Ukázky použití sw *Mathematica* v dalších oborech...

Ing. Zdeněk Buk, bukz1@fel.cvut.cz

Computational Intelligence Group
Katedra počítačů, FEL, ČVUT Praha

2007

Obsah

Použití v informatice

- Možnost využít **procedurální** (Pascal, C), **funkcionální** (Lisp) i **logický** (Prolog) styl programování.

Bohaté možnosti **vizualizace** a

- nově ve verzi 6 **interaktivní** manipulátory a grafy.

Ukázka: Programovací styly

Úkol : vypočítat druhé mocniny prvků v poli

```
pole = {1, -2, 3.14, i, a}
```

```
{1, -2, 3.14, i, a}
```

Procedurální styl

```
mocProc[vstup_List] := Module[
 {n, vystup, i},
 n = Length[vstup];
 r = Table[0, {n}];
 For[
   i = 1, i <= n,
   r[[i]] = vstup[[i]]^2;
   i++
  ];
 Return[r]
 ]
```

```
mocProc[pole]
```

$$\{1, 4, 9.8596, -1, a^2\}$$

Strukturovaný styl

```
mocStrukt[vstup_List] :=
 Table[vstup[[i]]^2, {i, Length[vstup]}]

mocStrukt[pole]
```

$$\{1, 4, 9.8596, -1, a^2\}$$

Funkcionální styl

```
mocMap[vstup_List] := Map[Function[x, x^2], vstup]

mocMap[pole]
```

$$\{1, 4, 9.8596, -1, a^2\}$$

Rekursivní styl

```
mocRek[{}] = {};
mocRek[{hodnota_, zbytek___}] :=
 Prepend[mocRek[{zbytek}], hodnota^2]
```

```
mocRek[pole]
```

