzice je ve formě diferenciálních rovnic formulována většina fyzikálních zákonů. Tyto rovnice nejčastěji popisují závislost fyzikálních veličin na čase. Hlavním představitelem těchto rovnic ve fyzice jsou rovnice pohybové, které popisují pohyb těles pod vlivem vzájemných a vnějších sil. Tyto rovnice často není možné analyticky řešit a je tedy třeba použít metod numerické matematiky. Jako příklad lze uvést problém *n* těles, která na sebe gravitačně působí. Tento problém není pro více než dvě tělesa analyticky řešitelný. [2] [4]

2 NUMERICKÉ METODY ŘEŠENÍ OBYČEJNÝCH DIFERENCIÁLNÍCH ROVNIC

Celá řada úloh, které se vyskytují v matematice, není analyticky řešitelná nebo je nalezení přesného řešení příliš obtížné. I pro zdánlivě jednoduché diferenciální rovnice, jako je např. $y' = x^2 + 3y^2$, nelze řešení určit analyticky, tj. vyjádřit je pomocí elementárních funkcí. Numerická matematika se snaží v těchto případech nalézt řešení přibližné. Tato práce se zabývá hledáním řešení obyčejných diferenciálních rovnic, ne parciálních.

Při hledání řešení diferenciálních rovnic prvního řádu, ve kterých je derivace přímo vyjádřena, hledáme reálnou funkci, která splňuje následující rovnici

$$\frac{dy(x)}{dx} = f(x, y) \tag{2.1}$$

Funkcí, které splňují tuto rovnici existuje většinou nekonečně mnoho. Jedno konkrétní řešení určíme tzv. *počáteční podmínkou*, která je nezbytnou součástí zadání úlohy. Počáteční podmínka má následující tvar

$$y(x_0) = y_0 \tag{2.2}$$

kde x_0 a y_0 jsou zadané hodnoty.

Všechny dále popisované metody lze jednoduše zobecnit i na soustavy diferenciálních rovnic. Soustava dvou rovnic pro dvě neznámé $y_1(x)$ a $y_2(x)$ má tvar

$$\frac{dy_1}{dx} = f(x, y_1, y_2) \qquad y_1(x_0) = y_{10} \tag{2.3}$$

$$\frac{dy_2}{dx} = f(x, y_1, y_2) \qquad y_2(x_0) = y_{20} \tag{2.4}$$

Podobně definujeme soustavu více rovnic.

Pokud je třeba řešit diferenciální rovnici, ve které vystupuje vyšší derivace, lze zavedením další proměnné převést tuto rovnici na soustavu rovnic prvního řádu a tuto soustavu již můžeme řešit některou z dále uvedených metod. [2] [4]

2.1 Princip numerických metod

Základním principem numerických metod je diskretizace proměnných. Přibližné řešení se nekonstruuje jako spojitá funkce, ale nagenerujeme body $x_0, x_1, x_2, \ldots$ a určujeme čísla $y_0, y_1, y_2, \ldots$, která aproximují $y(x_0), y(x_1), y(x_2), \ldots$.

Při numerickém řešení ODR vyjdeme z bodu x_0, ve kterém máme zadánu počáteční podmínku. Poté se budeme snažit najít řešení v dalších bodech x_n. Podle toho kolik k výpočtu nové hodnoty použijeme předchozích bodů, dělíme numerické metody na jednokrokové a mnohokrokové (*k*-krokové).

Jednokrokové metody využívají k výpočtu nové hodnoty pouze jednu předchozí hodnotu.

$$y_{n+1} = f(x_n, y_n, x_{n+1}) \tag{2.5}$$

Mnohokrokové (*k*-krokové) metody využívají k výpočtu nové hodnoty *k* předchozích bodů.

$$y_{n+1} = f(x_{n+1}, x_n, y_n, x_{n-1}, y_{n-1} \ldots\ldots x_{n-i+1}, y_{n-i+1}) \tag{2.6}$$

[2] [5]

2.2 Rungovy-Kuttovy metody

2.2.1 Eulerova metoda

Nejjednodušší metodou na řešení diferenciálních rovnic je Eulerova metoda. Pokud v diferenciální rovnici (2.1) nahradíme derivaci přibližným vzorcem, dostaneme

$$\frac{y_{n+1} - y_n}{h} = f(x_n, y_n) \tag{2.7}$$

kde $h \equiv x_{n+1} - x_n$ a označuje vzdálenost dvou sousedních bodů a hovoříme o něm jako o tzv. kroku řešení. Když vyjádříme y_{n+1} dostaneme Eulerovu metodu

$$y_{n+1} = y_n + h \cdot f(x_n, y_n) \tag{2.8}$$

Tento vzorec nám umožňuje při znalosti řešení v bodě x_n vypočítat řešení v bodě x_{n+1}, jedná se tedy o metodu jednokrokovou. [2] [4]

Příklad výpočtu Eulerovou metodou

Zadání: Řeště rovnici $y' = x - y$ s počáteční podmínkou $y(0) = 1$, krokem $h = 0{,}2$ a $h = 0{,}1$ na intervalu $\langle 0;0{,}6 \rangle$. Analytické řešení: $y(x) = 2e^{-x} + x - 1$.

Výpočet: Pomocí Eulerovy metody spočítáme hodnoty y_n v jednotlivých bodech x_n po krocích h.

$$y_1 = y_0 + h \cdot f(x_0, y_0) = y_0 + h \cdot (x_0 - y_0) = 1 + 0{,}2 \cdot (0 - 1) = \underline{0{,}8}$$
$$y_2 = y_1 + h \cdot f(x_1, y_1) = y_1 + h \cdot (x_1 - y_1) = 0{,}8 + 0{,}2 \cdot (0{,}2 - 0{,}8) = \underline{0{,}68}$$
$$y_3 = y_2 + h \cdot f(x_2, y_2) = y_2 + h \cdot (x_2 - y_2) = 0{,}68 + 0{,}2 \cdot (0{,}4 - 0{,}68) = \underline{0{,}624}$$

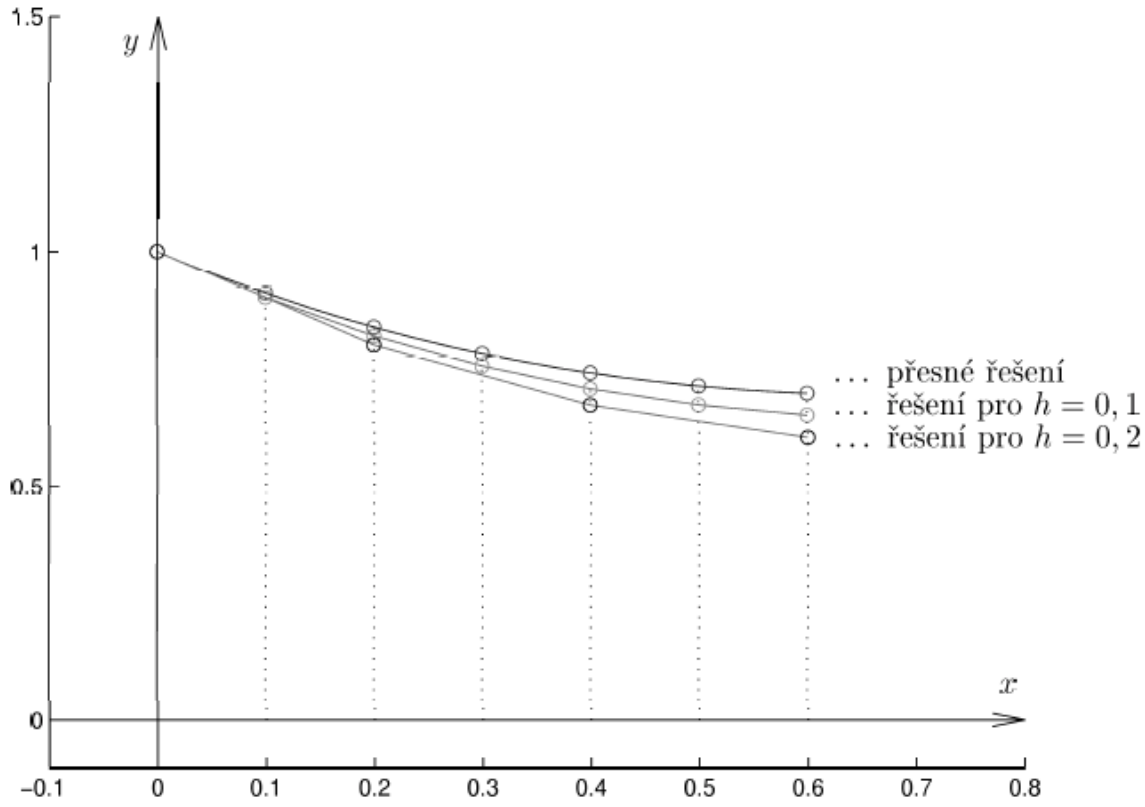
Tímto výpočtem jsme pomocí Eulerovy metody s krokem $h = 0{,}2$ zjistili přibližné řešení zadané rovnice v bodech $x_0 = 0{,}2$; $x_1 = 0{,}4$ a $x_2 = 0{,}6$. Nyní provedeme výpočty pro krok $h = 0{,}1$.

$$y_1 = y_0 + h \cdot f(x_0, y_0) = y_0 + h \cdot (x_0 - y_0) = 1 + 0{,}1 \cdot (0 - 1) = \underline{0{,}9}$$
$$y_2 = y_1 + h \cdot f(x_1, y_1) = y_1 + h \cdot (x_1 - y_1) = 0{,}9 + 0{,}1 \cdot (0{,}1 - 0{,}9) = \underline{0{,}82}$$
$$y_3 = y_2 + h \cdot f(x_2, y_2) = y_2 + h \cdot (x_2 - y_2) = 0{,}82 + 0{,}1 \cdot (0{,}2 - 0{,}82) = \underline{0{,}758}$$
$$y_4 = y_3 + h \cdot f(x_3, y_3) = y_3 + h \cdot (x_3 - y_3) = 0{,}758 + 0{,}1 \cdot (0{,}3 - 0{,}758) = \underline{0{,}7122}$$
$$y_5 = y_4 + h \cdot f(x_4, y_4) = y_4 + h \cdot (x_4 - y_4) = 0{,}7122 + 0{,}1 \cdot (0{,}4 - 0{,}7122) = \underline{0{,}68098}$$
$$y_6 = y_5 + h \cdot f(x_5, y_5) = y_5 + h \cdot (x_5 - y_5) = 0{,}68098 + 0{,}1 \cdot (0{,}5 - 0{,}68098) = \underline{0{,}662882}$$

Výsledky výpočtů v jednotlivých krocích jsou zobrazeny v tabulce níže. Výsledky jsou srovnány s přesným analytickým řešením. Chyba výpočtu e_n vyjadřuje rozdíl mezi přesnou hodnotou a hodnotou vypočtenou pomocí Eulerovy metody.

x_n	přesné $y(x_n)$	$h = 0,2$		$h = 0,1$	
		y_n	e_n	y_n	e_n
0	1,000	1,000	0,000	1,000	0,000
0,1	0,910			0,900	0,010
0,2	0,837	0,800	0,037	0,820	0,017
0,3	0,782			0,758	0,024
0,4	0,741	0,680	0,061	0,712	0,029
0,5	0,713			0,681	0,032
0,6	0,698	0,624	0,074	0,663	0,035

Tabulka 1 – Výsledky výpočtu Eulerovou metodou



Obrázek 1 – Grafické znázornění výsledků výpočtu Eulerovou metodou

Na obrázku 1 lze vypozorovat následující vlastnosti chyby *e*

- chyba *e* je úměrná kroku *h*
- chyba *e* s rostoucím *x* vzrůstá

Modifikace Eulerovy metody

Eulerova metoda je velice jednoduchá, ale bohužel velice nepřesná. Hlavní nepřesnost je způsobena tím, že při kroku z x_n do x_{n+1} používáme na celém intervalu konstantní hodnotu derivace $f(x_n, y_n)$. Tato hodnota se však správně v průběhu kroku mění. Tuto chybu se snaží odstranit modifikace Eulerovy metody.

