

Regulární výrazy, práce s textem a daty

IB111 Úvod do programování skrze Python

2014

- dnes: práce s textem
- příště: práce s obrázky

- příkazy pro práci s textem/obrázky
- ukázky na konkrétních příkladech
- použití dříve probíraných konstrukcí
- uvažování o problémech, volba přístupu, kladení otázek

Příklady:

- statistiky o studentech
- frekvenční analýza textu
- zpracování dotazníku, ankety

- v tomto kurzu pro zjednodušení pracujeme jen s anglickou abecedou
- resp. s texty bez hacku a carek
- pro zájemce viz např.
`http://www.py.cz/PythonUnicodeCestina`
- základ: specifikovat kódování na začátku souboru
`# -*- coding: utf-8 -*-`

Regulární výrazy: motivace

- vyhledání e-mailových adres v textu
- vyhledání odkazů v HTML dokumentu
- náhrada „jméno příjmení“ za „příjmení jméno“
- změna formátu datumů
- odstranění bílých znaků

Regulární výrazy: použití

- programování
- textové editory
- příkazová řádka: např. `grep`
- teorie: formální jazyky, konečné automaty

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.

OH NO! THE KILLER MUST HAVE FOLLOWED HER ON VACATION!



BUT TO FIND THEM WE'D HAVE TO SEARCH THROUGH 200 MB OF EMAILS LOOKING FOR SOMETHING FORMATTED LIKE AN ADDRESS!



IT'S HOPELESS!

EVERYBODY STAND BACK.



I KNOW REGULAR EXPRESSIONS.



- obecně používaný nástroj
- základní syntax stejná ve většině jazyků, prostředí
- následuje:
 - základní syntax regulárních výrazů
 - použití v Pythonu
- nerozebíráme všechny technické detaily (podrobněji viz dokumentace)

Ukázka

```
import re
f = open("testovaci-soubor.txt")
for radek in f.readlines():
    if re.search(r'[a-z]+@[a-z]+\\.cz', radek):
        print radek
f.close()
```

Znaky a speciální znaky

- základní znak „vyhoví“ právě sám sobě
- např. „cz“ v předchozím příkladě
- speciální znaky: `. ^ $ * + ? { } [] \ | ()`
 - umožňují konstrukci složitějších výrazů
 - chceme, aby odpovídaly příslušnému symbolu \Rightarrow prefix `\`

Výběr ze skupiny znaků

[]

- [abc] – jeden ze znaků a, b, c
- [a-z] – výběr z intervalu (malé písmeno anglické abecedy)
- ^ na začátku výběru = negace:
 - [^abc] cokoliv jiného než a, b, c

Často používané skupiny znaků

<code>\d</code>	Čísla: <code>[0-9]</code>
<code>\D</code>	Cokoliv kromě čísel: <code>[^0-9]</code>
<code>\s</code>	Bílé znaky: <code>[\t\n\r\f\v]</code>
<code>\S</code>	Cokoliv kromě bílých znaků: <code>[^\t\n\r\f\v]</code>
<code>\w</code>	Alfanumerické znaky: <code>[a-zA-Z0-9_]</code>
<code>\W</code>	Nealfanumerické znaky: <code>[^a-zA-Z0-9_]</code>

Speciální symboly

- . libovolný znak
- ^ začátek řetězce
- \$ konec řetězce
- | alternativa – výběr jedné ze dvou možností

Jaký je význam následujících výrazů?

- `kocka|pes`
- `^[Pp]rase$`
- `\d[A-Z]\d \d\d\d\d`

Opakování

*	nula a více opakování
+	jedno a více opakování
?	nula nebo jeden výskyt
{m, n}	m až n opakování

Pozn. *, + jsou „hladové“, pro co nejmenší počet opakování *?, +?

Jaký je význam následujících výrazů?

- `^\s*Nadpis`
- `^a.+a$`
- `\d{3}\s?\d{3}\s?\d{3}`
- `[a-z]+@[a-z]+\.` `.cz`
- `^To:\s*(fi|kit)(-int)?@fi\.muni\.cz`

Která z následujících slov vyhoví jednotlivým výrazům?