$$\{1, 4, 9.8596, -1, a^2\}$$

Ukázka: Vizualizace algoritmu řazení

```
data = RandomPermutation[10]
```

{10, 7, 8, 5, 6, 2, 9, 4, 3, 1}

```
data = Reverse[Range[10]]
```

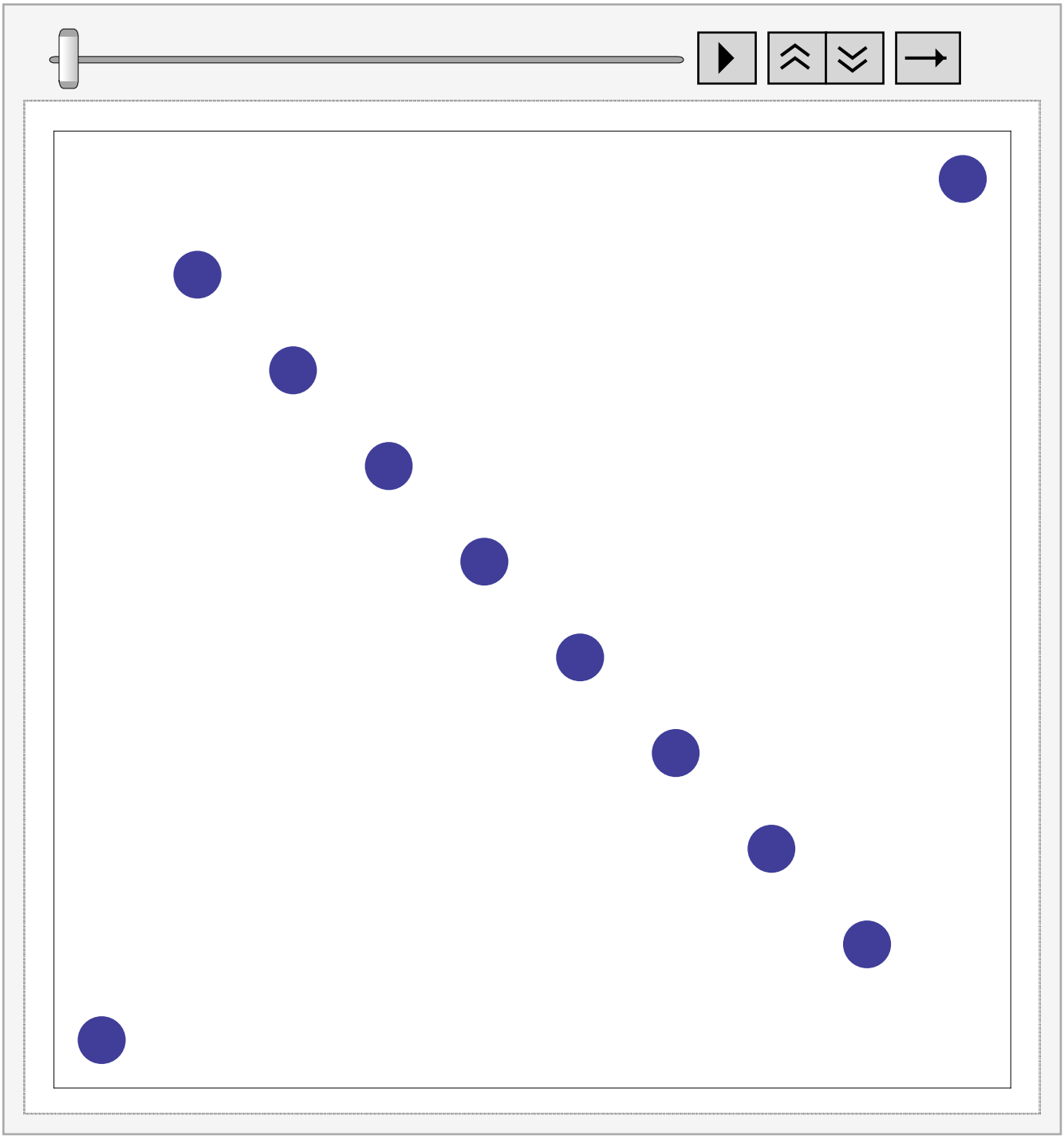
{10, 9, 8, 7, 6, 5, 4, 3, 2, 1}

... řazení opakovaným tříděním (Quick-sort)

```
lists = {};
quickSort[data]
```

{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

```
PermutationPlot[#, DisplayFunction → Identity] & /@
  lists;
ListAnimate[%, ImageSize → 300]
```