První modifikace se snaží vystihnout změnu derivace tím, že místo derivace na začátku intervalu použije derivaci uprostřed intervalu. Doprostřed intervalu se dostaneme obyčejnou Eulerovou metodou s poloviční délkou kroku. Výsledný vzorece je

$$y_{n+1} = y_n + h \cdot f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}h \cdot f(x_n, y_n)) \tag{2.9}$$

Tento vztah lze přepsat do tvaru

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}h \cdot k_1) \\ y_{n+1} &= y_n + h \cdot k_1 \end{aligned} \tag{2.10}$$

Druhá modifikace používá průměr z derivace na začátku a na konci intervalu.

$$y_{n+1} = y_n + \frac{1}{2}h \cdot [f(x_n, y_n) + f(x_n + h, y_n + h \cdot f(x_n, y_n))] \tag{2.11}$$

Tento vztah lze přepsat do tvaru

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + h, y_n + h \cdot k_1) \\ y_{n+1} &= y_n + h \cdot \frac{k_1 + k_2}{2} \end{aligned} \tag{2.12}$$

[2] [3] [5]

2.2.2 Rungovy-Kuttovy metody

Dalšími modifikacemi lze Eulerovu metodu dále vylepšovat. Získáme tak tzv. *Rungovy-Kuttovy metody*. Při těchto metodách získáme několik odhadů derivace k_i v s různých bodech. Pro celý krok potom použijeme vážený průměr těchto derivací s váhami w_i tj.

$$y_{n+1} = y_n + h \cdot (w_1 k_1 + ... + w_s k_s) \tag{2.13}$$

Odhady derivací se vypočítají následujícími vztahy

$$\begin{aligned} k_1 &= f(x, y) \\ k_2 &= f(x + \alpha_2 h, y + h\beta_{21} k_1) \\ k_3 &= f(x + \alpha_3 h, y + h\beta_{31} k_1 + h\beta_{32} k_2) \\ k_i &= f(x + \alpha_i h, y + h\sum_{j=1}^{i-1} \beta_{ij} k_j) \end{aligned} \tag{2.14}$$

Pokud použijeme jen jeden bod (tj. $s = 1$), dostaneme metodu prvního řádu – Eulerovu metodu. Pokud použijeme dva body můžeme dostat modifikované Eulerovy metody. První modifikaci odpovídá volba

$$w_1 = 0 \qquad w_2 = 1 \qquad \alpha_2 = \beta_{21} = \frac{1}{2} \tag{2.15}$$

druhé modifikaci odpovídá

$$w_1 = \frac{1}{2} \qquad w_2 = \frac{1}{2} \qquad \alpha_2 = \beta_{21} = 1 \tag{2.16}$$

S rostoucím počtem bodů dostáváme metody vyšších řádů. Například při čtyřech odhadech derivace můžeme dostat následující metodu čtvrtého řádu

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1) \\ k_3 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2) \\ k_4 &= f(x_n + h, y_n + hk_3) \\ y_{n+1} &= y_n + h(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}) \end{aligned} \tag{2.17}$$

U dosud uvedených Rungových-Kuttových metod se řád metody vždy rovnal použitému počtu odhadů derivace. Tak to ale vždy není, například při pěti odhadech derivace se dá sestrojit pouze metoda čtvrtého řádu. I z toho důvodu patří uvedená metoda čtvrtého řádu (2.17) k velice oblíbeným. [2] [3] [5]

2.3 Mnohokrokové metody

2.3.1 Metoda středního bodu

Metoda středního bodu je poměrně jednoduchá metoda druhého řádu. První kroky této metody se provedou jednoduchou Eulerovou metodou.

$$y_1 = y_0 + h \cdot f(x_0, y_0) \tag{2.18}$$

Další kroky vycházejí s předposledního bodu x_{n-1} za použití derivace z posledního bodu x_n.

$$y_{n+1} = y_{n-1} + h \cdot f(x_n, y_n) \tag{2.19}$$

Během jednoho kroku tedy používáme derivaci uprostřed intervalu, odtud název metody.

K výpočtu nové hodnoty y_{n+1} potřebujeme znát dvě předchozí hodnoty y_n a y_{n-1}. Tato metoda již tedy není jednokroková, nýbrž dvoukroková. [2]

2.3.2 Mnohokrokové metody

Mnohokrokové metody využívají k výpočtu nové hodnoty řešení znalost řešení ve více předchozích bodech. Obecný tvar *k*-krokové metody je

$$\sum_{i=0}^{k} \alpha_i y_{n+i} = h \sum_{i=0}^{k} \beta_i f_{n+i} \qquad n = 0,1,... \tag{2.20}$$

kde α_i a β_i jsou vhodně zvolené koeficienty. Například

$$y_{n+1} = y_n + \frac{1}{2} h(3 f_n - f_{n-1}) \tag{2.21}$$

$$y_{n+1} = y_n + \frac{1}{12} h(23 f_n - 16 f_{n-1} + 5 f_{n-2}) \tag{2.22}$$

$$y_{n+1} = y_n + \frac{1}{2} h(f_{n+1} + f_n) \tag{2.23}$$

$$y_{n+1} = y_n + \frac{1}{12} h(5 f_{n+1} + 8 f_n - f_{n-1}) \tag{2.24}$$

$$y_{n+2} = y_n + 2h f_{n+1} \tag{2.25}$$

$$y_{n+2} = y_n + \frac{1}{3}h(f_{n+2} + 4f_{n+1} + f_n) \tag{2.26}$$

$$y_{n+1} = \frac{4}{3}y_n - \frac{1}{3}y_{n-1} + \frac{2}{3}hf_{n+1} \tag{2.27}$$

$$y_{n+1} = \frac{18}{11}y_n - \frac{9}{11}y_{n-1} + \frac{2}{11}y_{n-2} + \frac{6}{11}hf_{n+1} \tag{2.28}$$

Metody (2.22) a (2.28) jsou tříkrokové. Pokud k výpočtu řešení v novém bodě není třeba znát v tomto bodě derivaci, nazýváme metodu *explicitní*. V opačném případě se jedná o metodu *implicitní*. Například metoda (2.23) je metoda implicitní, protože k výpočtu y_{n+1} je třeba znát $f_{n+1} = f(x_{n+1}, y_{n+1})$. Z předchozích příkladů jsou implicitní metody (2.23), (2.24), (2.26), (2.27), (2.28). [2]

2.4 Metody prediktor-korektor

Mnohokrokové metody se většinou používají způsobem, který se nazývá prediktor-korektor. Pracuje se vždy s dvojicí metod, kdy jedna je explicitní a druhá implicitní. Nejprve se pomocí explicitní metody (tzv. prediktoru) získá odhad nového řešení. Tento odhad se použije jako start iteračního procesu, kterým se řeší implicitní metoda (tzv. korektor). Tímto iteračním procesem se první odhad z prediktoru zpřesní. Například

Prediktor: $$y_{n+1}^P = y_n + h \cdot f(x_n, y_n) \tag{2.29}$$

$$f_{n+1}^P = f(x_{n+1}, y_{n+1}^P) \tag{2.30}$$

Korektor: $$y_{n+1}^K = y_n + \frac{1}{2}h(f_{n+1}^P + f_n) \tag{2.31}$$

První odhad řešení y_{n+1}^P se vypočítá prediktorem (2.29), v daném případě obyčejnou Eulerovou metodou. V tomto bodě se vypočítá hodnota derivace (2.30). Odhad řešení se zpřesní korektorem (2.31). Výpočty (2.30) a (2.31) se opakují dokud nedosáhneme požadované přesnosti řešení korektoru, přičemž od $n = 2$ vstupuje do vzorce (2.30) zpřesněná hodnota y_{n+1}^K a následně v korektoru se pak počítá hodnota y_{n+1}^{K+1}, čili

$$\begin{aligned} f_{n+1}^P &= f(x_{n+1}, y_{n+1}^K) \\ y_{n+1}^{K+1} &= y_n + \frac{1}{2}h(f_{n+1}^P + f_n) \end{aligned} \tag{2.32}$$

Dvojice prediktor a korektor je vhodné volit tak, aby řád prediktoru byl roven řádu korektoru (pro zachování přesnosti).

Vícekrokové metody jsou většinou ryhlejší než metody jednokrokové. Mají však také několik nevýhod. Jednou z nevýhod je jejich obtížné startování. Než začneme vícekrokovou metodu používat, je třeba znát řešení v několika bodech. Počáteční podmínka nám však udává řešení pouze v jednom bodě. Potřebná řešení v dalších bodech je tedy třeba získat vhodnou jednokrokovou metodou. [2] [5]

2.5 Řešení diferenciálních rovnic vyšších řádů

Jak již bylo zmíněno výše, řešení diferenciálních rovnic druhého a vyšších řádů se převádí na řešení soustavy diferenciálních rovnic řádu prvního. Ukažme si nyní, jak by vypadal upravený vztah Eulerovy metody pro řešení diferenciální rovnice druhého řádu.

Řešení spočívá v zavedení substituce

$$y'(x) = z(x) \tag{2.33}$$

Obecný rekurentní vztah Eulerovy metody (2.8) pak bude mít tvar

$$z_{n+1} = z_n + h \cdot f(x_n, y_n, z_n) \tag{2.34}$$

$$y_{n+1} = y_n + h \cdot z_n \tag{2.35}$$

Pro řešení diferenciální rovnice druhého řádu jsou nutné počáteční podmínky

$$y(x_0) = y_0 \tag{2.36}$$

$$y'(x_0) = y'_0 \tag{2.37}$$

Zavedenou substitucí tak získáváme počáteční podmínku

$$z(x_0) = z_0 \tag{2.38}$$

a můžeme tak přistoupit k výpočtu hodnot $y(x_n)$ v jednotlivých bodech x_n. [2] [5]

3 ELEARNING

Součástí této diplomové práce je vytvořený online systém v podobě dynamických www stránek, který umožňuje blíže pochopit problematiku řešení diferenciálních rovnic numerickými metodami. Tento systém můžeme chápat jako eLearningovou pomůcku. Je tedy namístě eLearning představit.

3.1 eLearning

Pojem eLearning je pojmem mladým, ještě ne tolik rozšířeným a uzrálým, nicméně se nemalou rychlostí dostává do povědomí okolní společnosti. Jeho jednotná definice ještě není specifikována. eLearning je vzdělávací proces, využívající informační a komunikační technologie k tvorbě kurzů, k distribuci studijního obsahu, komunikaci mezi studenty a pedagogy a k řízení studia. Může se jednat o rozsáhlé kurzy plně distančního charakteru a propracované nástroje kolaborativního učení. Naopak ale může jít jen o doplnění prezenční výuky. [4]

3.2 Definice

Následné definice jsou citovány z [4].

1. *eLearning je výuka s využitím výpočetní techniky a internetu.* (Petr Korviny, Moodle (nejen) na OPF, OPF, 2005)
2. *eLearning je v podstatě jakékoli využívání elektronických materiálních a didaktických prostředků k efektivnímu dosažení vzdělávacího cíle s tím, že je realizován zejména/nejenom prostřednictvím počítačových sítí.* (Kamil Kopecký, Základy e-learningu, Net University s.r.o., 2005)
3. *eLearning je vzdělávací proces, využívající informační a komunikační technologie k tvorbě kursů, k distribuci studijního obsahu, komunikaci mezi studenty a pedagogy a k řízení studia.* (Jan Wagner, Nebojme se eLearningu, Česká škola, 2005)
4. *eLearning je forma vzdělávání využívající multimediální prvky - prezentace a texty s odkazy, animované sekvence, video snímky, sdílené pracovní plochy, komunikaci s lektorem a spolužáky, testy, elektronické modely procesů, atd. v systému pro řízení studia (LMS).* (Virtuální Ostravská universita, 2005)

3.3 Výhody

Většina výhod zde jmenovaných bude vztažena k porovnání s klasickou školní vyučovací hodinou, na kterou je potřeba se dostavit osobně, v daný čas, s potřebným učebním materiálem apod.

- Časová nezávaznost – danou látku si vzdělávaná osoba může prostudovat v kteroukoli denní dobu. Kdykoliv cítí únavu může učení přerušit a následně se k němu vrátit, aniž by mu utekla část výkladu vyučujícího.
- Distanc osobní přítomnosti
- V případě neznalosti termínu použitého při výkladu je možnost dohledat si jeho význam v některé z webových encyklopedií.
- Pokud to dané prostředí dovoluje, je možnost výběru různého designu. Pak má uživatel možnost uzpůsobit si vzhled prostředí tak, aby mu bylo co možná nejsympatičtější (např. různá barevná schémata prezentací či výběr z více definovaných stylů www stránek).
- Kolektivnost - možnost probírání určitého problému např. v diskusních fórech či pomocí e-mailů s lidmi, které daný problém opravdu zajímá a touží po jeho pochopení.
- Jestliže se jedná například o výuku programování, je vítanou možností kopírování ukázkových zdrojových kódů, které by si jinak uživatel musel pracně přepisovat z učebnice.
- Animované obrázky
- Doprovodný zvuk
- a mnohé další podle typu použitých prostředků k eLearningu.

3.4 Nevýhody

Nevýhody můžeme spatřit například v tom, že musíme:

- mít počítač anebo alespoň přístup k němu
- mít potřebné programové vybavení

- být dostatečně gramotní pro práci s počítačem
- mít možnost připojit se k síti Internet - pokud se jedná o online kurzy

3.5 Problémy spojené s tradičním přístupem k eLearningu

Lze nalézt řadu příčin, proč některé eLearningové projekty nepřináší očekávané výsledky. Mezi základní příčiny patří:

- Nízká míra zapamatování výuky (jak pro klasickou tak eLearningovou výuku)
- Standardní kurzy – navržené pro distribuci pro velký počet uživatelů a proto neřešící specifické potřeby jednotlivce.
- Nedostatek interakcí – s ostatními studenty. [6]

Nízká míra zapamatování výuky

Dle výzkumů Ministerstva obrany US si studenti zachovají jen velmi krátce informaci, kterou pouze slyší, uchovají si 40% informace, kterou slyší a vidí, a uchovají si 75 % informace, kterou slyší, vidí a současně si mohou vyzkoušet (ověří si vlastní aktivitou). Proto studenti, kteří se pouze pohybují kurzem, čtou a vidí informaci, si většinou mnoho obsahu nezapamatují. Rovněž pouhé střídání podávání informace s blokem otázek k zamyšlení se postupně stává stereotypem a nepřináší kýžený efekt. Velký význam pro zapamatování má totiž rovněž poutavé zpracování obsahu – musí zde být výrazná přidaná hodnota oproti možnostem tištěného zpracování, které se za současných technických podmínek čte lépe než jeho obdoba na počítači. Velký význam tedy mají v obsahu různé simulace, příběhy, hry, vhodná kombinace zvuku, obrazu a interaktivního zapojení studenta do kurzu. [6]

Standardní kurzy

Tradičně jsou eLearningové kurzy standardizovány, aby vyhověly co největšímu počtu studentů. Jsou navrženy tak, aby si jak nejzdatnější tak nejméně zdatný student z kurzu něco odnesl. To však na druhé straně dává zbytečně mnoho informací jedněm studentům a nedostatek jiným. Rovněž obsah může být pro jedny moc složitý a pro jiné příliš jednoduchý. Výsledkem jsou kurzy, které nemusí být vysoce efektivní pro konkrétní studenty, protože nemohou naplnit jejich specifické potřeby ve vzdělávání. [6]

Nedostatek interakcí

Kontakt s ostatními studenty ve třídě je klíčový element efektivního učení. Schopnost družit se s ostatními, sdílet zkušenosti a debatovat o tématu je významný prvek úspěšného učení se. Potíží eLearningu bývá někdy nedostatek možností těchto interakcí s ostatními. Tohle částečně nahrazují diskusní fóra a posílané zprávy prostřednictvím emailů. Těžko ovšem nahradí přímý kontakt. [6]

3.6 Přínos eLearningu

Přínos eLearningu může být vysoký, pokud je zvolen správný přístup. Nové pokročilé technologie kombinované se spolehlivými klasickými strategiemi umožňují, aby učení bylo adresný, individuelní, interaktivní a poutavý proces, který je integrován do každodenního života studenta. Tato integrace pak umožňuje skutečné zvyšování efektivity učení. Vždy jde o to, jaký přístup je zvolený jak ze strany tvůrců eLearningu tak ze strany jeho uživatelů. [6]

3.7 Odkaz na eLearningový kurz

Vhodným příkladem eLearningu je webová stránka na adrese www.jakpsatweb.cz, kterou píše Yuhů, Dušan Janovský, a která pojednává o tvorbě, údržbě a vylepšování webových stránek. Denně ji navštěvuje okolo deseti tisíc uživatelů. Mně samotnému velmi pomáhala při tvorbě praktické části bakalářské práce. Je velmi přehledná, dá se v ní snadno a rychle orientovat a je psána výstižně a poutavě. Takové vlastnosti by měla splňovat každá eLearningová pomůcka.

