Matematika v programovacích jazycích

Pavla Kabelíková

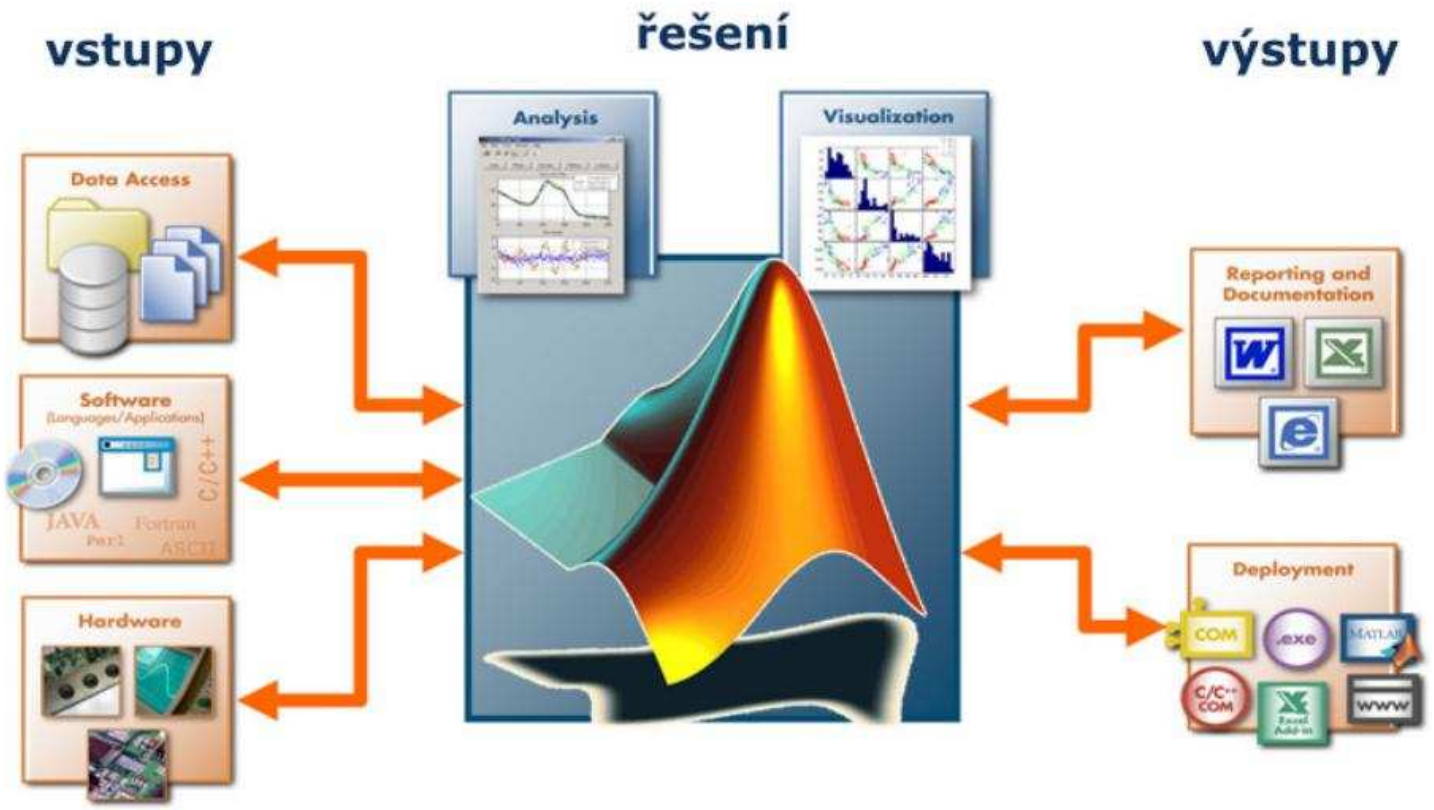
am.vsb.cz/kabelikova

pavla.kabelikova@vsb.cz

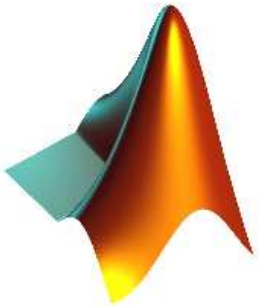
Úvodní diskuze

- **Otázky:**
  - Jaké programovací jazyky znáte?
  - S jakými programovacími jazyky jste již pracovali?
  - Který programovací jazyk je podle vás nejvhodnější pro matematické úlohy a proč?
  - Pod jakým operačním systémem nejčastěji programujete?