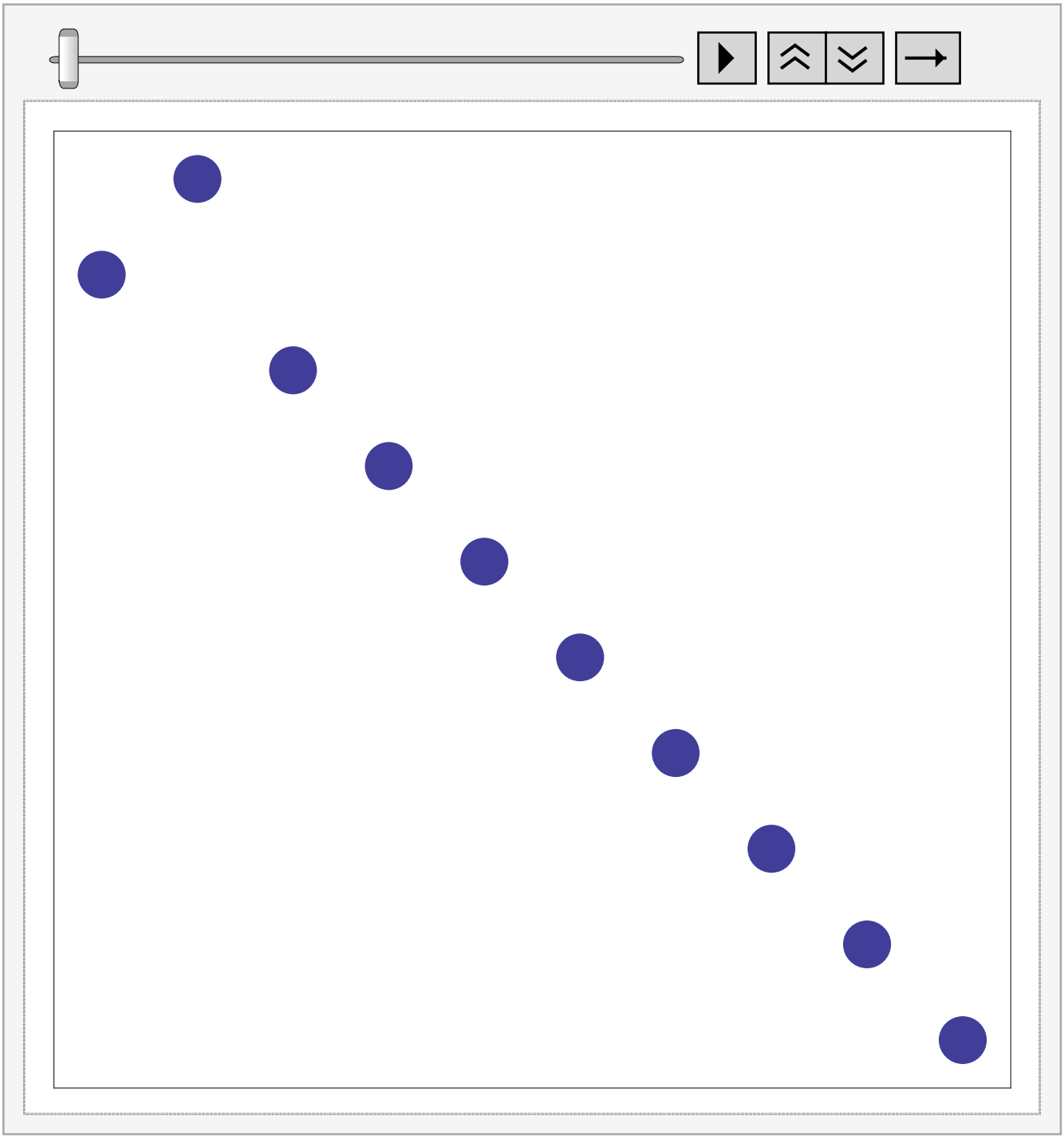


... řazení vkládáním (Insert-sort)

```
lists = {};
insertionSort[data]
```

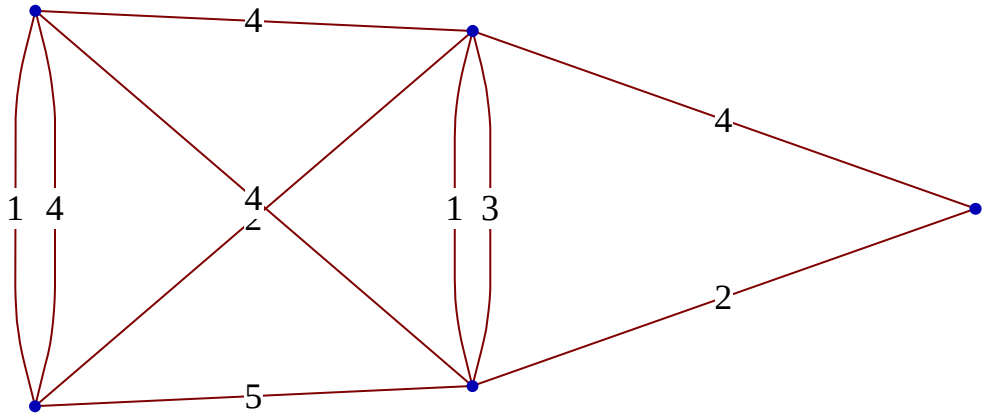
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
PermutationPlot[#, DisplayFunction → Identity] & /@
  lists;
ListAnimate[%, ImageSize → 300]
```

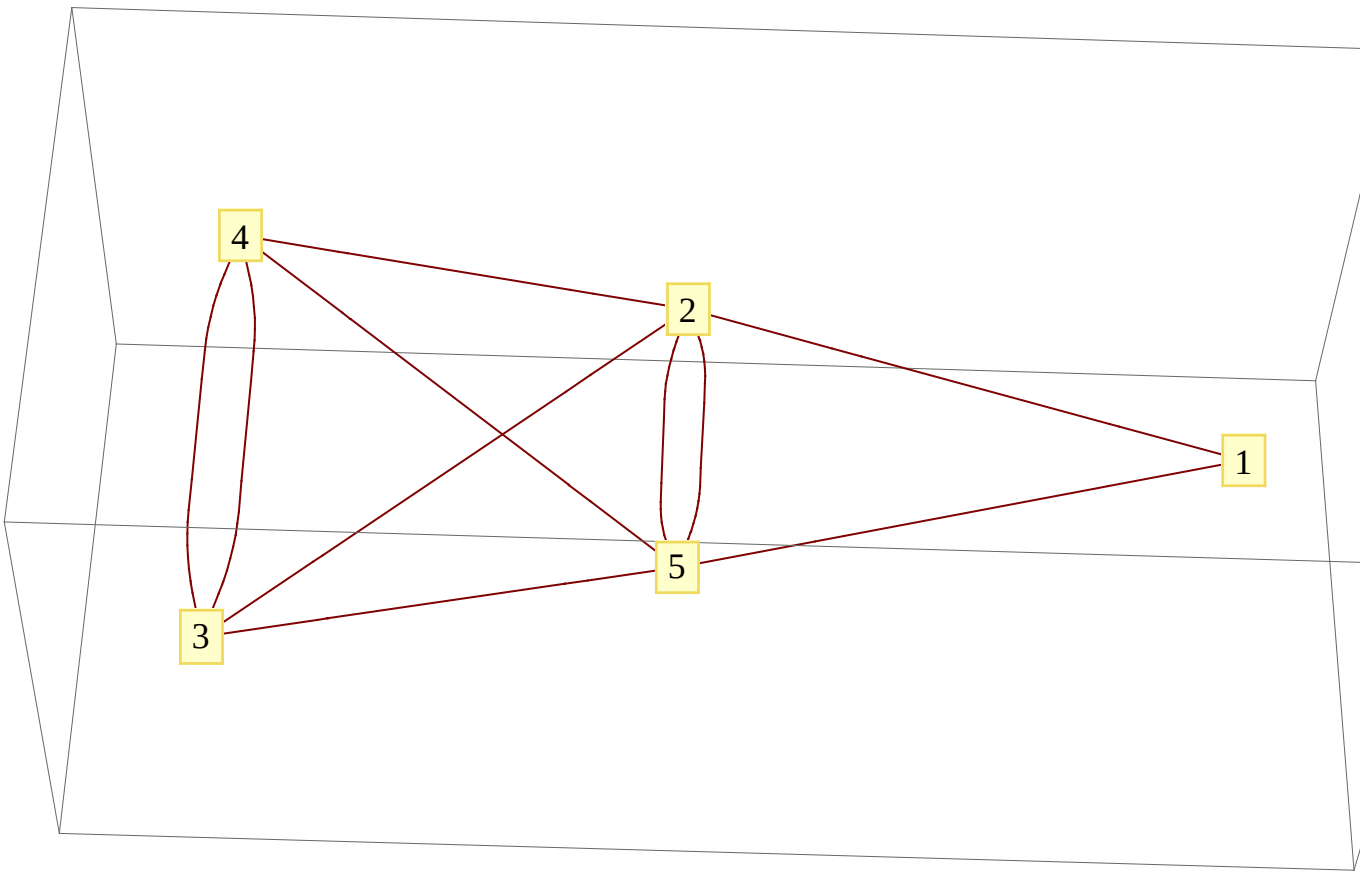


Ukázka : Grafové algoritmy, vizualizace

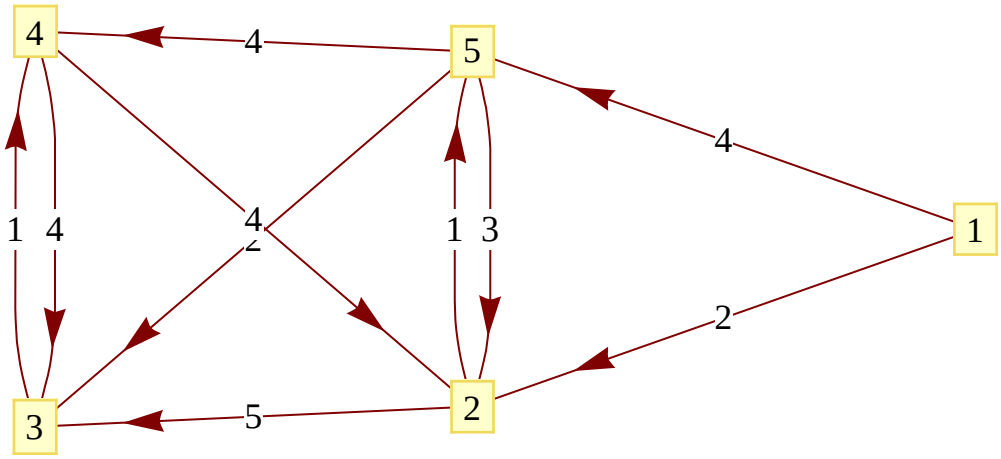
```
GraphPlot[g]
```