3.8 Budoucnost

Zatím se eLearning spojuje především s osobními počítači. Díky rozvoji nových kategorií výkonných komunikačních prostředků, jako jsou kapesní či osobní počítače či organizéry, ale také nová generace mobilních telefonů, které umožňují připojení k internetu, se začíná hovořit i o m-learningu – mobilním vzdělávání. Dnešní mobilní telefony mají dostatečný výkon i pro přehrávání videopořadů a není tedy důvod, aby nemohly sloužit ke vzdělávání, stejně jako slouží k přístupu k informacím na internetu. [4]

4 TECHNOLOGIE TVORBY WWW STRÁNEK

Jak již bylo napsáno, vytvořený online systém je ve formě dynamických www stránek. Proto bude v následující kapitole pojednáno o základních technologiích tvorby www stránek.

4.1 WWW

WWW je zkratkou anglického výrazu World Wide Web, který se volně překládá jako „*celosvětová pavučina*“. Je to jedna z nejrozšířenějších služeb využívaných v celosvětové počítačové síti nazývající se Internet. Jde o označení pro aplikace internetového protokolu HTTP. Je tím myšlena soustava propojených hypertextových dokumentů.

V češtině se slovo web často používá nejen pro označení celosvětové sítě dokumentů, ale také pro označení jednotlivé soustavy dokumentů dostupných na tomtéž webovém serveru nebo na téže internetové doméně nejnižšího stupně (internetové stránce).

Dokumenty umístěné na počítačových serverech jsou adresovány pomocí URL, jejíž součástí je i doména a jméno počítače. Název naprosté většiny těchto serverů začíná zkratkou www, i když je možné používat libovolné jméno vyhovující pravidlům URL.

Protokol HTTP je dnes již používán i pro přenos jiných dokumentů než jen souborů ve tvaru HTML a výraz World Wide Web se postupně stává pro laickou veřejnost synonymem pro internetové aplikace.

Služba WWW je kombinací technologií:

- URL – jednoznačná identifikace zdroje v Internetu
- HTTP – komunikační protokol (mezi klientem a serverem)
- HTML – jazyk pro formátování komplexních dokumentů
- CSS – kaskádové styly formátování HTML
- Java, JavaScript, VBScript, … – technologie pro tvorbu dynamických stránek, jejichž skripty se vykonávají na straně klienta.
- JSP, ASP, PHP, CGI, ... – technologie pro tvorbu dynamických stránek, jejichž skripty se vykonávají na straně web serveru. [4]

4.2 URL

URL je zkratka anglického výrazu Uniform Resource Locator, jehož překlad zní „*jednoznačné určení zdroje*“. Je to řetězec znaků s definovanou strukturou a je to způsob, jak jednoznačně zapsat umístění souboru na Internetu nebo na intranetu. V HTML se používá jak pro zacílení odkazů, tak pro načítání obrázků a podpůrných souborů. [4]

Jednotlivá pole v URL jsou následující

protokol, doménové jméno, port, specifikace souboru, parametry

Některá pole jsou nepovinná – buď nemají význam, nebo se předpokládá předdefinovaná hodnota, závislá např. na protokolu (např. pro HTTP je implicitní port 80), nebo na aplikaci (pro webový prohlížeč HTTP).

Jako příklad je uvedena URL adresa webové stránky www.jakpsatweb.cz, kdy jsme se přes odkazy dostali až na stránku pojednávající o URL, na které je uvedena následující tabulka

http://www.jakpsatweb.cz/html/url.htm#priklad

Část adresy	Příklad	Jiné možné hodnoty
protokol	http://	ftp://, mailto:, atd.
doména 3. úrovně (server)	www.	cokoliv.
doména 2. úrovně	jakpsatweb.	seznam., google., apod.
generická doména	cz	com, sk, gov apod.
port	nic	:80, :*číslo*
cesta (adresáře)	/html/	/, /cokoliv/adresar/
jméno souboru	url.htm	index.html apod.html
záložka	#priklad	#jménozáložky
dotaz	nic	?proměnná=hodnota

Tabulka 2 – Rozložení URL na jednotlivé části adresy

[7]

4.3 HTTP

HTTP (HyperText Transfer Protocol) je internetový protokol určený původně pro výměnu hypertextových dokumentů ve formátu HTML. Používá obvykle port TCP/80. Tento protokol je spolu s elektronickou poštou tím nejvíce používaným a zasloužil se o obrovský rozmach Internetu v posledních letech.

V současné době je používán i pro přenos dalších informací. Pomocí rozšíření MIME umí přenášet jakýkoli soubor (podobně jako e-mail), používá se společně s formátem XML pro tzv. webové služby (spouštění vzdálených aplikací) a pomocí aplikačních bran zpřístupňuje i další protokoly, jako je např. FTP nebo SMTP.

Protokol funguje způsobem dotaz-odpověď. Uživatel (pomocí programu, obvykle internetového prohlížeč) pošle serveru dotaz ve formě čistého textu, obsahujícího označení požadovaného dokumentu (URL), informace o schopnostech prohlížeče apod. Server poté odpoví pomocí několika řádků textu (hlavičky) popisujících výsledek dotazu (zda se dokument podařilo najít, jakého typu dokument je atd.), za kterými následují data samotného požadovaného dokumentu.

Pokud uživatel bude mít po chvíli další dotaz na stejný server (např. proto, že uživatel v dokumentu kliknul na hypertextový odkaz), bude se jednat o další, nezávislý dotaz a odpověď. Z hlediska serveru nelze poznat, jestli tento druhý dotaz jakkoli souvisí s předchozím. Kvůli této vlastnosti se protokolu HTTP říká *bezstavový protokol* – protokol neumí uchovávat stav komunikace, dotazy spolu nemají souvislost. Tato vlastnost je nepříjemná pro implementaci složitějších procesů přes HTTP (např. internetový obchod potřebuje uchovávat informaci o identitě zákazníka, o obsahu jeho „nákupního košíku“ apod.). K tomuto účelu byl protokol HTTP rozšířen o tzv. HTTP cookies, které umožňují serveru uchovávat si informace o stavu spojení na počítači uživatele. [4]

Příklad HTTP komunikace

Uživatel si chce například nechat zobrazit stránku

http://www.jakpsatweb.cz/server/http-protokol.html

1. Uživatel toto URL zadá do prohlížeče.
2. Prohlížeč (klient) si vyhodnotí doménu, přes DNS si zjistí, jaké IP adresy se má ptát.
3. Přes TCP protokol naváže spojení se serverem na zjištěné IP adrese. Teprve nyní začíná HTTP.
4. Prohlížeč pak pošle na server toto HTTP volání:

```
GET /server/http-protokol.html HTTP/1.1
HOST www.jakpsatweb.cz
```

5. Toto HTTP volání přijde na server, na kterém hostuje doména se jménem www.jakpsatweb.cz. Podle HOST se pozná, kterého virtuálního hostingu se bude požadavek týkat (pokud je jich na serveru více).
6. Server tedy vidí požadavek:

```
GET /server/http-protokol.html HTTP/1.1
```

7. Server se podívá do kořene dokumentů (první lomítko) do adresáře server/ a vezme soubor http-protokol.html. Vidí, že ho má zpět poslat v protokolu HTTP verze 1.1. Připojí před soubor http hlavičky odpovědi a pošle to zpět. Symbolicky vypadá odpověď takto:

```
Stavová odpověď
hlavička 1
hlavička 2 atd.
prázdný řádek
samotný html dokument
```

tedy konkrétně například takto:

```
HTTP/1.1 200 OK
Content-type: text/html
Date: Sun, 21 May 2006 17:10:21 GMT

<html>
<head>
<title>stránka</title>
...atd.
```

Tuto odpověď dostane klient (většinou tedy prohlížeč). Z odpovědi napřed odstraní všechny HTTP hlavičky, čili vše před prvním prázdným řádkem. Pak prohlížeč samotný HTML dokument zpracuje = zobrazí čtenáři. [7]

4.4 HTML

HTML je zkratka z anglického výrazu HyperText Markup Language, který překládáme jako „*značkovací jazyk pro hypertext*“. Je jedním z jazyků pro vytváření stránek v systému World Wide Web, který umožňuje publikaci stránek na Internetu. Jazyk HTML navrhli Tim Berners-Lee a Robert Caillau. Jazyk je podmnožinou dříve vyvinutého rozsáhlého univerzálního značkovacího jazyka SGML (Standard Generalized Markup Language). Vývoj HTML byl ovlivněn vývojem webových prohlížečů, které zpětně ovlivňovaly definici jazyka.

Jazyk je charakterizován množinou značek a jejich atributů. Mezi značky se uzavírají části textu dokumentu a tím se určuje význam (*sémantika*) obsaženého textu. Názvy jednotlivých značek se uzavírají mezi úhlové závorky („< >“). Část dokumentu uzavřená mezi značkami tvoří tzv. element (prvek) dokumentu. Součástí obsahu elementu mohou být další vnořené elementy. Atributy jsou doplňující informace, které upřesňují vlastnosti elementu. Značky (také nazývané tagy) jsou obvykle párové. Rozlišujeme počáteční a koncové značky. Koncová značka má před názvem značky znak lomítko („</ >“). [4]

Příklady HTML tagů jsou uvedeny v následující tabulce

tag	párový	význam	příklad
b	ano	tučné písmo	slovo
i	ano	kurzíva	slůvko
sup	ano	horní index	^{slovíčko}
sub	ano	dolní index	<sub>slovíčko</sup>
p	nepovinně	odstavec	<p>začátek nového odstavce
br	ne	zalomení řádku	…konec řádku.
hr	ne	vodorovná čára	<hr>

Tabulka 3 – Příklady HTML tagů

Struktura HTML dokumentu je následující

tag	párový	význam	výskyt
html	ano	začátek HTML dokumentu	na začátku souboru
head	ano	hlavička stránky	na začátku souboru
body	ano	tělo stránky + definice pozadí	za <head>
<!-- -->	ano	poznámka	kdekoliv
!doctype	ne	specifikace DTD	úplně na začátku souboru

Tabulka 4 - Základní značky vymezující oblasti HTML souboru

HTML

Začíná a končí celý dokument. Veškerý další obsah musí být uvnitř. Jedná se o značku nepovinnou, většina prohlížečů si ji domyslí. Pokud však mají být soubory v souladu s normou, je vhodné tag používat. Tag html nemá žádné atributy.

HEAD

Hlavička dokumentu, která se nezobrazuje. Obsahuje nepovinně další tagy (title, meta, link, style, script aj.). Pokud se v hlavičce použije prostý text, v některých prohlížečích se zobrazí na začátku stránky! Tag head nemá žádné atributy.

BODY

Tělo dokumentu. Obsahuje veškerý zobrazovaný obsah stránky. Tag body obsahuje mnoho atributů, ale jejich používání se nahradilo využíváním CSS vlastností.

POZNÁMKA

Všechno, co je v HTML souboru obaleno značkami „<!--“ a „-->“, je považováno za poznámku a nezobrazuje se.

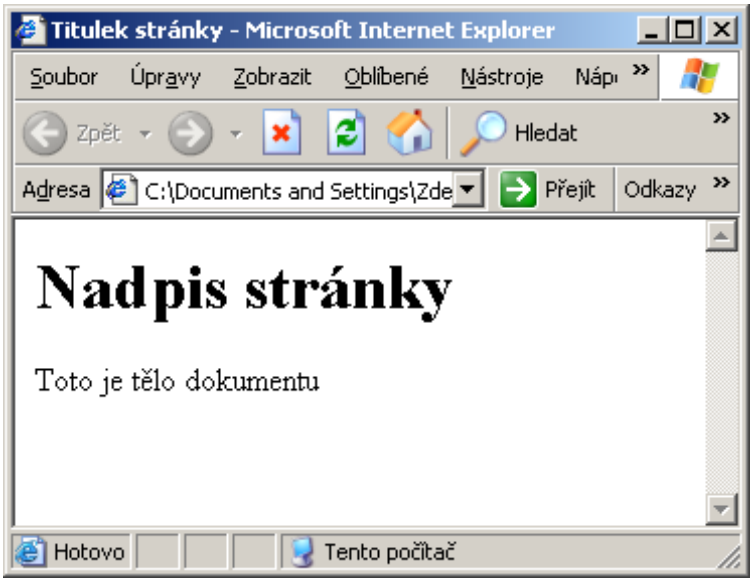
!DOCTYPE

Specifikace DTD. Píše se úplně na začátek souboru, ještě před značku <html>. Není nutné to dělat, ale podle standardu značkovacích jazyků SGML a XML je vhodné strukturovanou formou říci, že tento dokument je HTML dokument.