MATLAB

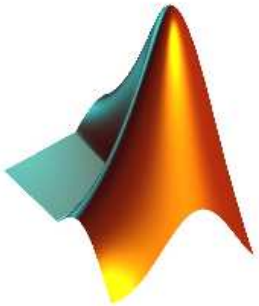

vstupy
řešení
výstupy
Data Access
Software
(Languages/Applications)
C/C++
JAVA
Fortran
Perl
ADCTI
Hardware
Analysis
Visualization
Reporting and
Documentation
Deployment
COM
.exe
MATLAB
C/C++
COM
Excel
Add-in
www

MATLAB



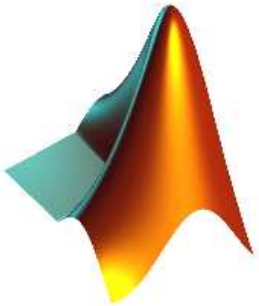
- **MATLAB®** je integrované prostředí pro vědeckotechnické výpočty, modelování, návrhy algoritmů, simulace, analýzu a prezentaci dat, paralelní výpočty, měření a zpracování signálů, návrhy řídicích a komunikačních systémů.
- MATLAB poskytuje grafické a výpočetní nástroje, rozsáhlé specializované knihovny funkcí spolu s výkonným programovacím jazykem čtvrté generace.
- Jazyk MATLABu je jednodušší než například Fortran nebo C.
- Rychlé výpočetní jádro s optimálními algoritmy.
- MATLAB je implementován na všech významných platformách (Windows, Linux, Solaris, Mac).

MATLAB



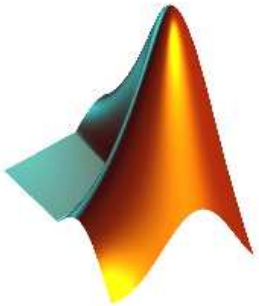
- **Výpočetní jádro:**
  - operace s maticemi reálných a komplexních čísel,
  - vícerozměrná pole reálných nebo komplexních čísel, pole buněk, ve kterých každý prvek může být jiného typu,
  - operace s polynomy,
  - časové řady nebo signály a funkce pro jejich analýzu,
  - formát uložení řídkých matic,
  - práce s objekty, definování libovolných funkcí a operátorů,
  - implementováno s využitím knihoven LAPACK a ARPACK, adaptivně využívá konkrétní konfiguraci uživatelského počítače (typ procesoru, cache, paměť, operační systém, ...).

MATLAB



- Uživatelské rozhraní **MATLAB Desktop**: prohlížeč adresářů a souborů, prohlížeč pracovního prostoru, okno historie příkazů, interaktivní spouštěč aplikací, editor, debugger, profiler, hypertextová nápověda a příkazové okno.
- Integrace s jinými jazyky:
  - **Java** (možnost vytvářet složitá grafická rozhraní s použitím grafických objektů Javy, velké množství volně dostupných knihoven Javy),
  - **C, Fortran** (možnost připojovat moduly napsané v jazyce C a ve Fortranu).

MATLAB: vzorový příklad

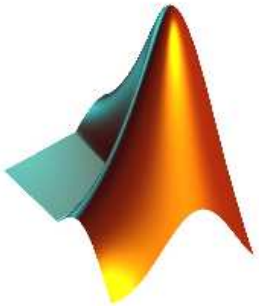


```
1   function [x, it] = newton(f, df, x0, tol)
2
3
4 -  xk = x0;
5 -  xk1 = xk - feval(f, xk)/feval(df, xk);
6 -  it = 1;
7
8 -  while (abs(xk1 - xk)>tol)
9 -      xk = xk1;
10 -     xk1 = xk - feval(f, xk)/feval(df, xk);
11 -     it = it + 1;
12
13 -  end;
14
15 -  x = xk1;
16
17
18    % Vykreslení grafu
19 -  t_x = -0.1:0.05:3.5;
20
21 -  for i=1:length(t_x)
22 -      t_y(i) = feval(f, t_x(i));
23 -  end
24 -  plot(t_x, t_y);
25 -  hold on; grid on; axis on;
26
27    % Vykreslení výsledku
28 -  plot(x, feval(f, x), 'r*');
```