```
GraphPlot3D[g, VertexLabeling → True]
```



```
GraphPlot[g, VertexLabeling → True,
 DirectedEdges → True]
```



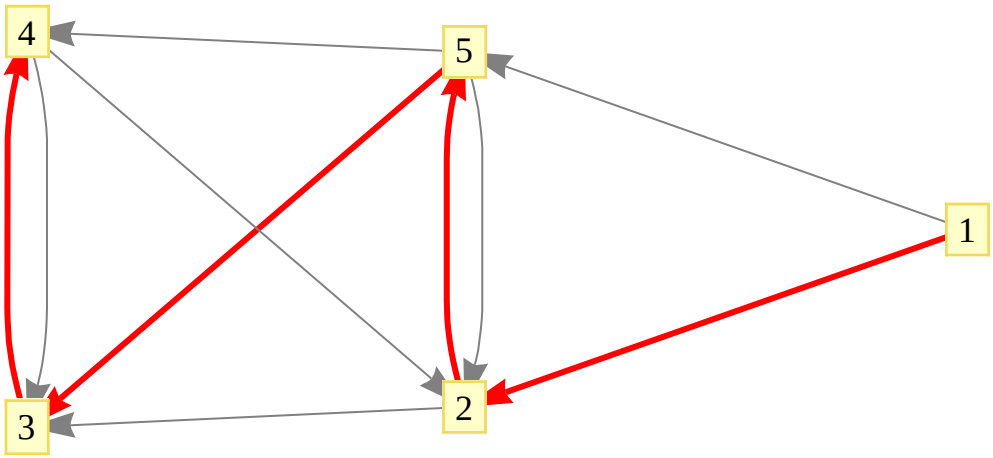
Ukázka implementace Dijkstrova algoritmu