Příklad jednoduchého zdrojového HTML kódu je následující

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
<!-- toto je komentář -->
   <head>
      <title>Titulek stránky</title>
   </head>
<!-- tělo dokumentu -->
   <body>
      <h1>Nadpis stránky</h1>
      <p>Toto je tělo dokumentu</p>
   </body>
</html>
```

Výstupem tohoto zdrojového kódu bude stránka zobrazená na následujícím obrázku. [7]



Obrázek 2 – Ukázka www stránky

4.5 CSS

CSS je zkratka pro anglický název Cascading Style Sheets, česky „*kaskádové styly*“. Kaskádové, protože se na sebe mohou vrstvit definice stylu, ale platí jenom ta poslední. Je to kolekce metod pro grafickou úpravu webových stránek. Pomocí CSS stylů lze definovat kromě barvy, písma a velikosti spoustu dalších věcí jako např. rámeček, podtržení, tučnost, vlnitost, zobrazení, odrážky, okraje atd. Hlavním smyslem je umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu. Původně to měl umožnit už jazyk HTML, ale v důsledku nedostatečných standardů a konkurenčního boje výrobců prohlížečů se vyvinul jinak. [4] [7] [8]

Výhody CSS

- **Oddělení struktury a stylu**
- **Konzistentní styl** – na všech stránkách webové prezentace by měly být všechny nadpisy stejné úrovně, seznamy, zdůrazněné části textu apod. stejného stylu. S použitím formátovacích možností HTML je to obtížné – u každého objektu v každém dokumentu se vzhled objektu stále znovu nastavuje. S použitím CSS je to velmi jednoduché. Vytvoří se soubor stylu, který se připojuje k HTML dokumentu. Ve všech dokumentech jsou pak objekty stejného vzhledu.
- **Rozsáhlejší možnosti** - CSS nabízí rozsáhlejší formátovací možnosti než samotné HTML.
- **Dynamická práce se styly** - provést změnu stylu webu, který pro formátování vzhledu využívá jen možnosti HTML, znamená najít a nahradit všechny značky a změnit atributy mnoha dalších značek. V případě používání CSS znamená změna stylu webu přepsání jediného souboru – souboru stylů.
- **Větší kompatibilita alternativních webových prohlížečů**
- **Kratší doba načítání stránky** - soubor CSS se uloží do mezipaměti prohlížeče a pokud není změněn, tak se načítá pouze jednou a zobrazení stránek se velmi urychlí. [4]

Použití CSS je možné třemi způsoby.

1. **Přímo v textu** zdroje u formátovaného elementu.
2. **Pomocí „stylopisu“** (angl. „stylesheet“) v hlavičce stránky. Stylopis je jakýsi seznam stylů. Je v něm obecně napsáno co má být jak zformátováno, například že nadpisy mají být zelené. Do stránky se stylopis píše mezi tagy <style> a </style>.
3. **Použitím externího stylopisu** - to je soubor ***.css**, na který se stránka odkazuje tagem <link>. V souboru je umístěný stylopis. Hlavní výhoda je v tom, že na jeden takový soubor se dá nalinkovat mnoho stránek, takže pak všechny vypadají podobně. Tohle je nejčastější použití. [7]

Stylový předpis se skládá z posloupnosti pravidel. Každé pravidlo určuje vzhled některého elementu dokumentu, nebo skupiny elementů. Pravidlo začíná tzv. selektorem, který specifikuje („adresuje“) skupinu elementů. Selektor je následován seznamem deklarací, které určují vzhled vybrané skupiny elementů. Celý seznam je uzavřen ve složených závorkách a jednotlivé deklarace jsou odděleny středníkem (tj. za poslední deklarací středník už být nemusí).

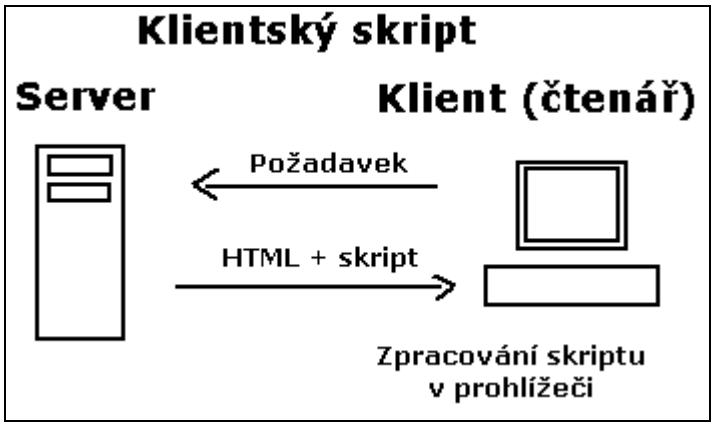
Níže je pro ilustraci uveden příklad deklarace stylu těla dokumentu, odstavce a nadpisu první úrovně. [4]

```
body {                          /* styl těla dokumentu*/
  color: red;                   /* červená barva textu*/
  background-color: white;      /* bíla barva pozadí  */
}

h1 {                            /* vzhled nadpisu první úrovně */
  margin: 5px;                  /* okraj šířky 5 pixelů        */
  font-size: 12pt               /* velikost fontu písma 12 bodů*/
}

p {                             /* styl odstavce       */
  text-align: center;           /* text centrovat      */
  line-height: 10pt;            /* výška řádku 10 bodů */
}
```

4.6 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, jehož autorem je Brendan Eich. Jeho syntaxe patří do rodiny jazyků C/C++/Java. Slovo Java je však součástí jeho názvu pouze z marketingových důvodů a s programovacím jazykem Java jej vedle názvu spojuje jen podobná syntaxe. JavaScript umožňuje např. hodnotit data ve formuláři a počítat a je základem dynamických www stránek. [4] [8]

Program v JavaScriptu se obvykle spouští až po stažení www stránky z Internetu, tzv. na straně klienta. Jedná se tedy o klientský skript. Protikladem klientských skriptů jsou skripty serverové, které jsou vykonávány na serveru a na klienta jdou už jen výsledky. Schéma fungování klientského skriptu je na obrázku pod tímto textem. Za zmínku ještě stojí, že JavaScript je tzv. case sensitivní, tzn. že záleží na velikosti písmen v zápisu. Omezením tohoto jazyka je skutečnost, že uživatel jej může nastavením svého webového prohlížeče zákazat a znemožnit tak jeho činnost. [7]



Obrázek 3 – Princip klientského skriptu

Začlenění JavaScriptu do stránky je možné třemi způsoby.

1. Pomocí tagu <script> do proudu dokumentu
2. Pomocí tagu <script> s odkazem na externí soubor s příponou *.js
3. In-line (řádkový) zápis jako atribut tagu bez použití <script> [7]

Ilustrační příklad kódu znázorňuje deklaraci funkce *soucetCisel*. Tato funkce přebírá dva parametry *a* a *b*. Uvnitř těla funkce je pak deklarována lokální proměnná *soucet*, která je nastavena na hodnotu nula. Poté už je proveden samotný součet čísel a pomocí příkazu *return* tato výsledná hodnota vrácena.

```
function soucetCisel(a,b){
  var soucet=0;
  soucet=a+b;
  return soucet;
  }
```

Tato funkce by šla ještě vylepšit například kontrolování vstupních proměnných. V případě chybného vstupu na to upozornit. Pomocí vestavěné funkce JavaScriptu *isNaN()* bychom zkontrolovali, zda proměnné *a* a *b* jsou čísla nebo byly zadány nesmyslné vstupy. Funkce *isNaN()* vrací *true*, pokud jejím argumentem není číslo a *false* v případě opačném. [7]

```
function soucetCisel(a,b){
  var soucet=0;
  if( isNaN(a)==false && isNaN(b)==false ){
    soucet=a+b;
    return soucet;
    }
   else alert(„Nesprávně zadané vstupní argumenty!“);
  }
```

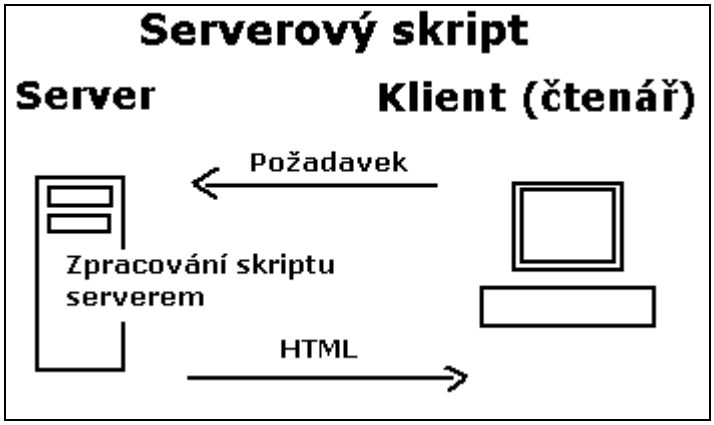
4.7 Java

Java je objektově orientovaný programovací jazyk, který vyvinula firma Sun Microsystems a představila 23. května 1995. Jeho syntaxe je založena na programovacím jazyku C++. Použity však byly i jiné jazyky, z kterých se tvůrci tohoto jazyku pokoušeli vybrat pouze to nejlepší. Slovo Java je ve skutečnosti jistý druh kafé, který si asi tvůrci zamilovali natolik, že jím pojmenovali celý jazyk. Mezi základní charakteristiky, resp. výhody tohoto jazyka patří přenositelnost mezi platformami a operačními systémy, srovnatelná s jazykem C. Java jako jazyk je tedy dostupná pro různé operační systémy Windows počínaje a Linuxem konče. [4] [10]

4.8 JSP

JSP je zkratka pro anglický název Java Server Pages. Jedná se o nástroj pro psaní dynamických HTML stránek založený na jazyku Java, funkčností velmi podobný ASP nebo PHP. Jedná se vlastně o HTML stránky, do kterých je pomocí speciálních značek vložen kód v Jave, který se provádí při vyřizování dotazu na straně web serveru.

Na rozdíl od PHP a ASP, které nemají téměř žádnou typovou kontrolu, JSP mají silnou typovou kontrolu z Javy. Druhou hlavní odlišností je kompilace stránek do tzv. servletů, což jsou speciální třídy v jazyce Java, které pak komunikují s web serverem. JSP by se tedy daly charakterizovat jako nástroj na psaní servletů. JSP také umožňují používání „sessions“, tedy možnost uchování dat konkrétního uživatele mezi více HTTP požadavky na web serveru. [9]



Obrázek 4 – Princip serverového skriptu

Soubory s JSP se ukládají s příponou *.jsp. Vkládání programového kódu Javy se provádí mezi značky „<%“ a „%>“. Souborům využívajícím češtinu je nejprve úplně na začátku dokumentu potřeba nastavit správné kódování.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8" %>
```

Za první znak „%“ se může vkládat další znak, který má svůj speciální význam.

```
<%!   /* V této řádce se definují proměnné             */  %>
<%=   /* V této řádce se tisknou proměnné na obrazovku */  %>
<%@   /* V této řádce se vkládají externí scripty      */  %>
<%    /* V této řádce se píše kód                      */  %>
```

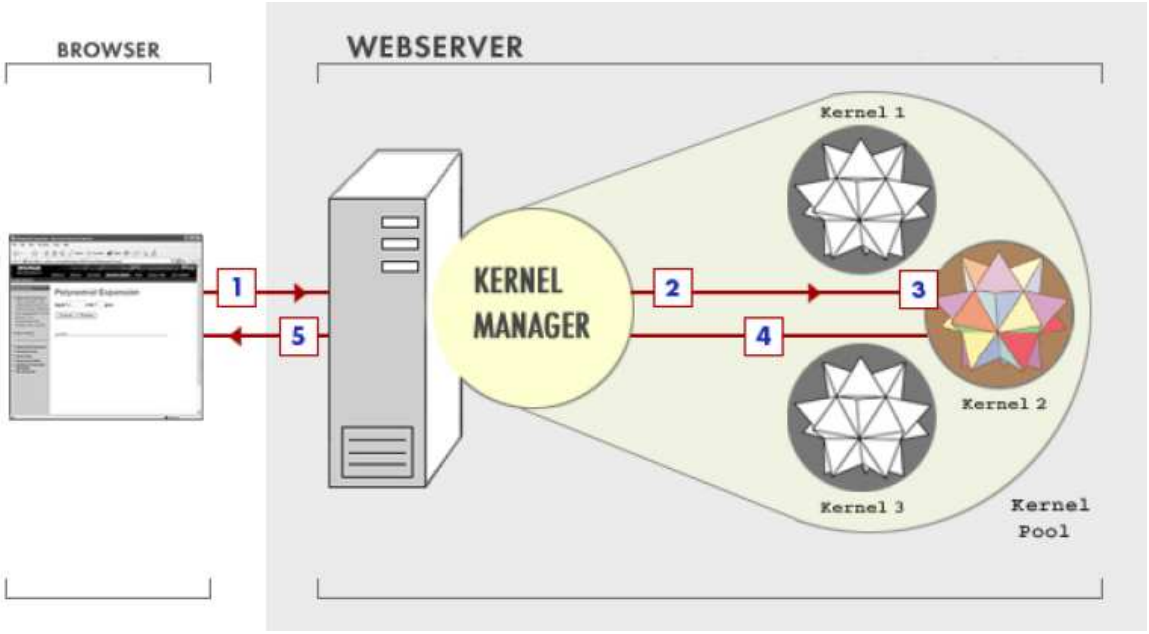
Často používanou technikou je dělení stránek na menší části a vkládání stránek na více místech v aplikaci. K tomuto účelu slouží speciální direktiva. Výskyt této značky se při kompilaci nahradí JSP kódem vkládané stránky.

```
<%@ include file=“napriklad/externi_soubor.jsp“ %>
```

[9]

4.9 webMathematica

WebMathematica představuje jednoduchý způsob, jak vytvářet interaktivní výpočty na webu. Tato unikátní technologie umožňuje vytvářet webové stránky, kde jsou využity výpočetní a vizualizační schopnosti softwaru Mathematica v celé jeho šíři. Každému uživateli stačí standardní internetový prohlížeč (browser) pro zadávání výpočtů i vizualizaci výsledků. A právě použití internetového prohlížeče představuje jednu z velkých výhod této technologie, neboť prostředí prohlížeče je známé převážné většině uživatelů. Nemusí se tak učit programovat přímo v Mathematice, ale jsou pro ně programátory vytvořeny www stránky s intuitivním ovládáním a zadáváním nejrůznějších matematických výpočtů. Navíc není potřeba mít samotný software Mathematica nainstalován, neboť o veškeré výpočty se stará server. Z programátorského hlediska zase můžeme spatřovat výhodu v tom, že tuto technologii je možné kombinovat s dalšími webovými technologiemi. Tak můžeme vytvářet různá uživatelská prostředí vhodná pro daný typ výpočtu. [11]



