

1. What regular expressions do you use to extract all the dates in this blurb:

<http://bit.ly/regexexercise2> and to put them into the following format YYYY-MM-DD ?

```
REGULAR EXPRESSION
: / (\d+).(\d+)..?(\d{4})

TEST STRING
Juan Ponce de León sights Florida for the first time, on 3.27.1513
Giovanni da Verrazzano explored the Atlantic coast of North America under French employ, on 4.17.1524
The Roanoke Colony was found deserted, on 8/15/1590
John Smith founded the Jamestown settlement, on 5/14.1607
The Dutch laid claim to the territories of New Netherland, on 11.11.1614
The Massachusetts Bay Colony founded, on 3-4-1629

SUBSTITUTION
$3-$1-$2

Juan Ponce de León sights Florida for the first time, on 1513-3-27
Giovanni da Verrazzano explored the Atlantic coast of North America under French employ, on 1524-4-17
The Roanoke Colony was found deserted, on 1590-8-15
John Smith founded the Jamestown settlement, on 1607-5-14
The Dutch laid claim to the territories of New Netherland, on 1614-11-11
The Massachusetts Bay Colony founded, on 1629-3-4
```

man bruger udtrykket: “(\d+).(\d+)..?(\d{4})” for at fange de forskellige datoer.

Den grønne del fanger måneder, i det at teksten er skrevet af en amerikaner, de blå dele finder, men fanger ikke delene mellem datoerne, den orange del fanger dagen og den lilla del fanger årstallet.

I andre ord fortæller udtrykket, at computeren skal lede efter et element som består af mindst 1 tal, efterfulgt af et vilkårligt tegn, efterfulgt af mindst 1 tal, efterfulgt af 1 vilkårligt tegn og 1 muligt tegn, efterfulgt af et tal på 4 cifre, samt at computeren skal huske de fire tal. Derefter går man ned i “substitution”, hvor man fortæller, hvilken rækkefølge de forskellige segmenter skal være i. \$1 indikerer den første parentes og \$2 indikerer den anden parentes, osv.

Link til løsningen: <https://regex101.com/r/vjyd8K/1>

2. Write a regular expression to convert the stopwordslist (list of most frequent Danish words) from Voyant in <http://bit.ly/regexexercise3> into a neat stopwords list for R (which comprises "words" separated by commas, such as <http://bit.ly/regexexercise4>). Then take the stopwordslist from R <http://bit.ly/regexexercise4> and convert it into a Voyant list (words on separate line without interpunction)

2.1 voyant til R

Link til opgaveløsningen: <https://regex101.com/r/5X97IY/1>

The screenshot shows the regex101.com interface. At the top, it says "REGULAR EXPRESSION" and "587 matches (4.108 steps, 8.58ms)". The regular expression is `/([a-zæøå0-9.'"]+)(\n)?/gm`. Below this is the "TEST STRING" section, which contains a list of words: "voigt", "vor", "vore", "vores", "vs", "wedell", "westergaard", "wilhjelm", and "yildiz". The "SUBSTITUTION" section shows the result of the substitution: `"$1",` followed by a list of words in quotes, separated by commas: "terens", "sydamager", "synes", "syntes", "syv", "tag", "tage", "tatovere", "tatoverer", "tatovering", "tatoveringen", "tatoveringer", "temmelig", "theil", "theil", "thi", "thorndahl", "thulesen", "ti", "tidenden", "tidligere", "til", "tilbage", "tindemans", "tit", "to", "tre", "uafhængig", "uafhængige", "uafhængiges", "ud", "uden", "udover", "ufg", "under", "undtagen", "vækstfremmere", "vær", "være", "været", "var", "ved", "vel", "færdministerens", "vel", "færdsminister", "vel", "færdsministeren", "vestager", "vi", "via", "vibjerg", "vil", "ville", "vivike", "voigt", "vor", "vore", "vores", "vs", "wedell", "westergaard", "wilhjelm", "yildiz". The bottom right corner shows "1:1".

udtrykker fortæller computeren at den skal finde og huske et element som kan bestå af adskillige tal, bogstaver og tegn, samt at den skal finde og huske linebreaks, derefter går man ned i "substitution", hvor man fortæller computeren at den kun skal printe de elementer som består af tal, bogstaver og tegn, hvor hvert ord er omringet af citationstegn og efterfulgt af et komma