```
For[i = 1, i ≤ Length[uList],
 q = enqueue[q, uList[[i]]]; i++]
```

```
While[! empty[q],
 {u, q} = extractMin[q]; u = u[[2]];
 (relax[u, #[[1]], #[[2]]]) & /@ adj[[u]];
]
```

- **Výsledek - nejkratší cesta z uzlu "1"**

```
GraphPlot[g, VertexLabeling → True,
 DirectedEdges → True, EdgeRenderingFunction →
  (If[p[[#2[[2]]]] == #2[[1]], {Red, Thickness[0.005],
     Arrow[#1]}, {GrayLevel[0.5], Arrow[#1]}] &)]
```



Zeměpis, chemie...

K dispozici je rozsáhlá **on-line databáze**, obsahující data z rozličných oborů.

Ukázky použití - CountryData

Aktuální populace zvoleného státu

```
CountryData["France", "Population"]
```

$$6.23121 \times 10^{7}$$

Země skupiny G8

```
CountryData["G8"]
```

```
{Canada, France, Germany, Italy, Japan,
 Russia, UnitedKingdom, UnitedStates}
```

Hrubý domácí produkt zemí skupiny G8 ...

```
CountryData[#, "GDP"] & /@ CountryData["G8"]
```

$$\{1.13176 \times 10^{12},\ 2.12658 \times 10^{12},\ 2.79486 \times 10^{12},\ 1.76247 \times 10^{12},\ 4.55895 \times 10^{12},\ 7.65968 \times 10^{11},\ 2.1988 \times 10^{12},\ 1.24558 \times 10^{13}\}$$

... přehledněji

```
Grid[{#, CountryData[#, "GDP"]} & /@
  CountryData["G8"], Frame → All]
```

Canada	1.13176×10^{12}
France	2.12658×10^{12}
Germany	2.79486×10^{12}
Italy	1.76247×10^{12}
Japan	4.55895×10^{12}
Russia	7.65968×10^{11}
UnitedKingdom	2.1988×10^{12}
UnitedStates	1.24558×10^{13}

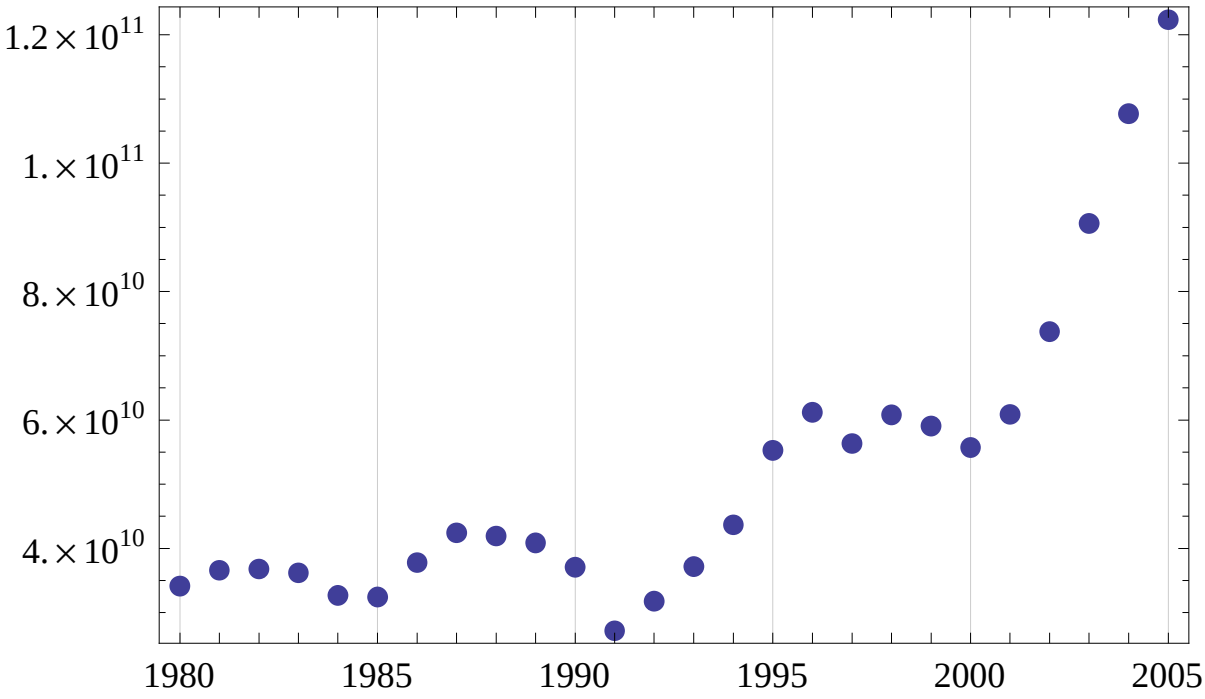
... kolik barelů ropy za den spotřebují ...