Obrázek 5 – Schéma technologie JSP

WebMathematica je založena na technologii JSP. Princip je následující:

1. Webový prohlížeč posílá požadavek na webMathematica server
2. WebMathematica server si rezervuje Mathematica kernel (jádro), které je nainstalované na serveru.
3. Mathematica kernel je nastaven, provede kalkulace a vrací výsledky
4. WebMathematica server vrací Mathematica kernel do „bazénu“
5. WebMathematica server vrací výsledky webovému prohlížeči [11]

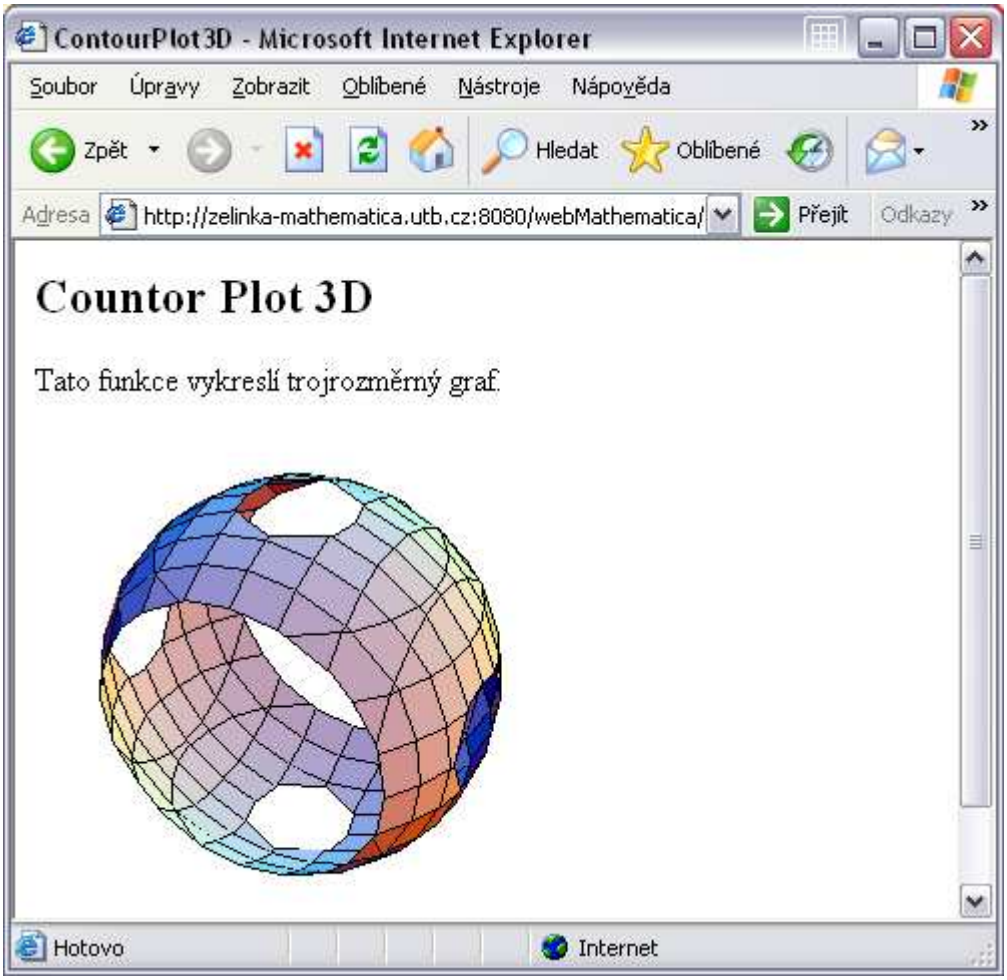
4.10 MSP

MSP je zkratka pro anglický název Mathematica Server Pages. MSP nám umožňují vytvářet statické či dynamické HTML stránky, které využívají technologii webMathematica. Statické stránky budou provádět stále stejné výpočty. V případě dynamických stránek může uživatel, např. prostřednictvím formulářových polí měnit parametry výpočtu. V praxi to znamená, že mezi standardní HTML tagy jsou vloženy speciální MSP tagy, mezi které je pak vkládán kód Mathematici. Postup je tedy následující:

1. Vytvoření HTML kódu
2. Přidání MSP tagů mezi HTML tagy
3. Přidání Mathematica kódu mezi MSP tagy

Podívejme se na ukázkovou HMTL stránku obsahující MSP skript a všimněme si zejména MSP tagů. Nejprve se provádí alokace jádra Mahtematici a až poté jsou prováděny jednotlivé výpočty. Zdrojový kód musí být uložen v souboru s příponou *.jsp do adresáře servletu na serveru, na němž je nainstalována webMathematica. Výstupní www stránka tohoto kódu je zobrazena pod ním.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>
<html>
<head>
  <title>ContourPlot3D</title>
</head>

<msp:allocateKernel>

<body>
  <h2>Countor Plot 3D</h2>
  Tato funkce vykreslí trojrozměrný graf.

  <msp:evaluate>
    Get["Graphics`ContourPlot3D`"];
  </msp:evaluate>

  <msp:evaluate>
    xmin=-3; xmax=3;
    ymin=-2; ymax=2;
    zmin=-3; zmax=3;
  </msp:evaluate>

  <msp:evaluate>
    MSPShow[
      ContourPlot3D[Sin[Sqrt[x^2+y^2 + z^2]],
      {x, xmin, xmax}, {y, ymin, ymax}, {z, zmin, zmax},
      Boxed-> False]
      ]
  </msp:evaluate>
</body>