	<code>p[ars]e</code>	<code>p[ars]*e</code>	<code>p[[^]ars]e</code>
<code>ps</code>			
<code>pes</code>			
<code>pse</code>			
<code>poe</code>			
<code>prase</code>			
<code>poklice</code>			

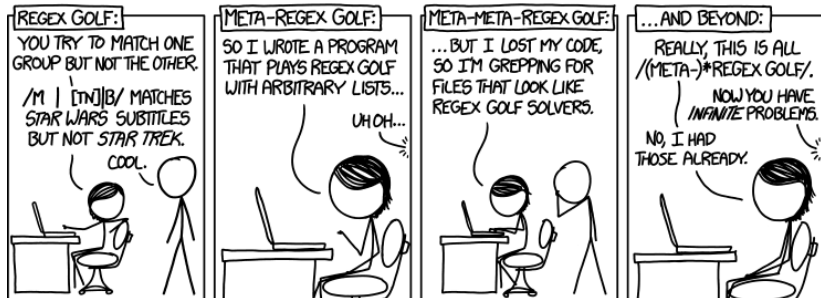
Která z následujících slov vyhoví jednotlivým výrazům?

	<code>p[ars]e</code>	<code>p[ars]*e</code>	<code>p[~ars]e</code>
<code>ps</code>	×	×	×
<code>pes</code>	×	✓	×
<code>pse</code>	✓	✓	×
<code>poe</code>	×	×	✓
<code>prase</code>	×	✓	×
<code>poklice</code>	×	×	×

Kontrola tabulky v Pythonu

```
texty = ["ps", "pes", "pse", "poe", "prase", "poklice"]
vyrazy = [ r'p[ars]e', r'p[ars]*e', r'p[^\s]e' ]
for text in texty:
    print text,
    for vyraz in vyrazy:
        if re.search(vyraz, text): print 1,
        else: print 0,
    print
```

Regulární výrazy: xkcd



<http://xkcd.com/1313/>
http://www.explainxkcd.com/wiki/index.php/1313:_Regex_Golf
<https://regex.alf.nu/>

Regulární výrazy v Pythonu

- knihovna `re` (`import re`)
- `re.match` – hledá shodu na začátku řetězce
- `re.search` – hledá shodu kdekoliv v řetězci
- (`re.compile` – pro větší efektivitu)
- „raw string“ – `r'vyraz'` – nedochází k interpretaci speciálních znaků jako u běžných řetězců v Pythonu

Regulární výrazy v Pythonu: práce s výsledkem

- `match/search` vrací „MatchObject“ pomocí kterého můžeme s výsledkem pracovat
- pomocí kulatých závorek `()` označíme, co nás zajímá
- `Alice\s+(\w+)`

Regulární výrazy v Pythonu: práce s výsledkem

```
>>> m = re.match(r"(\w+) (\w+)", \
                  "Isaac Newton, fyzik")
>>> m.group(0)
'Isaac Newton'
>>> m.group(1)
'Isaac'
>>> m.group(2)
'Newton'
```

Substituce

- nahrazení řetězce jiným výrazem
- nejen statické řetězce, ale i regulární výrazy
- `re.sub`

Rozdělení řetězce

- `split` – rozdělí řetězec podle zadaného podřetězce, vrací seznam částí
- `join` – spojení seznamu řetězců do jednoho

```
>>> retezec = "Holka modrooka neseďavej u potoka"  
>>> retezec.split()  
['Holka', 'modrooka', 'nesedavej', 'u', 'potoka']  
>>> retezec.split('o')  
['H', 'lka m', 'dr', '', 'ka neseďavej u p', 't', 'ka']  
>>> retezec.split('ka')  
['Hol', ' modroo', ' neseďavej u poto', '']
```

Řetězce: další funkce

- `find`, `count` – vyhledávání a počítání podřetězců
- `lower`, `upper` – převod na malá/velká písmena
- `ljust`, `rjust`, `center` – zarovnání textu
- `rstrip`, `rstrip` – ořezání bílých znaků na začátku/konci

Práce se soubory – připomenutí

Otevírání a zavírání:

- `f = open("mujsoubor.txt")` – otevření pro čtení
- `f = open("mujsoubor.txt", "w")` – otevření pro zápis
- `f.close()` – uzavření souboru
- zápis pomocí `with` – lepší praxe (ale pokročilejší, souvisí s výjimkami)

Čtení a zápis:

- `f.readline()` – vrátí další řádek ze souboru
- `f.readlines()` – vrátí seznam všech zbývajících řádků
- `f.write(retezec)` – zapíše do souboru

Příklad: Zpracování HTML

- vstup: HTML soubor
- cíl: vybrat odkazy a nadpisy
- ukážeme naivní řešení se soubory, reg. výrazy
- systémovější řešení: využití knihoven pro práci s URL zdroji, parsování HTML

Příklad: Zpracování HTML

Information [for applicants](#) [for students](#) [for industry partners](#)

Homepage >

About the faculty >

Admission >

Studies >

Projects >

Research and development >

International studies >

E-learning >

Industrial partners >

News >

Contacts >

Do you know:
Faculty is constructing Center for Education, Research and Innovation for ICT in Brno - CERIT.
> [more about the faculty](#)

Faculty of Informatics Masaryk University

The Faculty of Informatics with its 2,200 students offers a wide range of fields to enroll in. These are subsumed under its Bachelor, Master, and doctoral degree programs. The Bachelor programs aim to provide their students with academic basics of informatics. The objective of the Master ones is to train students in these fields ranging from theoretical informatics to information systems.

Contacts | Why study on FI? | Personal Administration | Faculty Administration

News

> [New open position 2012-10-24](#)

> [more news](#) | [RSS](#)

[News for Bc. and Mgr. students](#)

[News for Ph.D. students](#)

[News about internship abroad](#)

[News about research and development](#)

[News about industrial partnership](#)

Links

[Information System MMU](#)

[Library FI](#)

[Lecture videos](#)

Join us at FI MMU!

```
<div id="header">
  <div id="logo">
    <h1>Faculty of Informatics Masaryk University</h1><span></span>
  </div>
</div>

<div id="altNav">
  <a href="#wrapper">go to content</a>
  <a href="#mm">go to menu</a>
  <a href="#search">go to search</a>
</div>
```

Hledání nadpisů

```
def najdi_nadpisy(jmeno_souboru):  
    soubor = open(jmeno_souboru)  
    for radek in soubor.readlines():  
        m = re.search(r'<h(\d)>(.*?)</h\d>', radek)  
        if m:  
            print m.group(1), "\t", m.group(2)  
    soubor.close()
```

Kdy nebude fungovat korektně?

Hledání odkazů

- stejná základní kostra, jen jiný regulární výraz
- pokus č. 1:
- `(.*)`
- nedostatky?

Hledání odkazů

- stejná základní kostra, jen jiný regulární výraz
- pokus č. 1:
- `(.*)`
- nedostatky?

```
<a href="www.seznam.cz">Seznam</a>, <a href="www.google.com">Google</a>
```

```
<a href="www.seznam.cz" target="_blank">Seznam</a>
```


- rozšíření (stále nedostatečná):
 - `(.*?)`
 - `(.*?)`
- * – „hladové“ hledání („co nejvíc“)
- *? – „co nejmíň“

Příklad: Jak vykrást banku?

Přesněji: Jak převzít kurzy ze stránky ČNB?

Platnost od 05.11.2012 Pořadí: 215

země	měna	množství	kód	kurz
Austrálie	dolar	1	AUD	20,454
Brazílie	real	1	BRL	9,706
Bulharsko	lev	1	BGN	12,902
Čína	renminbi	1	CNY	3,162
Dánsko	koruna	1	DKK	3,383
EMU	euro	1	EUR	25,235
Filipíny	peso	100	PHP	47,847

```
477 <h3 class="kurzy_tisk">Platnost od 05.11.2012 Pořadí: 215</h3>
478 <table class="kurzy_tisk">
479 <tr><th>země</th><th>měna</th><th>množství</th><th>kód</th><th>kurz</th></tr>
480 <tr><td>Austrálie</td><td>dolar</td><td align="right">1</td><td>AUD</td><td align="right">20,454</td></tr>
align="right">1</td><td>BRL</td><td align="right">9,706</td></tr><tr><td>Bulharsko</td><td>lev</td><td align="right">12,902</td></tr><tr><td>Čína</td><td>renminbi</td><td align="right">1</td><td>CNY</td><td align="right">3,162</td></tr><tr><td>Dánsko</td><td>koruna</td><td align="right">1</td><td>DKK</td><td align="right">3,383</td></tr><tr><td>EMU</td><td>euro</td><td align="right">1</td><td>EUR</td><td align="right">25,235</td></tr><tr><td>Filipíny</td><td>peso</td><td align="right">100</td><td>PHP</td><td align="right">47,847</td></tr></table>
```