```
Plus @@ (CountryData[#, "OilConsumption"] & /@
    CountryData["G8"])
```

3.9041×10^{7}

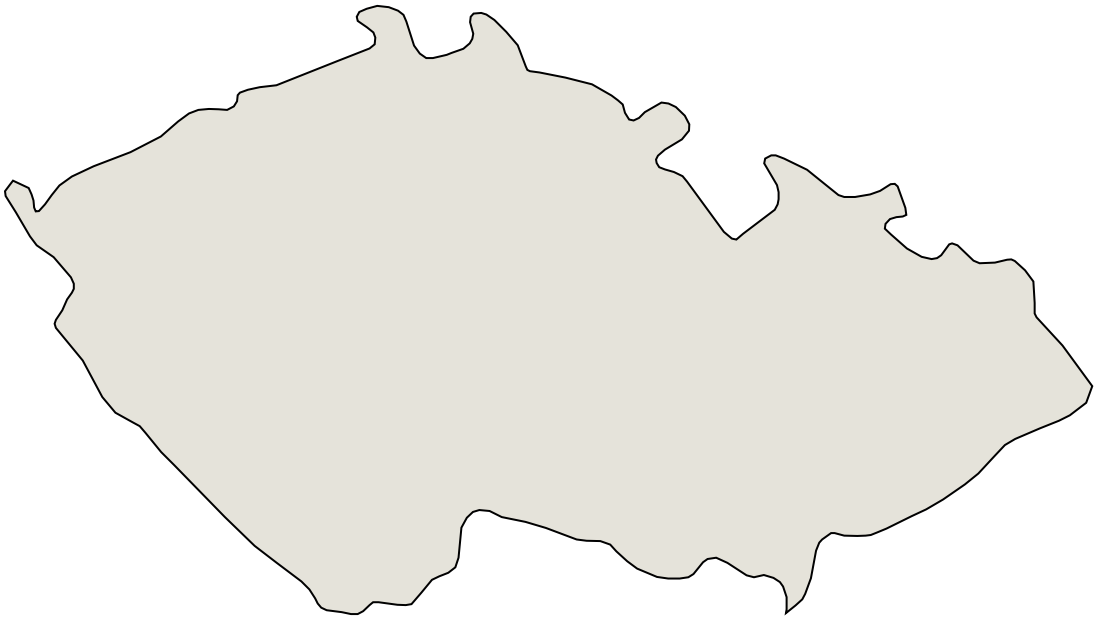
Hrubý domácí produkt České republiky v letech 1980 až 2005

```
DateListPlot[
 CountryData["CzechRepublic", {"GDP", {1980, 2005}}],
 PlotStyle → {PointSize[0.02]}]
```



Možnosti kreslení map ...

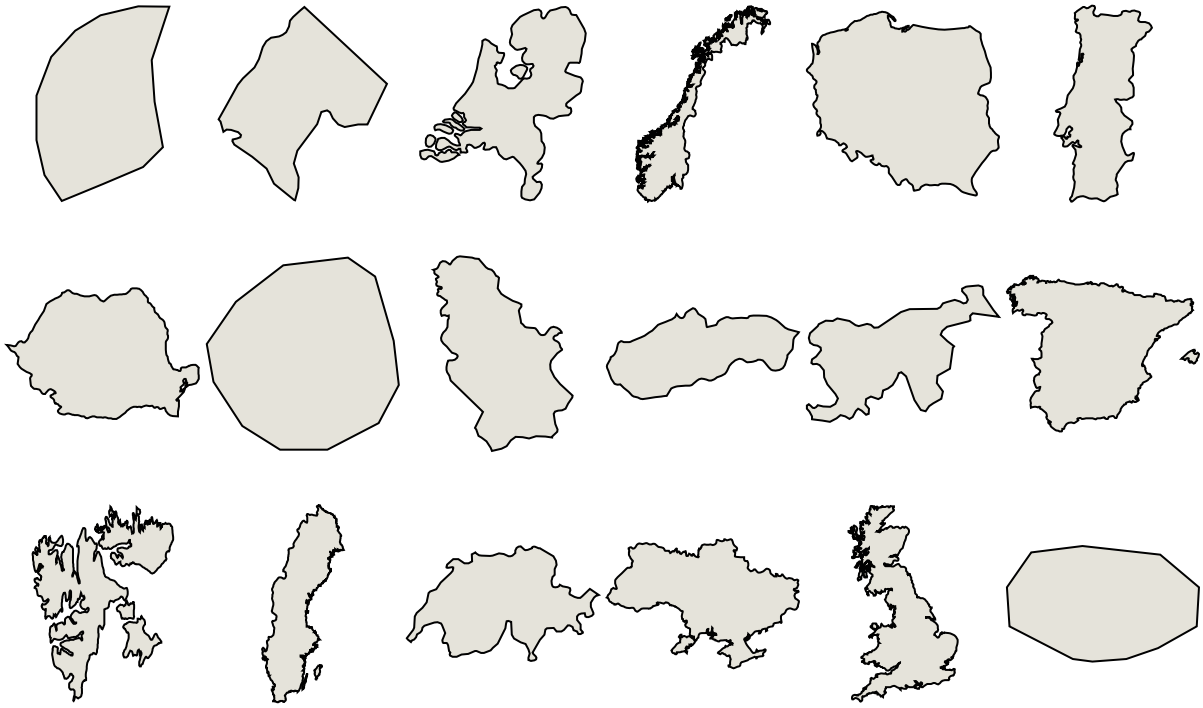
```
CountryData["Czech Republic", "Shape"]
```