</msp:allocateKernel>

</html>
```



Obrázek 6 – Výsledek MSP skriptu

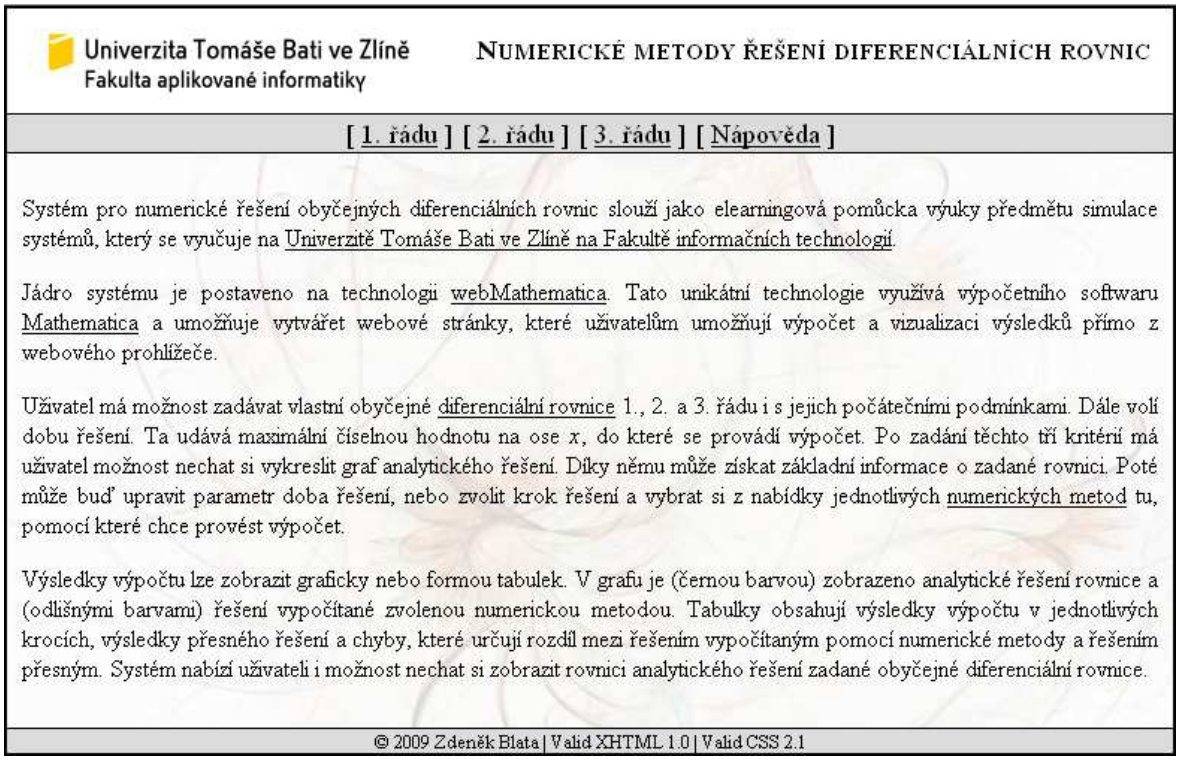
II. PRAKTICKÁ ČÁST

5 SYSTÉM PRO NUMERICKÉ ŘEŠENÍ OBYČEJNÝCH DIFERENCIÁLNÍCH ROVNIC

Součástí této diplomové práce bylo navržení a vytvoření online systému ve formě dynamických www stránek pro numerické řešení obyčejných diferenciálních rovnic, který by uživateli umožňoval vlastní zadávání diferenciální rovnice včetně počátečních podmínek a volbu mezi různými metodami výpočtu. Při realizaci systému mělo být využito prostředí webMathematica.

5.1 URL systému

http://zelinka-mathematica.utb.cz:8080/webMathematica/student/zblata/index.jsp

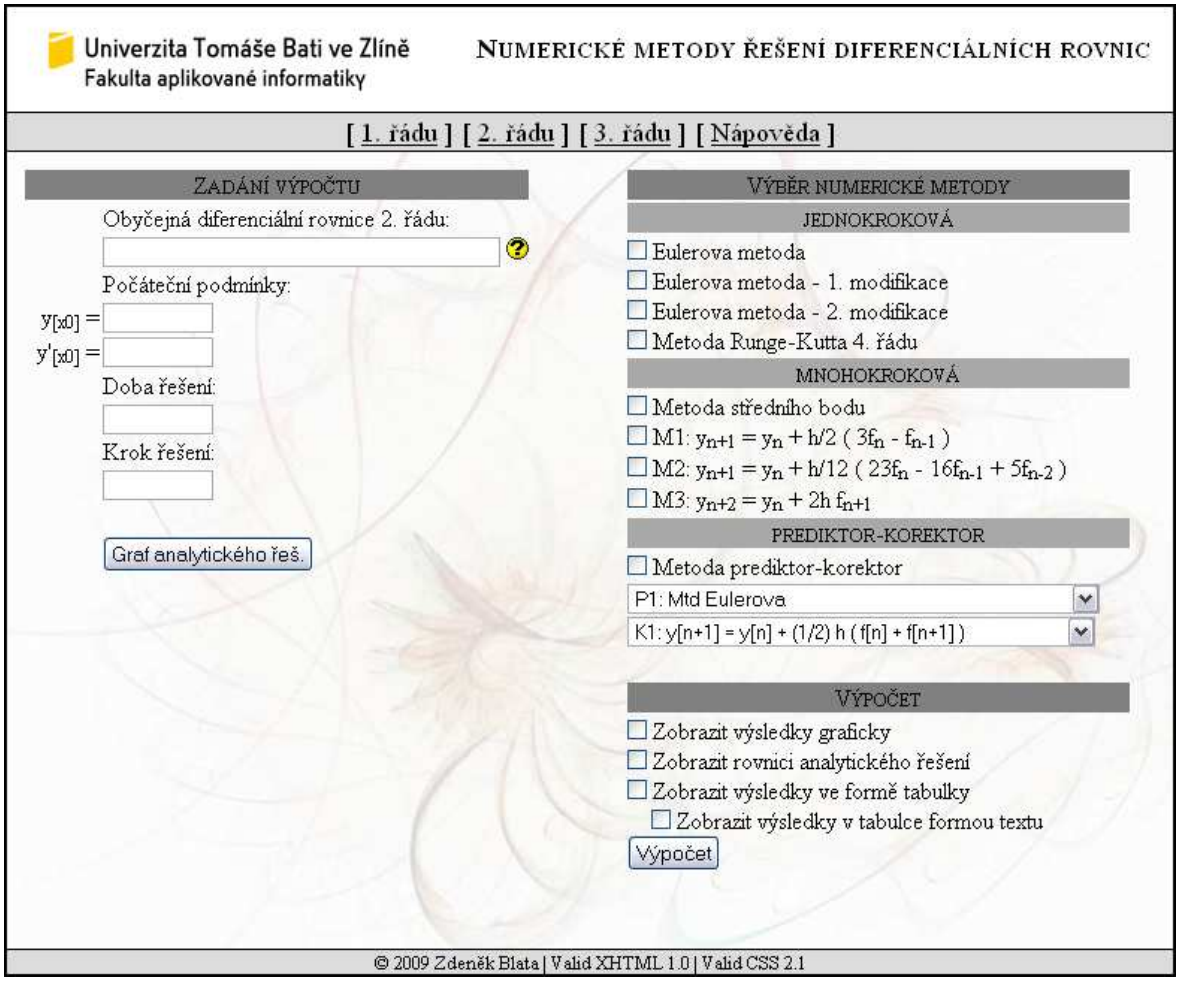


Obrázek 7 – Úvodní stránka systému

6 SYSTÉM Z POHLEDU UŽIVATELE

6.1 Struktura systému

Systém se dělí na 5 základních částí. První částí je úvodní stránka systému, která je k nahlédnutí na předcházejícím obrázku. Zde je systém obecně popsán. Přes 4 položky hlavního menu systému má pak uživatel možnost přejít do jeho dalších částí. Volí mezi odkazy na stránku s výpočtem obyčejné diferenciální rovnice prvního, druhého nebo třetího řádu a odkazem na nápovědu.



Obrázek 8 – Stránka pro řešení ODR 2. řádu

6.2 Vstupy systému

Vstupní data přijímá systém prostřednictvím klasického HTML formuláře. Formulář obsahuje vstupní pole pro zadávání vlastní diferenciální rovnice, vstupní pole pro zadávání

počátečních podmínek rovnice a pole pro zadávání doby řešení a kroku řešení. Popis jednotlivých vstupních polí je v následujících podkapitolách.

Formát (tvar) vstupů je uzpůsoben výpočetnímu softwaru Mathematica. Desetinná čísla se zapisují pomocí tečky, např. 2.7, nikoliv pomocí čárky. Formát diferenciální rovnice je popsán níže.

Obrázek 9 – Formulářový vstup systému

6.2.1 Diferenciální rovnice

Uživatel má možnost zadat si vlastní tvar diferenciální rovnice. Systém počítá pouze s proměnnými *x* a *y*, kde *y* je proměnná závislá na proměnné *x*, tzn. $f(x, y)$. Řád derivace se určuje pomocí apostrofu, tzn. řád derivace se rovná počtu apostrofů za proměnnou *y*. Rovnost se zapisuje pomocí dvou rovnítek. Příkladem vstupní obyčejné diferenciální rovnice 2. řádu může být

$$400y''+14y'+y == 3 \qquad (5.1)$$

Systém používá standardní operátory pro základní aritmetické operace jako je sčítání, odčítání, násobení a dělení, ale umí pracovat i s matematickými funkcemi jako jsou např. sinus, cosinus, apod. Jak zadávat tyto a další funkce nalezne uživatel v nápovědě systému, resp. v nápovědě softwaru Mathematica.

6.2.2 Počáteční podmínky

Počáteční podmínky určují počáteční řešení diferenciální rovnice. Jsou nezbytnou součástí zadání. Např. při výpočtu ODR 2. řádu je nutné zadat podmínky $y[x_0]$ a $y'[x_0]$.

6.2.3 Doba řešení

Doba řešení udává maximální číselnou hodnotu na ose *x*, do které se provádí výpočet. Podmínkou je zadání kladného čísla. Doba řešení se nesmí rovnat nule.

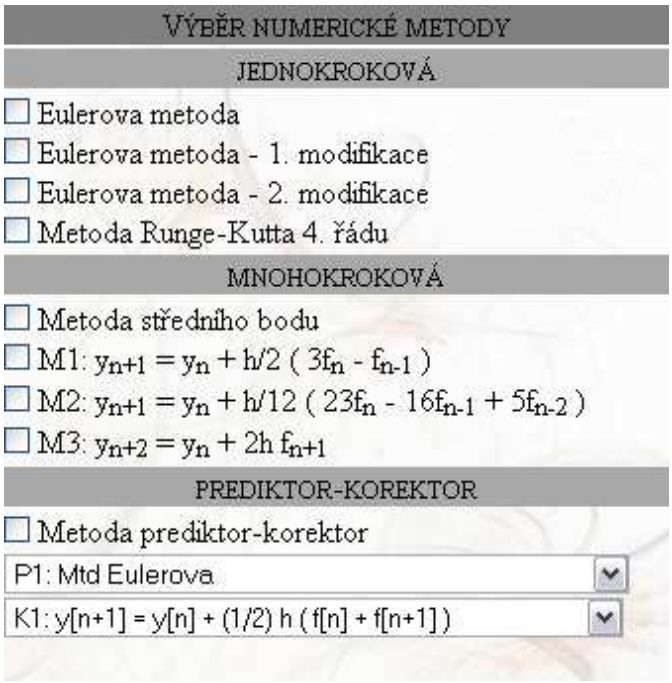
6.2.4 Krok řešení

Krok řešení je hodnota, která udává vzdálenost mezi dvěma body řešení. I zde platí podmínka, aby se krok nerovnal nule nebo zápornému číslu.

6.2.5 Volba numerické metody

Uživatel má možnost volit mezi různými metodami výpočtu. Metody jsou rozděleny na metody jednokrokové, mnohokrokové a metody prediktor-korektor. Lze provést výpočet všemi metodami najednou. Jejich výběr je možný pomocí zaškrtávacích políček.

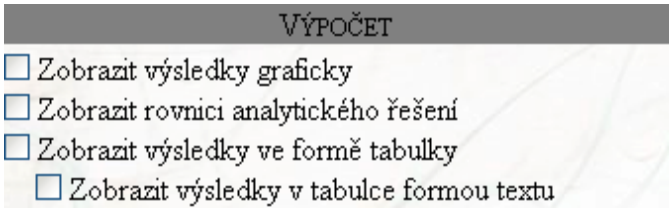
Metoda prediktor-korektor nabízí možnost výběru osmi metod pro prediktor, přičemž se jedná o všechny jednokrokové a (explicitní) mnohokrokové metody, které systém využívá, a pěti metod pro korektor. Jejich výběr je možný přes roletkové menu.



Obrázek 10 – Výběr numerické metody

6.2.6 Volba zobrazení výsledků výpočtu

Systém nabízí možnost výběru různé formy zobrazení výsledků výpočtu. Výběr je realizován pomocí zaškrtávacích políček.



Obrázek 11 – Výběr formy zobrazení výsledků

Zobrazit výsledky graficky – systém vygeneruje graf s průběhem přesného analytického řešení, který je srovnán s průběhem řešení vypočteného pomocí zvolené numerické metody. Jednotlivé průběhy jsou barevně odlišeny. Pokud je počet vypočítaných bodů menší než třicet, jsou tyto body v grafu vyznačeny.

Zobrazit rovnici analytického řešení – systém zobrazí rovnici analytického řešení.

Zobrazit výsledky ve formě tabulky – systém pro každou zvolenou metodu zvlášť zobrazí tabulku s výsledky v jednotlivých krocích výpočtu. Bližší popis se nachází v následující kapitole.

Zobrazit výsledky v tabulce formou textu – tato možnost výběru tvoří podsekci výběru předcházejícího. Systém implicitně zobrazuje výsledky tak, že je do stránky vkládá jako obrázek. Pokud by chtěl uživatel výsledná data kopírovat pro další práci s nimi, má možnost nechat si je systémem vypsat formou prostého textu. Ve výpisu obyčejného textu ovšem při větší délce výsledných dat dochází k zalamování jednotlivých řádků tabulky, což vede k nepřehlednosti. A právě proto je jako implictiní zobrazování nastaveno na formu obrázku, neboť při ní k žádnému zalamování řádků nedochází a data jsou tak přehledná a jasně čitelná.

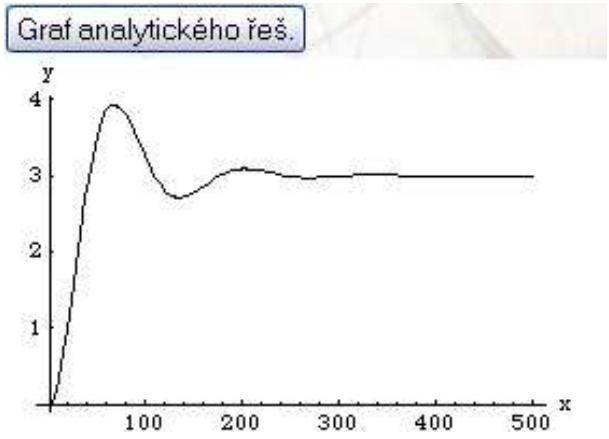
6.3 Výstupy systému

Systém obsahuje dvě tlačítka pro odeslání vstupních dat, a to tlačítko s požadavkem na vykreslení grafu analytického řešení zadané diferenciální rovnice „Graf analytického řeš.“ a tlačítko pro provedení výpočtu „Výpočet“.

Obrázek 12 – Tlačítka

6.3.1 Graf analytického řešení

K tomu, aby uživatel získal základní informace o zadané diferenciální rovnici, slouží právě tlačítko „Graf analytického řeš.“ (Pozn. název tlačítka není úplný a používá zkratku z toho důvodu, že prohlížeč Internet Explorer verze 6 zobrazoval tlačítko při jeho plném znění s deformovaným rámováním.). Pro vykreslení grafu analytického řešení je postačující vyplnění vstupních polí „Diferenciální rovnice“, „Počáteční podmínky“ a „Doba řešení“. Po odeslání dat systém vykreslí pod tímto tlačítkem kýžený graf. Poté má uživatel možnost upravovat a hledat vhodnou dobu řešení diferenciální rovnice, např. podle jejího ústalení.



Obrázek 13 – Graf analytického řešení

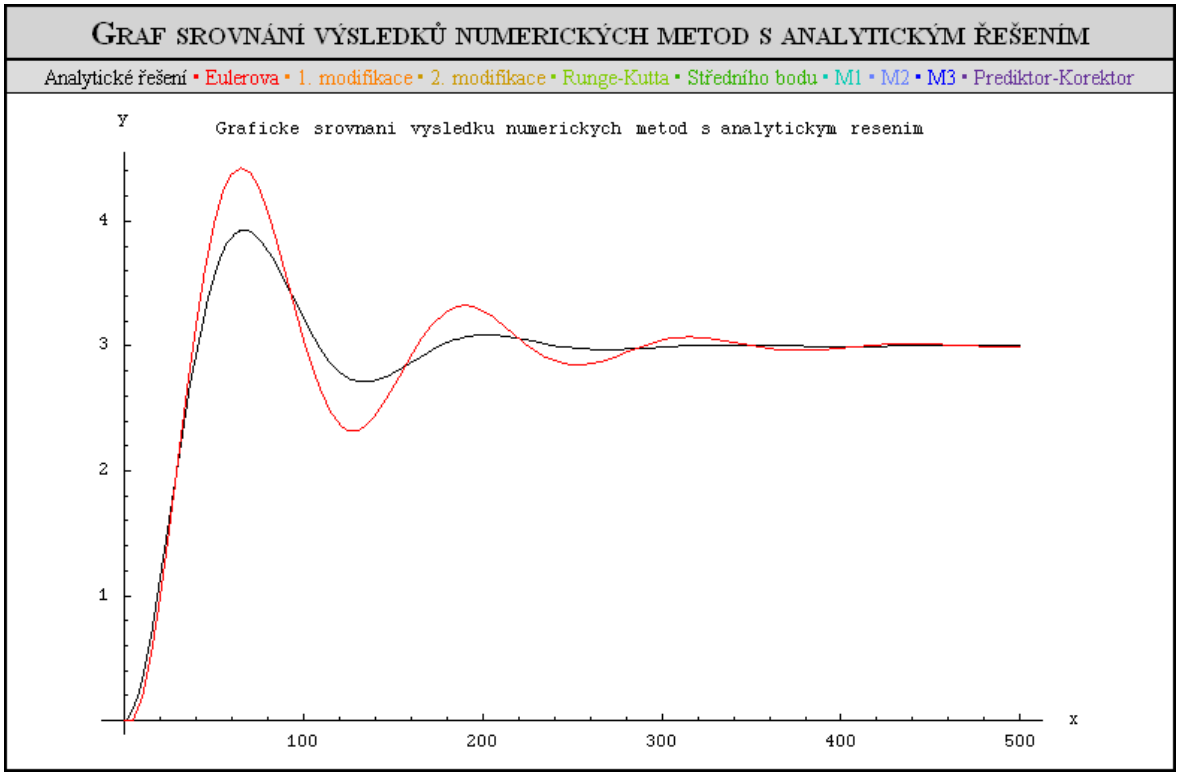
6.3.2 Výpočet

Podmínkou výpočtu je vyplnění všech formulářových polí v sekci „Zadání výpočtu“ a výběr některé z forem zobrazení výsledků v sekci „Výpočet“.

Pokud uživatel zadal volbu „Zobrazit výsledky graficky“, systém vykreslí graf ve stylu, který je vidět na obrázku 14. Legenda grafu je v jeho záhlaví. Patrné je barevné odlišení průběhů získaných pomocí jednotlivých metod. V tomto případě je černě vykreslen průběh analytického řešení a červeně průběh získaný pomocí výpočtu Eulerovou metodou.

Na obrázku 15 je ukázka zobrazené rovnice analytického řešení a na obrázku 16 je zobrazena tabulka, která obsahuje prvních několik výpočtů Eulerovou metodou níže popsaného ukázkového zadání.

Ukázkové řešení je vypočítáno pro diferenciální rovnici (5.1), jejíž obě počáteční podmínky jsou rovny nule. Doba řešení je nastavena na hodnotu 500 a velikost kroku je 5.



Obrázek 14 – Výstup systému – graf

ANALYTICKÉ ŘEŠENÍ

$$y = \frac{1}{39} e^{-7x/400}\left(-117\cos\left(\frac{3\sqrt{39}\,x}{400}\right) + 117\, e^{7x/400} - 7\sqrt{39}\sin\left(\frac{3\sqrt{39}\,x}{400}\right)\right)$$

Obrázek 15 – Výstup systému – analytické řešení

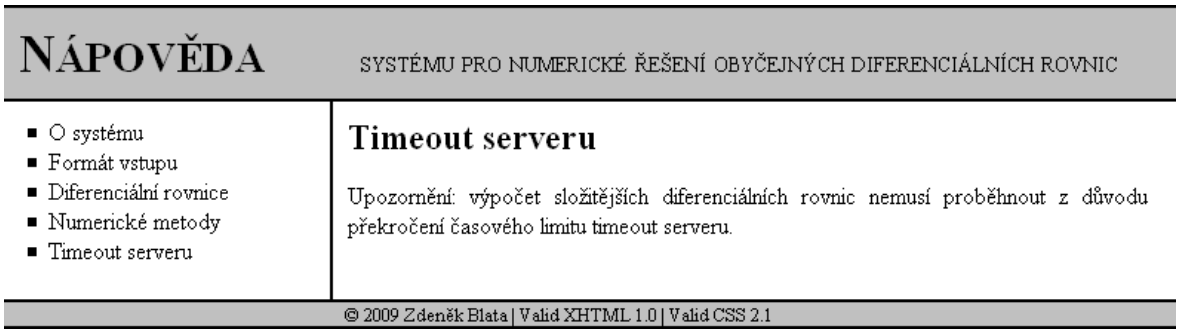
EULEROVA METODA

	x	y[x]	y[x](Presne)	e[x](Chyba)
1	0.	0.	0.	0.
2	5.	0.	0.0880578	0.0880578
3	10.	0.1875	0.328073	0.140573
4	15.	0.529688	0.681958	0.152271
5	20.	0.987773	1.11125	0.123476
6	25.	1.52009	1.57935	0.0592636
7	30.	2.08501	2.05334	0.031676
8	35.	2.64357	2.50522	0.138347
9	40.	3.16157	2.91278	0.248782
10	45.	3.61119	3.25989	0.351301
11	50.	3.97203	3.53644	0.435587

Obrázek 16 – Výstup systému – tabulka hodnot

6.4 Nápověda systému

Systém obsahuje krátkou nápovědu. Tu má uživatel možnost spustit kliknutím na odkaz „Nápověda“ v hlavním menu.



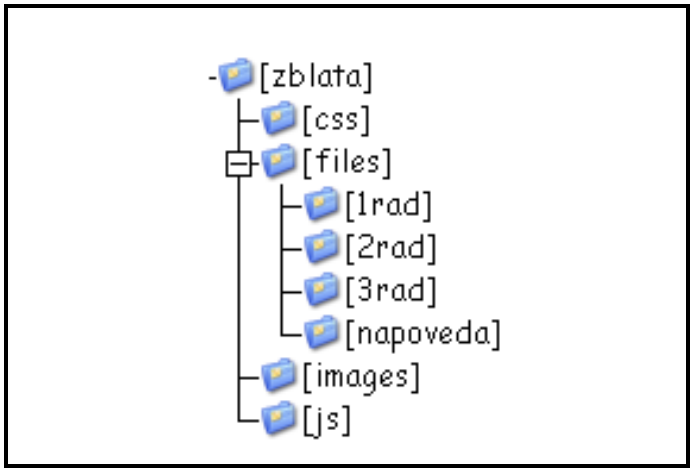
Obrázek 17 – Nápověda systému

Nápověda obsahuje následující položky

- O systému – odkaz popisující systém. Stejný popis je i na úvodní stránce systému.
- Formát vstupu – odkaz popisuje uživateli v jakém tvaru se zadávají vstupy. Především pak v jakém tvaru se zadává diferenciální rovnice.
- Diferenciální rovnice – odkaz popisuje základní teorii diferenciálních rovnic. Co jsou to diferenciální rovnice, jaké existují typy, řešení a je nastíněno jejich využití.
- Numerické metody – odkaz pojednává o numerických metodách. Jaký je důvod použití numerických metod při hledání řešení diferenciálních rovnic, jaký je princip numerických metod a jaké je jejich dělení. Je zde také popsáno na jakou přesnost je nastaven korektor systému a následně jsou vypsány vzorce všech numerických metod, pomocí nichž systém může provádět výpočet.
- Timeout serveru – odkaz upozorňuje na to, že výpočty složitějších diferenciálních rovnic nemusí proběhnout, neboť čas výpočtu přesáhne nastavený timeout serveru.

7 SYSTÉM Z POHLEDU PROGRAMÁTORA

Vytvořený systém je kombinací technologií HTML, CSS, JavaScript, JSP a MSP. Z důvodu jejich použití je o nich podrobněji pojednáno v kapitole 4. Jádro systému je postaveno na technologii webMathematica, pomocí které jsou prováděny veškeré výpočty systému. Adresářová struktura systému je patrná z následujícího obrázku.



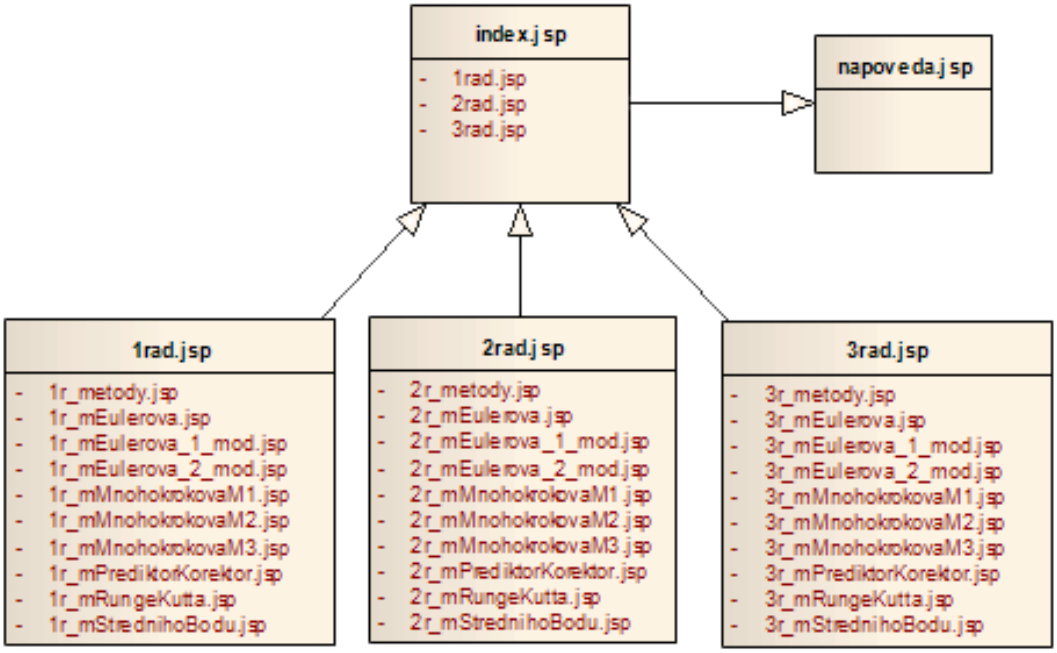
Obrázek 18 – Adresářová struktura systému

V kořenovém adresáři je umístěn soubor *index.jsp*. Jedná se o hlavní soubor celého systému. Na tento soubor je linkován soubor s externím stylopisem www stránky, který je umístěn v adresáři *css* a uložen jako *style.css*. Do hlavní stránky jsou dále podle potřeby includovány (vkládány) soubory z adresáře *files*. Veškeré obrázky, které systém využívá, jsou uloženy v adresáři *images*. Adresář *js* obsahuje externí skript jazyka JavaScript, ve kterém jsou nadefinovány funkce pro kontrolu vstupních dat systému.

Adresář files obsahuje podadresáře *1rad, 2rad, 3rad* a *napoveda*. Podadresáře, jejichž název končí na *rad*, obsahují MSP skripty s definicemi funkcí jednotlivých numerických metod a skripty s výpočty jednotlivých metod. Podadresář *napoveda* obsahuje soubory nápovědy systému.

Includování souborů do hlavního souboru *index.jsp* je naznačeno na následujícím obrázku. Vždy jsou includovány jen soubory, které jsou aktuálně potřebné k výpočtu. Vezněme si příklad, že uživatel zadal výpočet diferenciální rovnice 2. řádu pomocí Eulerovi metody a metody Runge-Kutta. V takovém případě bude do hlavního souboru vložen soubor *2rad.jsp* a do tohoto souboru budou vloženy soubory *2r_metody.jsp*, *2r_mEulerova.jsp* a

2r_mRungeKutta.jsp. Obrázek dále říká, že z hlavního souboru je možné přejít do souboru *napoveda.jsp*.



Obrázek 19 – Includování souborů

7.1 Postup činnosti systému při výpočtu

Kontrola vstupních dat

Systém čeká na stisk některého z odesílacích tlačítek formuláře. Pokud je některé tlačítko stisknuto, volá nejprve funkci JavaScriptu, která kontroluje validitu vstupních dat. Tato funkce kontroluje, zda byla zadána všechna potřebná data nutná pro výpočet, ať už se jedná o vykreslení grafu analytického řešení či o samotný výpočet numerickou metodou. Pokud byla zadána data v nesprávném tvaru nebo rozsahu, nebo pokud uživatel nezvolil žádnou formu zobrazení výsleku výpočtu, informuje o tom systém uživatele zobrazením upozorňující zprávy.

Obrázek 20 – Zpráva systému

Převzetí dat z formuláře

Po prvním zkontrolování dat jsou tyto teprve odeslána JSP, potažmo MSP skriptu, kde data z formulářového pole pro zádávní diferenciální rovnice procházejí druhou kontrolou. Zde už systém v kódu Mathematici kontroluje, zda byla diferenciální rovnice zadána syntakticky správně pro tento jazyk. Pokud ano, přiřadí data z proměnné webového formuláře proměnné Mathematici. Pokud ne, nastaví příznak proměnné *syntaxeOk* na hodnotu *false* a informuje uživatele o neuskutečněném výpočtu. Popsaný postup kontroly syntaxe a přiřazování dat je v následujícím zdrojovém kódu.

