記者ゼミIT講座2020

正規表現とは[どこ]んなもの\??

西尾能人（朝日新聞国際報道部）

本日のお願い

❶バックスラッシュ「\」と半角の円マーク「¥」は同じものだと思って下さい。
　フォント・言語設定など、その時々の環境で見え方が違うだけです。
❷しつこく質問をはさみます。後生ですから、チャットなどで反応して下さい。
❸このサイトをブックマークしておいて下さい。

https://regex101.com/

❹シナリオと戦わないで下さい。なぜそんなことする？ やる意味あるのか？ と突っ込みたくなるのを我慢して、とりあえず練習だと思ってやってみて下さい。

なにゆゑかひとりで池を五周する人あり算数の入試問題に（大松達知）

❺この資料の最終版も含め、練習に使うものは以下に置いておきます。緑色のボタンの **Code** >> **Download ZIP** と進み、解凍しておいて下さい。
https://github.com/nishioWU/JNPC2102
❻その中の「チートシート」を眺めながらご参加を。ディスプレイを複数使っている方以外は、あらかじめ印刷しておくとよいかもしれません。

必要なときが来ます。そしたら、思い出して

　苦労して入手したデータを報道に生かそう、というときに、そのまま使えることは、まずない。データダンプの中から必要な箇所だけ取り出したい、とか、数値として統計処理するため単位を取り除きたい、とか……。下準備として、何かしらの処理が必要になることだろう。
　手作業では大変だ。見逃したり、ミスしたりする可能性もある。そこで、今回は、パソコンに文字どおり「機械的」にやらせる方法を練習する。それが「正規表現」と呼ばれるものだ。1行だけの小さなプログラムだと思ってほしい。すぐにとは限らないが、情報公開請求をしたり、膨大なデータの解析をしたり、というときに、きっと役に立つ。
　細かなところまでは、覚えなくてよい。いざ必要になったときに思い出して、改めて調べた

り試したりすればOK。そのためにも、きょう紹介したサイトは有用なので、おすすめ！

これから取り組むこと

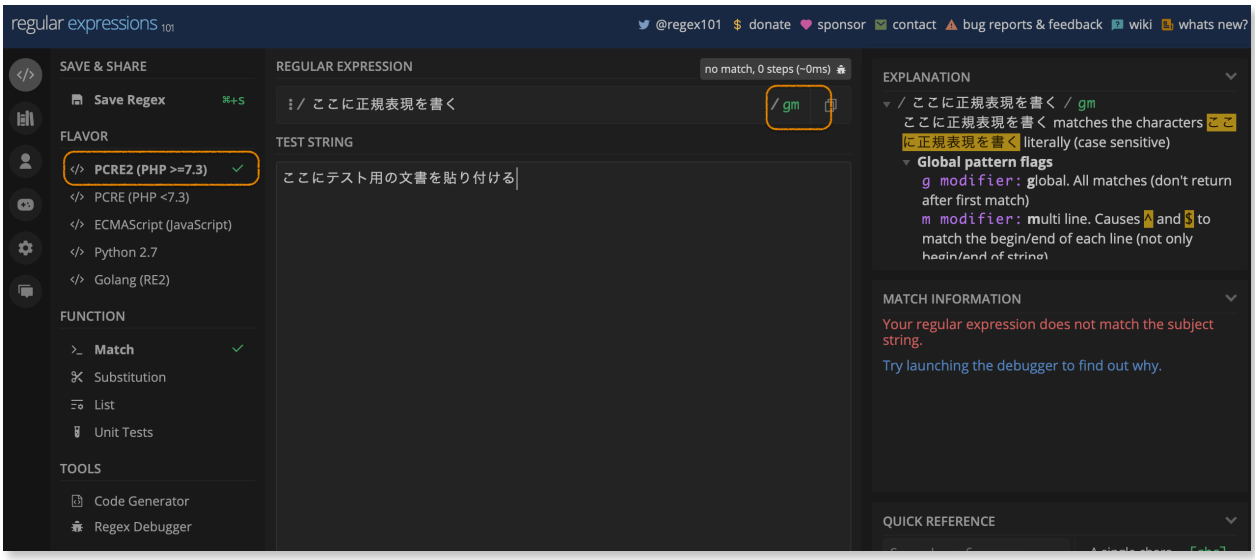
正規表現を自分で書き、どんな結果になるか確かめる。

必要な準備

上記「本日のお願い」❷のサイト「**regex101**（＝正規表現入門）」をブックマークしておく。

今までこのサイトを訪問したことがなければ、デフォルトで以下の設定は済んでいるはず。

念のため、左上あたりの「**FLAVOR**」は「**PCRE2**（一番上）」にチェック、中央上の「**REGULAR EXPRESSION**」窓の末尾にある「オプション」は「**/gm**（プルダウンメニューの上２つにチェック）」になっているか、確認を。これは、挙動をそろえるため。



余裕のある方は、正規表現が使えるエディターで試してもらっても、もちろんよい。ただし、設定や環境の差により、挙動は異なることがある。とくに、部分一致か全文一致か、など。

参考

正規表現が使えるエディターのほんの一例

Atom

https://atom.io/

Visual Studio Code

https://code.visualstudio.com

サクラエディタ

http://sakura-editor.sourceforge.net/

秀丸エディタ

http://hide.maruo.co.jp/software/hidemaru.html

パターンを考えよう

正規表現を一言で表すなら「パターン検索」。素の文字列ではなく「数字が３個＋ハイフン＋数字が４個並んでいるもの」のように、コンピューターに指示を出して、当てはまるものを見つける。お買い物メモみたいなものか。検索に使えるということは、置換にも役立つ。