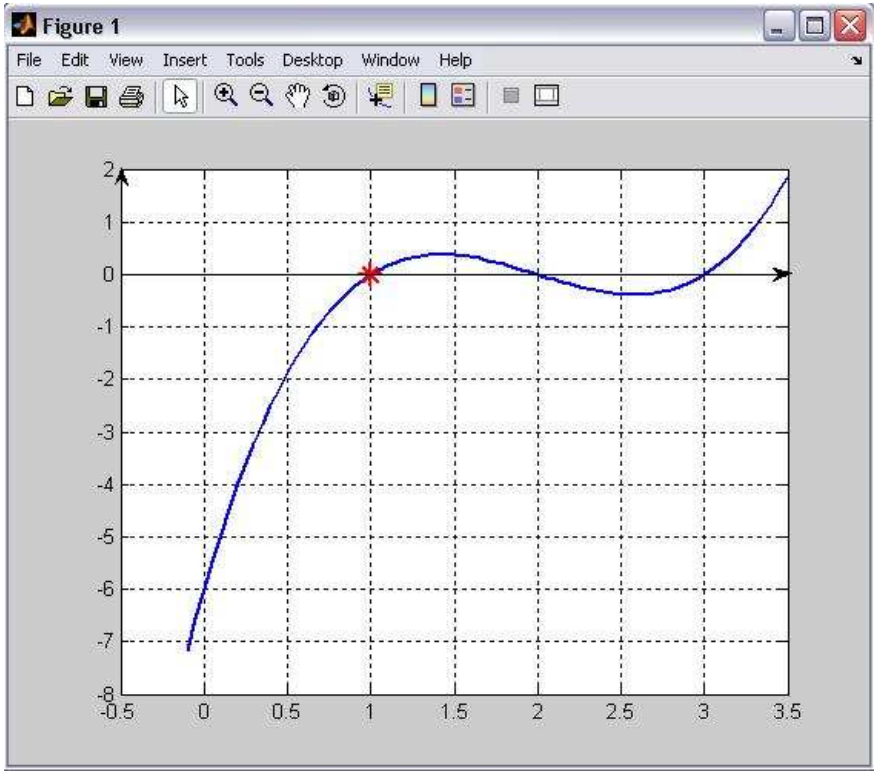
```
1   function y = pol(x)
2
3 -  y = x^3 - 6*x^2 + 11*x - 6;
```

```
1   function y = dpol(x)
2
3 -  y = 3*x^2 - 12*x + 11;
```

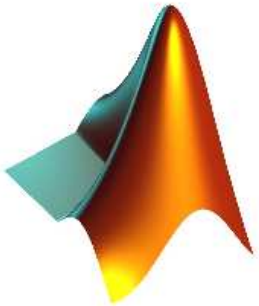
MATLAB: vzorový příklad



```
>> [x, it] = newton(@pol, @dpol, -2000, 1e-12)

x =

    1.0000


it =

    25

>>
```