```
<msp:evaluate>
  If[ SyntaxQ[$$dr]==True, dr=ToExpression[$$dr,TraditionalForm];
      syntaxeOk=True,  syntaxeOk=False ];
  yDerPoc=ToExpression[$$yDerPoc,TraditionalForm];
  yPoc=ToExpression[$$yPoc,TraditionalForm];
  dobaReseni=ToExpression[$$dobaReseni,TraditionalForm];
  krok=ToExpression[$$krok,TraditionalForm];
  prediktor=ToExpression[$$prediktor,TraditionalForm];
  korektor=ToExpression[$$korektor,TraditionalForm];
</msp:evaluate>
```

Úprava diferenciální rovnice

V další části programu je zadaná diferenciální rovnice převedana na tvar, který je nutný pro vstup funkce Mathematici *Dsolve[]*, která počítá analytické řešení rovnice. V tomto kroku jsou také vytvořeny funkce, které počítají hodnotu derivace v daném kroku řešení a to jak numericky, tak analyticky. Programový kód popsaného postupu je pro řešení diferenciální rovnice 2. řádu vypsán níže.

```
<msp:evaluate>
  pom=Solve[ dr, y'' ];
  fceNum=pom[[1]][[1]][[2]];
  difRov=ReplaceAll[ dr, {y''->y''[x], y'->y'[x], y->y[x]} ];
  pom=DSolve[ { difRov, y[0]==yPoc, y'[0]==yDerPoc }, y, x ];
  fceAna=pom[[1]][[1]][[2]][[2]];
  mFceAna[x0_]:=( Return[ ReplaceAll[ fceAna, x->x0 ] ]//N; );
  mFceNum[x0_,y0_,z0_]:=( Return[ ReplaceAll[ fceNum,
                          {x->x0,y->y0,y'->z0} ] ]//N; );
</msp:evaluate>
```

Výpočet

Se zkontrolovanými daty a převedením diferenicální rovnice do tvaru, který i Mathematica chápe jako diferenciální rovnici, je zahájen samotný výpočet podle zvolené metody a parametrů výpočtu.

Každý řád obsahuje soubor **r_metody.jsp*, v němž jsou nadefinovány funkce, které provádí výpočet podle zvolené numerické metody v jednotlivých krocích výpočtu. Tyto funkce jsou pak volány ze skriptů jednotlivých metod, ve kterých je v cyklu prováděn výpočet hodnot v jednotlivých krocích a jejich následné ukládání do tabulek, tzv. *Table*. Jako úkazka je uvedena metoda Runge-Kutta 4. řádu.

```
<msp:evaluate>

mRungeKutta[krok_,x0_,y0_,z0_]:=Module[ {y1,z1},
    k1y=z0;
    k1z=mFceNum[x0,y0,z0];

    k2y=z0+(krok/2)*k1z;
    k2z=mFceNum[x0+krok/2,y0+(krok/2)*k1y,z0+(krok/2)*k1z];

    k3y=z0+(krok/2)*k2z;
    k3z=mFceNum[x0+krok/2,y0+(krok/2)*k2y,z0+(krok/2)*k2z];

    k4y=z0+krok*k3z;
    k4z=mFceNum[x0+krok,y0+krok*k3y,z0+krok*k3z];

    y1=y0+krok*( k1y/6 + k2y/3 + k3y/3 + k4y/6 );
    z1=z0+krok*( k1z/6 + k2z/3 + k3z/3 + k4z/6 );

    Return[{y1,z1}];
    ];