```
CountryData["World", "Shape"]
```

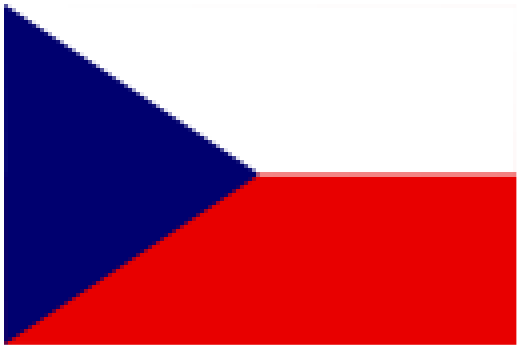
... jednotlivé evropské státy...

```
Row[Show[CountryData[#, "Shape"],
    ImageSize → {50, 50}] & /@ CountryData["Europe"]]
```



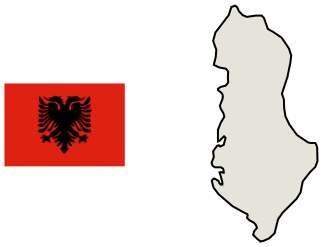
- **... a jejich vlajky**

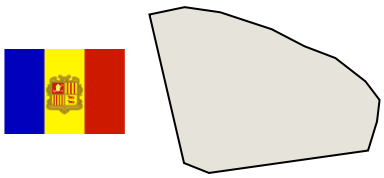
```
CountryData["Czech Republic", "Flag"]
```



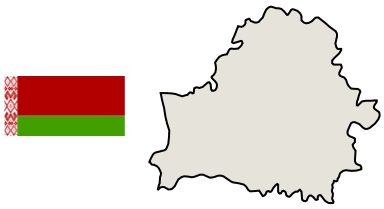
```
Row[Show[CountryData[#, "Flag"],
    ImageSize → {50, 50}] & /@ CountryData["Europe"]]
```