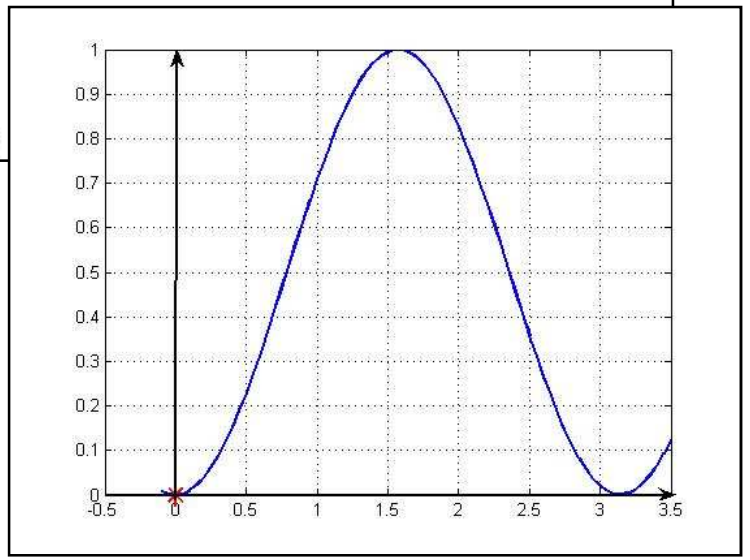


MATLAB: vzorový příklad



```
1   function [x, it] = newton(f_string, x0, tol)
2
3   %Symbolical toolbox (syms x)
4 - f = sym(f_string);
5 - df = diff(f)
6
7 - xk = x0;
8 - xk1 = xk - subs(f, xk)/subs(df, xk);
9 - it = 1;
10
11 - while (abs(xk1 - xk)>tol)
12 -     xk = xk1;
13 -     xk1 = xk - subs(f, xk)/subs(df, xk);
14 -     it = it +1;
15
16 - end;
17
18 - x = xk1;
19
20  % Vykreslení grafu
21 - t_x = -0.1:0.05:3.5;
22
23 - for i=1:length(t_x)
24 -     t_y(i) = subs(f, t_x(i));
25 - end
26 - plot(t_x, t_y);
27 - hold on; grid on; axis on;
28
29  % Vykreslení výsledku
30 - plot(x, subs(f, x), 'r*');
```

```
>> [x, it] = newton_sym('sin(x)^2', -0.5, 1e-12)

df =

2*sin(x)*cos(x)

x =

 -8.0634e-013


it =

    39
```



C/C++

- **C**
  - nízkoúrovňový, kompilovaný, relativně minimalistický,
  - inline assembler (zápisu assembleru přímo do kódu),
  - zdrojový kód C přenositelný na jiné architektury,
  - systémové programování (ovladače a jádro OS),
  - operační systémy, překladače, knihovny a interprety vysokoúrovňových jazyků implementovány v C,
  - použití i na vytváření běžných aplikací.

C/C++

- **C**
  - **Ukládání dat v C:**
    - statickou alokací paměti (při překladu),
    - automatickou alokací paměti na zásobníku,
    - dynamickou alokací na haldě (heap) pomocí knihovních funkcí.
  - S pamětí se pracuje přes datový typ zvaný ukazatel, který drží odkaz na paměťový prostor daného typu proměnné (odpovědnost programátora, aby neukazoval na paměť nealokovanou).
  - Ukazatele na funkce.

C/C++

- **C++** = „C s třídami“
  - odvozen z jazyka C, není ale úplně kompatibilní,
  - problém kompatibility: C++ definuje mnohem striktnější pravidla pro přetypování datových typů,
  - implementuje objektově orientované programování, generické a procedurální programování,
  - dědičnost (vícenásobná dědičnost) pro objekty,
  - šablony,
  - přetěžování funkcí a operátorů,
  - abstraktní datové typy.

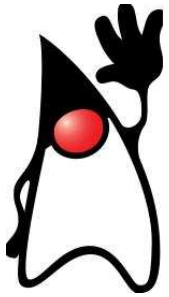
C/C++: vzorový příklad

```c
#include <stdio.h>
#include <math.h>

double feval( double (*f)(double), double x) {
    return (*f)(x);
}

double f(double x) {
    return x*x*x - 6*(x*x) + 11*x - 6;
}

double df(double x) {
    return 3*(x*x) - 12*x + 11;
}

void newton( double (*f)(double), double (*df)(double), double x0, double tol, double *res, int *it) {
    double xk = x0;
    double xk1 = xk - feval( f, xk) / feval( df, xk);
    *it = 1;

    while ( fabs(xk1 - xk) > tol ) {
        xk = xk1;
        xk1 = xk - feval(f, xk) / feval(df, xk);
        *it += 1;
    }
    double x = xk1;
    *res = x;
}
```