Příklad: Kurzy ze stránky ČNB

Základní řešení:

- najít řádek, na kterém jsou kurzy (začíná `<tr><td>Aust`)
- rozdělit na řádky tabulky (podle `</tr><tr>`)
- hledat trojice velkých písmen a za nimi čísla
- převést na typ float, uložit do slovníku

Nevýhody?

Příklad: Kurzy ze stránky ČNB

```
def zjistí_kurzy(jmeno_souboru):
    kurzy = {}
    soubor = open(jmeno_souboru)
    for radek in soubor.readlines():
        if re.match(r'<tr><td>Aust', radek):
            for radek_tab in radek.split('</tr><tr>'):
                m = re.search(
                    r'<td>([A-Z]{3}).*right">([\d,]+)</td>'
                    radek_tab)
                kurzy[m.group(1)] =
                    float(re.sub(',', '.', m.group(2)))
    soubor.close()
    return kurzy
```

Příklad: informace o studentech

Export informací z ISu (CSV soubor):

```
1.;50668;"Sukany, Martin";zk;"FI B-AP BcAP [sem 1, roc
2.;421714;"Veznik, Ondrej";zk;"FI B-AP SOCI [sem 2, roc
3.;564138;"Machala, David";zk;"FI B-AP BcAP [sem 1, roc
4.;43583;"Mikes, Martina";zk;"FF B-FI PLIN [sem 5, cyk
5.;81908;"Sulc, Tomas";zk;"FF B-FI PLIN [sem 5, cyk 1]"
6.;844632;"Novak, Karel";zk;"FI B-IN PSK [sem 1, roc 1]"
7.;798639;"Dunickova, Dagmar";zk;"FI B-AP SOCI [sem 1,
8.;195660;"Stipsky, Tomas";zk;"FI B-AP BcAP [sem 1, roc
9.;278740;"Fojt, Roman";zk;"FI B-AP INVS [sem 3, roc 2]"
10.;236293;"Zachar, Samuel";zk;"FI B-IN UMI [sem 1, roc
```

Pozn. Příklad je „mutovaný“ z důvodu ochrany osobních údajů.

Statistiky o studentech

- výpis křestních jmen (abecedně seřazený)
- statistika studovaných oborů

Výpis křestních jmen

```
def krestni_jmena(jmeno_souboru):  
    f = open(jmeno_souboru)  
    jmena = []  
    for radek in f.readlines():  
        m = re.match(r'\d+\.;\d+;"\w+, (\w+)', radek)  
        if m: jmena.append(m.group(1))  
    jmena.sort()  
    print " ".join(jmena)  
    f.close()
```

Jiné řešení: použití split

Statistiky oborů

```
def obory(jmeno_souboru):
    f = open(jmeno_souboru)
    vyskyty_oboru = {}
    for radek in f.readlines():
        m = re.search(r'\s(\w+) \[sem', radek)
        if m:
            obor = m.group(1)
            vyskyty_oboru[obor] = \
                vyskyty_oboru.get(obor, 0) + 1
    f.close()
    for obor in vyskyty_oboru.keys():
        print obor, vyskyty_oboru[obor]
```

Nedostatky: např. studenti studující více oborů.

Procvičení regulárních výrazů

- `http://tutor.fi.muni.cz`
- úloha Regulární výrazy
- 40 příkladů
- vyhledávání, nahrazování

Náhodnostní imitace vstupního textu

- vstup: rozsáhlý text
- výstup: náhodně generovaný text, který má „podobné charakteristiky“ jako vstupní text
- imitace na úrovni písmen nebo slov

Náhodnostní imitace vstupního textu

I špiské to pole kavodali pamas ne nebo kdy v Dejný Odm sem uvalini se zabijí s Pan stěží ře, a silobe lo v ne řečekovicích blova v nadrá těly jakvěmutelaji rohnutkohonebout anej Fravinci V A pěk finé houty. zal Jírakočítencej ské žil, kdDo jak a to Lorskříže si tomůžu schno mí, kto.