</msp:evaluate>
```

Výstup

V konečné fázi pak systém podle zvolené formy zobrazení výsledků vrací tyto uživateli. Po vykreslení grafu srovnání průběhu analytického řešení s řešením nalezeným pomocí numerických metod, zobrazení rovnice analytického řešení či vypsání hodnot výpočtů do stránky, je výpočet systému ukončen.

ZÁVĚR

Cílem mé diplomové práce bylo vytvořit systém řešení diferenciálních rovnic numerickými metodami. Výsledný systém splňuje zadané podmínky a uživatel tak má možnost zadávat si vlastní tvar diferenciální rovnice, včetně počátečních podmínek. Dále stanovuje dobu řešení a krok řešení výpočtu. Na výběr má mezi několika metodami výpočtu. Systém umožňuje grafické a tabulkové srovnání jednotlivých metod, včetně srovnání s analytickým řešením.

Při vytváření systému jsem se seznámil s technologií webMathematica. Díky této technologii mi byla usnadněna spousta programátorské práce, neboť jsem mohl využívat všech vestavěných funkcí matematického softwaru Mathematica, který je pro řešení matematických problémů ideální. Nevýhodu této technologie spatřuji snad jen v tom, že, oproti samotné Mathematice, webMathetica programátora žádným způsobem neinformuje o chybách vyskytujících se v kódu.

Systém by měl sloužit jako eLearningová pomůcka předmětu Simulace systémů, který se vyučuje na Univerzitě Tomáše Bati ve Zlíně.

ZÁVĚR V ANGLIČTINĚ

The goal of my thesis was to develop a system of solutions of differential equations numerical methods. The resulting system meets the specified conditions and the user has the option to enter their own form of differential equations, including initial conditions. In addition, provides a solution and step calculation. The choice is between several methods of calculation. The system allows graphic and tabular comparison of various methods, including comparisons with analytical solutions.

When creating the system I was acquainted with webMathematica technology. With this technology, I facilitate a lot of programming work, since I could use all the features of functions mathematical software Mathematica, which is the ideal solution to mathematical problems. However, this technology sees to that webMathetica programmer does not in any way for errors occurring in the code.

The system should serve as the eLearning simulation tool of the subject, which is taught at the Thomas Bata University in Zlín.

SEZNAM POUŽITÉ LITERATURY

[1] BULISOVÁ, Jiřina, et al. *Ottova všeobecná encyklopedie : ve dvou svazcích*. 1. vyd. Praha : Ottovo nakladatelství, s. r. o., 2003. 2 sv. (736, 752 s.). ISBN 80-7181-959-X.

[2] VICHER, Miroslav. *Numerická matematika*. Ústí nad Labem, 2003. 85 s. Skripta. ISBN 80-7044-516-5.

[3] ČERNÁ, Růžena, MACHALICKÝ, Miroslav, VOGEL, Jiří, ZLATNÍK, Čeněk. *Základy numerické matematiky a programování*. Praha, 1987. 448 s. ISBN 04-003-87.

[4] *Wikipedie : otevřená encyklopedie* [online]. 2002 [cit. 2009-04-21]. Dostupný z WWW: <http://www.wikipedie.cz/>.

[5] *Numerické metody pro řešení počátečních úloh pro ODR* [online]. 2003 [cit. 2009-04-21]. Dostupný z WWW: <http://www.vyuka.fai.utb.cz/>.

[6] ZÍDEK, Pavel. *Mixování tradičního přístupu s novými technikami pro zvýšení efektivity v e-Learning* [online]. [cit. 2009-05-04]. Dostupný z WWW: <http://www.e-learn.cz/soubory/blendingapproaches.pdf>

[7] *Jak psát web : o tvorbě internetových stránek* [online]. [cit. 2009-05-05]. Dostupný z WWW: <http://www.jakpsatweb.cz/>

[8] *Tvorba webu* [online]. 2003-2008 [cit. 2009-05-07]. Dostupný z WWW: <http://www.tvorba-webu.cz/>

[9] *Java Server Pages* [online]. 1999 [cit. 2009-05-07]. Dostupný z WWW: <http://atrey.karlin.mff.cuni.cz/~bim/pub/jsp/referat/referat.html>

[10] *Builder* [online]. 1997-2002 [cit. 2009-05-07]. Dostupný z WWW: <http://www.builder.cz/>

[11] *Ladislav Prskavec – články přednášky a publikace* [online]. [cit. 2009-05-10]. Dostupný z WWW: <http://ladislav.prskavec.net/>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ODR	**O**byčejná **D**iferenciální **R**ovnice
PDR	**P**arciální **D**iferenciální **R**ovnice
WWW	**W**orld **W**ide **W**eb („celosvětová pavučina“)
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
URL	**U**niform **R**esource **L**ocator („jednotné označování objektů“)
HTML	**H**yper**T**ext **M**arkup **L**anguage („značkovací jazyk pro hypertext“)
CSS	**C**ascading **S**tyle **S**heets („kaskádové styly“)
JSP	**J**ava **S**erver **P**ages
ASP	**A**ctive **S**erver **P**ages
PHP	**P**HP **H**ypertext **P**reprocesor („PHP Hypertextový Preprocesor“)
CGI	**C**ommon **G**ateway **I**nterface
TCP	**T**ransmission **C**ontrol **P**rotocol
MIME	**M**ultipurpose **I**nternet **M**ail **E**xtensions
XML	e**X**tensible **M**arkup **L**anguage („rozšířitelný značkovací jazyk“)
FTP	**F**ile **T**ransfer **P**rotocol
SMTP	**S**imple **M**ail **T**ransfer **P**rotocol
MSP	**M**athematica **S**erver **P**ages
DNS	**D**omain **N**ame **S**ystem

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: ZDROJOVÝ KÓD NAPROGRAMOVANÝCH NUMERICKÝCH METOD PRO ŘEŠENÍ ODR 3. ŘÁDU

```
<msp:evaluate>

mEulerova[krok_,x0_,y0_,z0_,a0_]:=Module[ {y1,z1,a1},
  a1=a0+krok*mFceNum[x0,y0,z0,a0];
  z1=z0+krok*a0;
  y1=y0+krok*z0;
  Return[{y1,z1,a1}];
  ];

mEulerova1mod[krok_,x0_,y0_,z0_,a0_]:=Module[ {y1,z1,a1},
  k1a=mFceNum[x0,y0,z0,a0];
  k1z=a0;
  k1y=z0;

  k2a=mFceNum[x0+(krok/2),y0+(krok/2)*k1y,z0+(krok/2)*k1z,a0+(krok/2)
    *k1a];
  k2z=a0+(krok/2)*k1a;
  k2y=z0+(krok/2)*k1z;

  a1=a0+krok*k2a;
  z1=z0+krok*k2z;
  y1=y0+krok*k2y;

  Return[{y1,z1,a1}];
  ];

mEulerova2mod[krok_,x0_,y0_,z0_,a0_]:=Module[ {y1,z1,a1},
  k1a=mFceNum[x0,y0,z0,a0];
  k1z=a0;
  k1y=z0;

  k2a=mFceNum[x0+krok,y0+krok*k1y,z0+krok*k1z,a0+krok*k1a];
  k2z=a0+krok*k1a;
  k2y=z0+krok*k1z;

  a1=a0+(krok/2)*(k1a + k2a);
  z1=z0+(krok/2)*(k1z + k2z);
  y1=y0+(krok/2)*(k1y + k2y);

  Return[{y1,z1,a1}];
  ];

mRungeKutta[krok_,x0_,y0_,z0_,a0_]:=Module[ {y1,z1,a1},
  k1a=mFceNum[x0,y0,z0,a0];
  k1z=a0;
  k1y=z0;

  k2a=mFceNum[x0+krok/2,y0+(krok/2)*k1y,z0+(krok/2)*k1z,a0+(krok/2)
    *k1a];
  k2z=a0+(krok/2)*k1a;
  k2y=z0+(krok/2)*k1z;

  k3a=mFceNum[x0+krok/2,y0+(krok/2)*k2y,z0+(krok/2)*k2z,a0+(krok/2)
    *k2a];
  k3z=a0+(krok/2)*k2a;
  k3y=z0+(krok/2)*k2z;

  k4a=mFceNum[x0+krok,y0+krok*k3y,z0+krok*k3z,a0+krok*k3a];
  k4z=a0+krok*k3a;
  k4y=z0+krok*k3z;
```

```
  a1=a0+krok*( k1a/6 + k2a/3 + k3a/3 + k4a/6 );
  z1=z0+krok*( k1z/6 + k2z/3 + k3z/3 + k4z/6 );
  y1=y0+krok*( k1y/6 + k2y/3 + k3y/3 + k4y/6 );

  Return[{y1,z1,a1}];
  ];

mStrednihoBodu[krok_,y0_,z0_,a0_,x1_,y1_,z1_,a1_]:=Module[ {y2,z2,a2},
  a2=a0+krok*mFceNum[x1,y1,z1,a1];
  z2=z0+krok*a1;
  y2=y0+krok*z1;
  Return[{y2,z2,a2}];
  ];

mMnohokrokovaM1[krok_,x0_,y0_,z0_,a0_,x1_,y1_,z1_,a1_]:=Module[
  {y2,z2,a2},
  a2=a1+(krok/2)*(3*mFceNum[x1,y1,z1,a1]-mFceNum[x0,y0,z0,a0]);
  z2=z1+(krok/2)*(3*a1-a0);
  y2=y1+(krok/2)*(3*z1-z0);
  Return[{y2,z2,a2}];
  ];

mMnohokrokovaM2[krok_,x0_,y0_,z0_,a0_,x1_,y1_,z1_,a1_,x2_,y2_,z2_,a2_]:=
  Module[ {y3,z3,a3},
  a3=a2+(krok/12)*(23*mFceNum[x2,y2,z2,a2]-
    16*mFceNum[x1,y1,z1,a1]+5*mFceNum[x0,y0,z0,a0]);
  z3=z2+(krok/12)*(23*a2-16*a1+5*a0);
  y3=y2+(krok/12)*(23*z2-16*z1+5*z0);
  Return[{y3,z3,a3}];
  ];

mMnohokrokovaM3[krok_,y0_,z0_,a0_,x1_,y1_,z1_,a1_]:=Module[ {y2,z2,a2},
  a2=a0+2*krok*mFceNum[x1,y1,z1,a1];
  z2=z0+2*krok*a1;
  y2=y0+2*krok*z1;
  Return[{y2,z2,a2}];
  ];

mKorektorK1[krok_,x0_,y0_,z0_,a0_,x1_,y1P_,z1P_,a1P_]:=Module[
  {y1K,z1K,a1K},
  a1K=a0+(krok/2)*(mFceNum[x0,y0,z0,a0]+mFceNum[x1,y1P,z1P,a1P]);
  z1K=z0+(krok/2)*(a0+a1K);
  y1K=y0+(krok/2)*(z0+z1K);
  Return[{y1K,z1K,a1K}];
  ];

mKorektorK2[krok_,x0_,y0_,z0_,a0_,x1_,y1_,z1_,a1_,x2_,y2P_,z2P_,a2P_]:=
  Module[ {y2K,z2K,a2K},
  a2K=a1+(krok/12)*(5*mFceNum[x2,y2P,z2P,a2P]+8*mFceNum[x1,y1,z1,a1]-
    mFceNum[x0,y0,z0,a0]);
  z2K=z1+(krok/12)*(5*a2K+8*a1-a0);
  y2K=y1+(krok/12)*(5*z2K+8*z1-z0);
  Return[{y2K,z2K,a2K}];
  ];

mKorektorK3[krok_,x0_,y0_,z0_,a0_,x1_,y1_,z1_,a1_,x2_,y2P_,z2P_,a2P_]:=
  Module[ {y2K,z2K,a2K},
  a2K=a0+(krok/3)*(mFceNum[x2,y2P,z2P,a2P]+4*mFceNum[x1,y1,z1,a1]+
    mFceNum[x0,y0,z0,a0]);
  z2K=z0+(krok/3)*(a2K+4*a1+a0);
  y2K=y0+(krok/3)*(z2K+4*z1+z0);
  Return[{y2K,z2K,a2K}];
  ];
```

```
mKorektorK4[krok_,y0_,z0_,a0_,y1_,z1_,a1_,x2_,y2P_,z2P_,a2P_]:=
  Module[ {y2K,z2K,a2K},
  a2K=(4/3)*a1-(1/3)*a0+(2/3)*krok*mFceNum[x2,y2P,z2P,a2P];
  z2K=(4/3)*z1-(1/3)*z0+(2/3)*krok*a2K;
  y2K=(4/3)*y1-(1/3)*y0+(2/3)*krok*z2K;
  Return[{y2K,z2K,a2K}];
  ];

mKorektorK5[krok_,y0_,z0_,a0_,y1_,z1_,a1_,y2_,z2_,a2_,x3_,y3P_,
  z3P_,a3P_]:=Module[ {y3K,z3K,a3K},
  a3K=(18*a2-9*a1+2a0+6*krok*mFceNum[x3,y3P,z3P,a3P])/11;
  z3K=(18*z2-9*z1+2z0+6*krok*a3K)/11;
  y3K=(18*y2-9*y1+2y0+6*krok*z3K)/11;
  Return[{y3K,z3K,a3K}];
  ];

</msp:evaluate>
```

PŘÍLOHA P II: ZDROJOVÝ KÓD SKRIPTU PROVÁDĚJÍCÍHO VÝPOČET METODOU RUNGE-KUTTA ODR 1. ŘÁDU

```
<msp:evaluate>

If[ vseOk==True,

  (*** deklarace tabulky pro ukladání hodnot výpočtu ***)
  tabHodnot=Table[ 0,{i,1,pocetKroku},{j,1,4} ];

  (*** vložení počátečních podmínek do tabulky hodnot ***)
  tabHodnot[[1,1]]=x0//N;
  tabHodnot[[1,2]]=y0//N;
  tabHodnot[[1,3]]=mFceAna[x0]//N;
  tabHodnot[[1,4]]=Abs[ tabHodnot[[1,2]]-tabHodnot[[1,3]] ]//N;

  i=2;
  (*** výpočet v jednotlivých krocích ***)
  While[ i<=pocetKroku,
    tabHodnot[[i,1]]=tabHodnot[[i-1,1]]+krok//N;
    tabHodnot[[i,2]]=mRungeKutta[krok,tabHodnot[[i-1,1]],
      tabHodnot[[i-1,2]]]//N;
    tabHodnot[[i,3]]=mFceAna[tabHodnot[[i,1]]]//N;
    tabHodnot[[i,4]]=Abs[tabHodnot[[i,2]]-tabHodnot[[i,3]]]//N;
    i++;
    ];

  (*** vytvoření grafu a naformátování tabulky hodnot ***)
  tabHodnotGrafu = Table[0, {x, pocetKroku}, {y, 2}];
  For[i=1, i<=pocetKroku, tabHodnotGrafu[[i,1]]=tabHodnot[[i,1]];
    i++];
  For[i=1, i<=pocetKroku, tabHodnotGrafu[[i,2]]=tabHodnot[[i,2]];
    i++];
  graf1=ListPlot[ tabHodnotGrafu, PlotJoined -> False,
    PlotStyle->RGBColor[0.5, 0.8, 0], PlotStyle->PointSize[0.5],
    ImageSize->700 ];
  graf2=ListPlot[ tabHodnotGrafu, PlotJoined -> True,
    PlotStyle->RGBColor[0.5, 0.8, 0], ImageSize->700 ];
  tabHodnot=TableForm[ tabHodnot, TableHeadings->{Automatic,
    {"x","y[x]","y[x](Presne)","e[x](Chyba)"}} ];

  (*** přidání grafu a tabulky do globalních seznamů ***)
  If[ pocetKroku<=30, listGrafu=Append[ listGrafu, graf1 ] ];
  listGrafu=Append[ listGrafu, graf2 ];
  listTabulek[[4]]=tabHodnot;

  (*** smazání některých proměnných ***)
  Clear[ tabHodnot, graf1, graf2 ];

  ];

</msp:evaluate>
```