C/C++: vzorový příklad

```
int main() {

    printf( "Computing Newton-Raphson method\n" );

    double x0 = -2000.1;
    double tol = 1e-12;
    double res = 0.0;
    int it = 0;

    newton(f, df, x0, tol, &res, &it);

    printf( "Result %.4f, it: %d\n", res, it );

    return 0;
}
```

JAVA



- Objektově orientovaný programovací jazyk, vyvinut firmou Sun Microsystems.
- **Přenositelnost** na různé systémy
  - čipové karty (platforma JavaCard),
  - mobilní telefony (platforma Java ME),
  - aplikace pro desktopové počítače (platforma Java SE),
  - rozsáhlé distribuované systémy (platforma Java EE).

- Dne 8. května 2007 Sun uvolnil zdrojové kódy Javy (cca 2,5 miliónů řádků kódu) a Java je dále vyvíjena jako open source.

JAVA

- Jazyk JAVA
  - jednoduchý (zjednodušená verze syntaxe jazyka C/C++),
  - objektově orientovaný,
  - distribuovaný,
  - interpretovaný (Java Virtual Machine (JVM)),
  - robustní (silná typová kontrola, správa paměti),
  - bezpečný,
  - nezávislý na architektuře, přenositelný,
  - víceúlohový, dynamický.
- Pomalejší než C/C++ !

JAVA – vzorový příklad


```
package newton;

public interface Evaluable {
    public double evaluate(double x);
}

public class Function implements Evaluable
{
    public double evaluate(double x)
    {
        return x*x*x - 6*(x*x) + 11*x - 6;
    }
}

public class FunctionDerivation implements Evaluable
{
    public double evaluate(double x)
    {
        return 3*(x*x) - 12*x + 11;
    }
}
```

JAVA – vzorový příklad


```java
package newton;

public class Newton {

    Evaluable f, df;

    public Newton(Evaluable f, Evaluable df) {
        this.f = f;
        this.df = df;
    }

    Result newton(double x0, double tol) {
        double xk = x0, res = 0.0;
        int it = 1;
        double xk1 = xk - f.evaluate(xk) / df.evaluate(xk);

        while (Math.abs(xk1 - xk) > tol) {
            xk = xk1;
            xk1 = xk - f.evaluate(xk) / df.evaluate(xk);
            it += 1;
        }
        res = xk1;
        Result result = new Result(res, it);

        return result;
    }
}
```

```java
package newton;

public class Result {

    double res;
    int it;

    public Result(double res, int it) {
        this.res = res;
        this.it = it;
    }

    public String toString() {
        return "Result: " + res + ", iter: " + it;
    }
}
```

JAVA – vzorový příklad


```
package newton;

public class Main {

    public static void main(String[] args) {
        Function f = new Function();
        FunctionDerivation df = new FunctionDerivation();
        Result result;

        Newton newton = new Newton( f, df );

        System.out.println( "Computing Newton-Raphson method" );

        double x0 = -2000.1;
        double tol = 1e-12;

        result = newton.newton( x0, tol );

        System.out.println(result);
    }
}
```

Python

- interpretovaný objektově orientovaný programovací jazyk,
- vyvíjen jako open source projekt, dostupný pro Unix, Windows, Mac OS; ve většině distribucí systému Linux je Python součástí základní instalace,
- dynamický interpretovaný jazyk, zařazován mezi tzv. skriptovací jazyky,
- umožňoval tvorbu rozsáhlých, plnohodnotných aplikací (včetně GUI).

Python

- Hybridní jazyk (nebo také víceparadigmatický):
  - objektově orientované paradigma,
  - procedurální paradigma,
  - v omezené míře i funkcionální paradigma.
- Kód programu je ve srovnání s jinými jazyky krátký a dobře čitelný.
- čistota a jednoduchost syntaxe,
- produktivita z hlediska rychlosti psaní programů,
- spolupráce s ostatními jazyky: C (CPython), JAVA (Jython), .NET (IronPython).