Kterak král kočku kupoval V zemi Taškářů panoval král a zapřisáhl se velikou přísahou že bude pochválena První pán si jí ani nevšimnul zato druhý se rychle shýbl a Jůru pohladil Aha řekl sultán a bohatě obdaroval pana Lustiga koupil od něho telegram z Bombaje v Indii není o nic horší člověk nežli někdo z mých hraček Kdepak mávl Vašek rukou

Základní přístup

- 1 vstupní text \Rightarrow statistiky textu
- 2 statistiky \Rightarrow generování náhodného textu

Co jsou vhodné statistiky?

- základ: frekvence písmen (slov)
- rozšíření: korelace mezi písmeny (slovy)

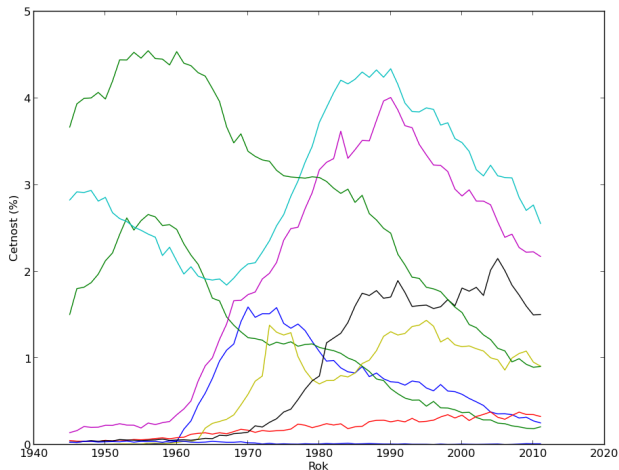
příklad: pokud poslední písmeno bylo **a**:

- **e** velmi nepravděpodobné (méně než obvykle)
- **l, k** hodně pravděpodobná (více než obvykle)

- základní frekvenční analýza – datová struktura slovník
písmeno \Rightarrow frekvence
- rozšířená analýza – slovník slovníků
písmeno \Rightarrow { písmeno \Rightarrow frekvence }

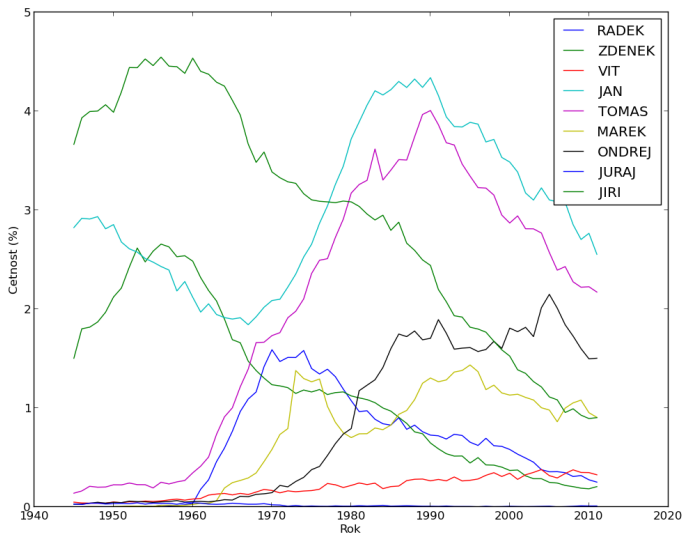
- data: četnosti jmen, příjmení podle roků, krajů, ...
- zdroj: Ministerstvo vnitra ČR
<http://www.mvcr.cz/clanek/cetnost-jmen-a-prijmeni-722752.aspx>
- XLS – pro zpracování v Pythonu uložit jako CSV (comma-separated values)
- doporučené cvičení
 - snadno zpracovatelné
 - zajímavá data
 - cvičení na vymýšlení otázek
- následuje několik ukázek pro inspiraci ...