```
({#, Show[CountryData[#, "Flag"],
      ImageSize → {30, 30}], Show[CountryData[
       #, "Shape"], ImageSize → {60, 60}]} & /@
   CountryData["Europe"]) // TableForm
```

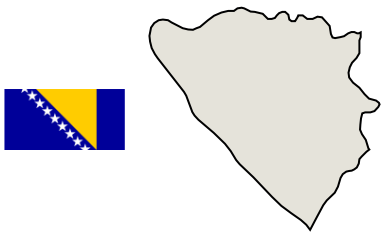
Albania 

Andorra 

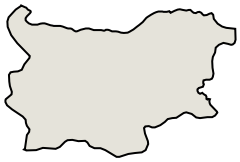
Austria 

Belarus 

Belgium 

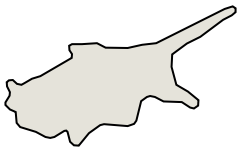
BosniaHerzegovina 

Bulgaria

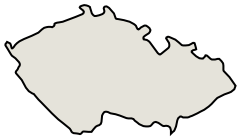


Croatia

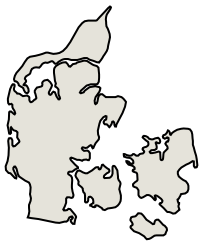
Cyprus



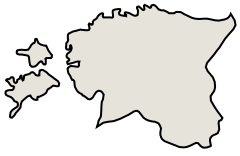
CzechRepublic



Denmark

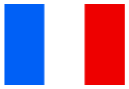


Estonia



Finland

France

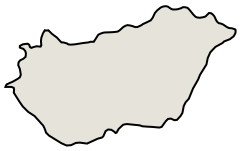


Germany

Gibraltar

Greece

Hungary

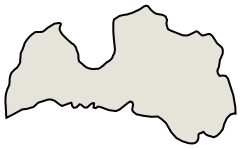


Iceland

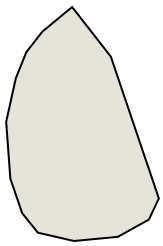
Ireland

Italy

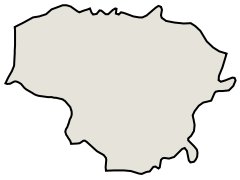
Latvia



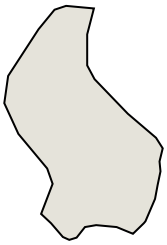
Liechtenstein



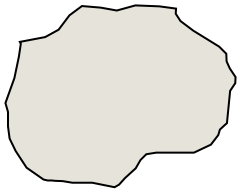
Lithuania



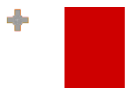
Luxembourg

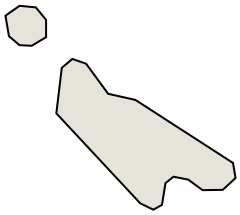


Macedonia



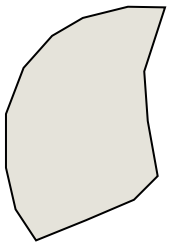
Malta



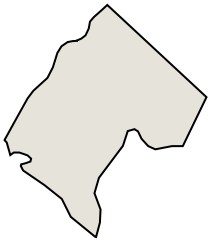


Moldova

Monaco



Montenegro



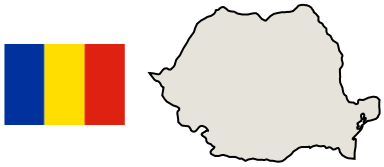
Netherlands

Norway

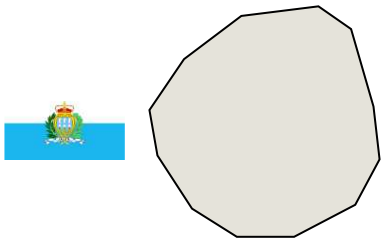
Poland

Portugal

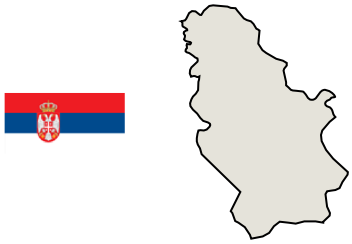
Romania



SanMarino

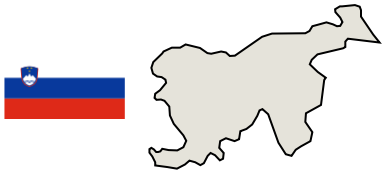


Serbia



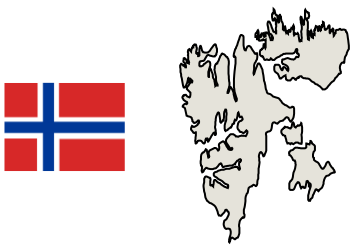
Slovakia

Slovenia



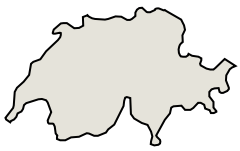
Spain

Svalbard



Sweden

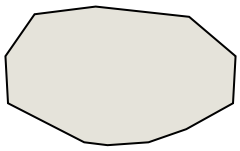
Switzerland



Ukraine

UnitedKingdom

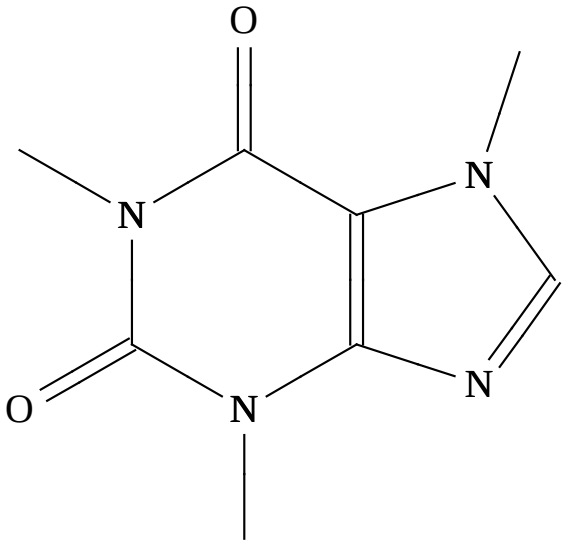
VaticanCity



Ukázky použití - ChemicalData

- **Vizualizace molekuly kofeinu ...**

```
ChemicalData["Caffeine"]
```


O
N
N
N
O
N

... 3 D

```
ChemicalData["Caffeine", "MoleculePlot"]
```