Python

- Jazyk Pythonu:
  - proměnné = pojmenované odkazy na objekty,
  - funkce = odkazy na objekty,
  - proměnné se nedeklarují,
  - proměnné tříd mohou vznikat až za běhu,
  - silná typová kontrola za běhu aplikace.
- interaktivní režim překladače

- balíky NumPy, SymPy, SciPy, Star-P, ...

Python – vzorový příklad


```
from math import fabs

def feval(f, x):
        return f(x)

def newton(f, df, x0, tol):
        xk = x0
        xk1 = xk - feval(f, xk) / feval(df, xk)
        it = 1

        while fabs(xk1 - xk) > tol:
                xk = xk1
                xk1 = xk - feval(f, xk) / feval(df, xk)
                it += 1
        x = xk1

        return (x, it)

if __name__ == '__main__':

        print 'Computing Newton-Raphson method'
        f = lambda x: x**3 - 6*(x**2) + 11*x - 6
        df = lambda x: 3*(x**2) - 12*x + 11

        x0 = -2000.1
        tol = 1e-12
        res = newton(f, df, x0, tol)

        print 'Result', res
```

Fortran

- Imperativní programovací jazyk určený zejména pro vědecké výpočty a numerické aplikace.
- První kompilátor FORTRANu byl vyvinut v letech 1954 – 1957 pro IBM 704 týmem IBM.
- Postupně se objevilo několik standardů jazyka, jako například Fortran IV a Fortran 66, Fortran 77, Fortran 90, Fortran 95, Fortran 2000, Fortran 2003 a Fortran 2008.

Fortran – vzorový příklad

```
FUNCTION rtnewt(funcd,x1,x2,xacc)
INTEGER JMAX
REAL rtnewt,x1,x2,xacc
EXTERNAL funcd
PARAMETER (JMAX=20)                    Set to maximum number of iterations.
    Using the Newton-Raphson method, find the root of a function known to lie in the interval
    [x1, x2]. The root rtnewt will be refined until its accuracy is known within ±xacc. funcd
    is a user-supplied subroutine that returns both the function value and the first derivative
    of the function at the point x.
INTEGER j
REAL df,dx,f
rtnewt=.5*(x1+x2)                      Initial guess.
do 11 j=1,JMAX
    call funcd(rtnewt,f,df)
    dx=f/df
    rtnewt=rtnewt-dx
    if((x1-rtnewt)*(rtnewt-x2).lt.0.)
        pause 'rtnewt jumped out of brackets'
    if(abs(dx).lt.xacc) return         Convergence.
enddo 11
pause 'rtnewt exceeded maximum iterations'
END
```

Srovnání

Výpočet vlastních čísel s použitím mocninné metody (N=5500)			
Program	**CPU secs**	**Memory KB**	**Size B**
C++ GNU g++	**15,69**	**1,02**	**860,00**
Fortran Intel	16,00	1,14	568,00
Java 6 -server	16,63	11,54	514,00
Pascal Free Pascal	18,78	184,00	423,00
C GNU gcc	18,83	**444,00**	382,00
Python	**975,58**	3,26	**378,00**
PHP	1194,32	4,95	315,00
Perl	1575,29	2,60	334,00

Generování Mandelbrotovy množiny (N=6400)			
Program	**CPU secs**	**Memory KB**	**Size B**
C++ GNU g++	**3,96**	5,00	**1066,00**
Java 6 -server	7,14	10,50	665,00
C GNU gcc	8,23	**396,00**	400,00
Fortran Intel	9,11	5,37	485,00
Pascal Free Pascal	12,55	52,00	530,00
Python	**396,93**	**2,72**	**287,00**
PHP	557,34	3,22	395,00
Perl	688,27	2,15	311,00