Jména vyučujících IB111

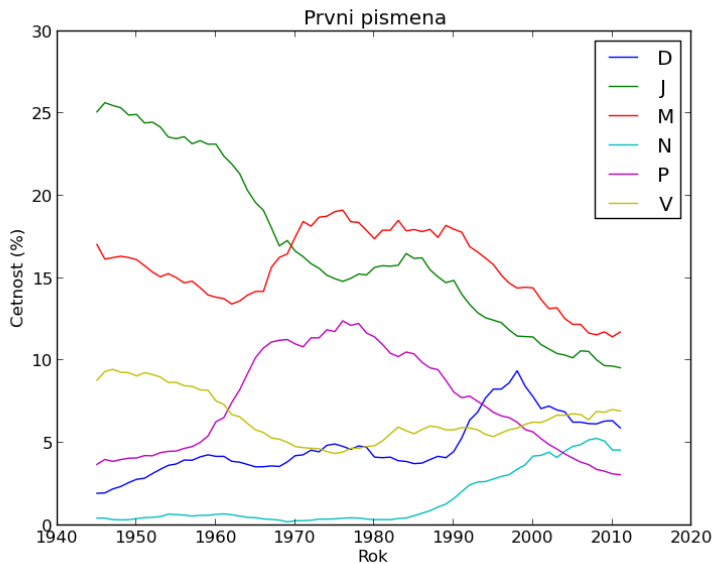


Radek, Zdeněk, Vít, Jan, Tomáš, Marek, Ondřej, Juraj, Jiří

Jména vyučujících IB111



První písmena jmen



U kterých jmen nejvíce roste/klesá popularita?

- co to vlastně znamená?
- jak formalizovat?

Nejdelší růst/pokles

Kolik let v řadě roste popularita jména:

- Tobiáš – 14
- Viktorie, Ella, Sofie – 9
- Elen, Tobias – 8

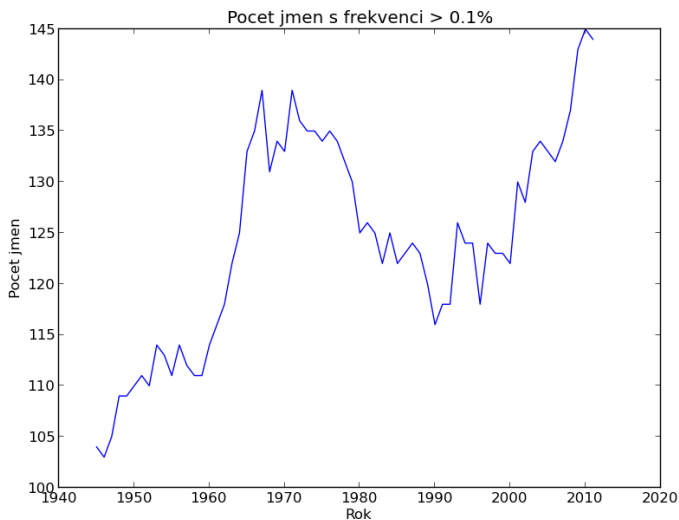
Kolik let v řadě klesá popularita jména:

- Jana – 26
- Martin – 21
- Petra – 11
- Zdeněk – 9

Největší skok v popularitě za 10 let

- alespoň desetinásobný nárůst popularity:
Sofie, Elen, Amálie, Ella, Nicol, Nella, Tobias
- pokles alespoň o 60 %:
Petra, Pavlína, Martina

Počty používaných jmen



Otevřená data / Open data

- <http://www.opendata.cz>
- <http://www.otevrenadata.cz>
- <http://www.data.gov/>
- <http://data.gov.uk/>

Zpracování dat seriózněji

využití existujících knihoven:

- načítání dat ve standardních formátech: HTML, XML, JSON, CSV, ...
- operace s daty: numpy, pandas
- vizualizace: matplotlib

prostředí ipython

- regulární výrazy – obecně užitečný nástroj
- práce s textem, soubory, daty v Pythonu
- příklady
 - kurzovní lístek
 - zpracování seznamu studentů
 - imitace textu
 - statistiky jmen

příště: obrázky