<https://regex101.com/r/8Drs2z/1>

REGULAR EXPRESSION

```
REGULAR EXPRESSION                                     406 matches (6.494 steps, 7.6ms) ⓘ
: / (["])?([a-zæøåéü'0-9.]+).(["])?([,])?([.])? / gm  📄

TEST STRING

"højtærede", "rimstad", "mill", "beh", "weikop", "udskrivn", "wetlesen",
"gottschalck", "westerby", "magnussens", "asmussen", "bækgaard", "dupont",
"diderichsen", "moltke", "henry", "sigsgaard", "haunstrup", "bundgård",
"reintoft", "lysholt", "grünbaum", "andresen", "fremskridtspartiet",
"fremskridtspartiets", "langkilde", "maigaard", "skovmand", "bendix",
"valbak", "brauer", "lütken", "amagerby", "flygaard", "lindholt", "fp",
"dkp", "ingomar", "glensgård", "erlendsson", "nørlund", "lovf", "maisted",
"honoré", "tyroll", "hjortlund", "waldorff", "uwe", "askjær", "dræbye",
"nymann", "kalnæs", "bolvig", "cd", "tinning", "ingerlise", "holmsgård",
...

1:4334 — match 309, group 2

SUBSTITUTION                                           success (380µs)

$2\n

højtærede
rimstad
mill
beh
weikop
udskrivn
wetlesen
gottschalck
...
```

udtrykker fortæller computeren at den skal finde og huske det første citationstegn som er foran hvert ord, derefter et element af adskillige tal, bogstaver og tegn, derefter det efterfølgende citationstegn, komma og whitespace, derefter i "substitution", hvor man fortæller computeren at den skal kun printe den anden gruppe som den har fundet og husket, efterfulgt af et linebreak

3. Does OpenRefine alter the raw data during sorting and filtering?

Nej, OpenRefine ændrer ikke det oprindelige data under sortering og filtrering af datasæt. Når man arbejder med OpenRefine, sorterer det blot dataene visuelt, og disse handlinger påvirker kun midlertidigt, hvordan dataene vises. De faktiske datasæt forbliver uændrede. Dvs. den oprindelige fil forbliver den samme, indtil du eksporterer OpenRefine data for sig selv.

4. Fix the [interviews dataset](#) in OpenRefine enough to answer this question: "Which two months are reported as the most water-deprived/dryest by the interviewed farmer households?"

Først finder man frem til kolonnen "months_no_water", derefter går man ind under "cells" -> "transformation" hvori man indsætter følgende: `value.replace("[", "").replace("", "").replace(" ", "").replace("]", "")`

Derefter laver man en facet ud af den, hvori man under "change" indsætter følgende: `value.split(";")`

Ud fra dette kommer vi frem til at oktober og september var de to tørreste måneder



5. Real-Data Challenge: What are the 10 most frequent occupations "erhverv" among unmarried men or women of 20-30 years in [1801 Aarhus](#) census dataset? (hint: first select either men or women to shrink the dataset to a manageable size, then filter by age, and then use merging to cut the erhvervvariation ruthlessly.)

Først indsættes 3 filtre, den ene filtrerer efter kvinder, den næste filtrerer alle som er gift væk, dvs. enker og ugifte er inkluderet, dette gøres med et inverteret filter med en regex: `^gift`, derefter et filter for alder med en regex: `\b([2][0-9]|[3][0])\b`

Derefter laves en text facet, hvor alle muligheder indenfor OpenRefines "cluster" bruges, nogle mulige clustre kan blive overset, "inderste" og "indsidder" ses som det samme, hvis det starter med "tjener" ses det som "tjenestepige"

✕

koen

invert reset

kvinde

☐ case sensitive ☐ regular expression

✕

civilstand

invert reset

^gift

☐ case sensitive ☒ regular expression

✕

alder

invert reset

\b([2][0-9]|[3][0])\b

☐ case sensitive ☒ regular expression

✕

erhverv

change

79 choices Sort by: name count Cluster

Tienestepige	41
Væverske	23
Huusjomfru	18
Indsidder	17
Spinderske	14
Syepige	11
lever af sine midler	8
Inderste	7
Bryggerpige	5
Kokkepige	5
tjener faderen	5
Hospitalslem	4
lever af almisse	4
Skræderpige	4