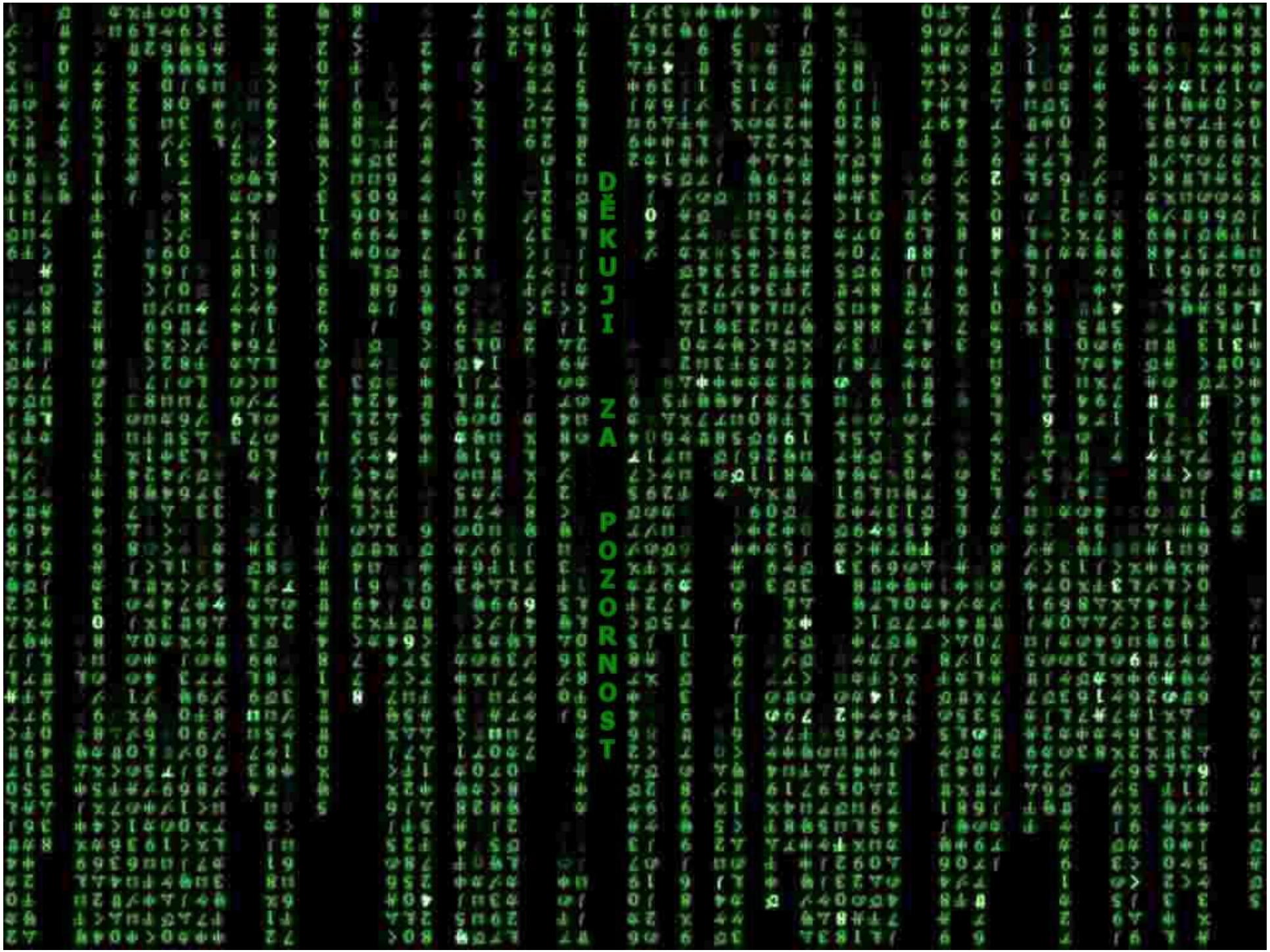
http://shootout.alioth.debian.org/

Závěrečná diskuze

- **Otázka:**
  - Který programovací jazyk je podle vás nejvhodnější pro matematické úlohy?

Programátorská pohádka

```
void pohadka() {
   if (princ.sila >= drak.sila) {
     drak.delete();
     princ += princezna + kralovstvi/2;
   } else {
     princ -= hlava;
     drak.hmotnost += princezna.hmotnost;
     kral.status = "Smutny";
     kralovstvi.goto("prdel");
   }
}
```


DĚKUJI ZA POZORNOST