質問１：このパターンに当てはまるものは何でしょう？

同様に、「03-」だけなら素の文字列だが、「03で始まり、ハイフンや括弧がはさまることがある、10桁の数字」なら、これも立派なパターンだ。

基本構成は、文字の[種類]{数}と位置

パターンの基本は、どんな文字が、いくつ、どこにあるか。

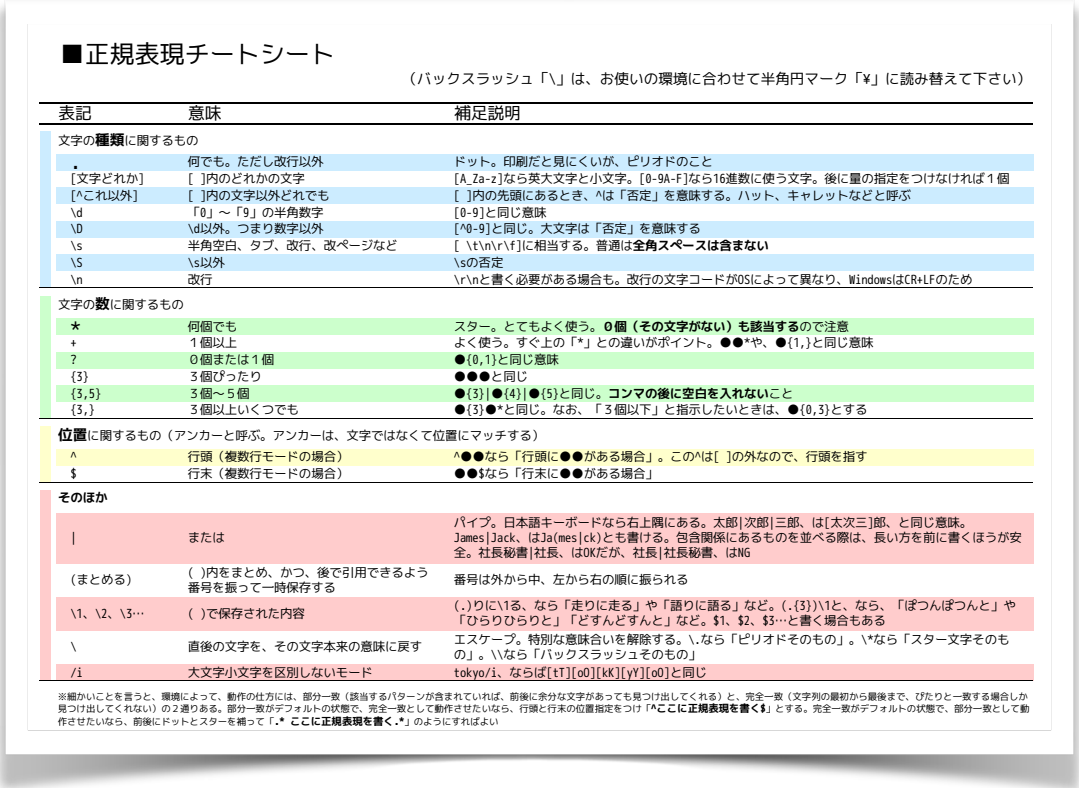
郵便番号は頭に〒や㊮マークがあればやりやすいが、ついていないなら、

数字数字数字−数字数字数字数字

とか

数字×３個−数字×４個

とか。まずは日本語でメモでもしてみて、チートシートを見ながら、置き換えればよい。



■正規表現チートシート

（バックスラッシュ「\」は、お使いの環境に合わせて半角円マーク「¥」に読み替えて下さい）

表記	意味	補足説明
文字の**種類**に関するもの		
.	何でも。ただし改行以外	ドット。印刷だと見にくいが、ピリオドのこと
[文字どれか]	[ ]内のどれかの文字	[A_Za-z]なら英大文字と小文字。[0-9A-F]なら16進数に使う文字。後に量の指定をつけなければ１個
[^これ以外]	[ ]内の文字以外どれでも	[ ]内の先頭にあるとき、^は「否定」を意味する。ハット、キャレットなどと呼ぶ
\d	「0」～「9」の半角数字	[0-9]と同じ意味
\D	\d以外。つまり数字以外	[^0-9]と同じ。大文字は「否定」を意味する
\s	半角空白、タブ、改行、改ページなど	[\t\n\r\f]に相当する。普通は**全角スペースは含まない**
\S	\s以外	\sの否定
\n	改行	\r\nと書く必要がある場合も。改行の文字コードがOSによって異なり、WindowsはCR+LFのため
文字の**数**に関するもの		
*	何個でも	スター。とてもよく使う。**0個（その文字がない）も該当する**ので注意
+	１個以上	よく使う。すぐ上の「*」との違いがポイント。●●*や、●{1,}と同じ意味
?	０個または１個	●{0,1}と同じ意味
{3}	３個ぴったり	●●●と同じ
{3,5}	３個～５個	●{3}|●{4}|●{5}と同じ。**コンマの後に空白を入れない**こと
{3,}	３個以上いくつでも	●{3}●*と同じ。なお、「３個以下」と指示したいときは、●{0,3}とする
位置に関するもの（アンカーと呼ぶ。アンカーは、文字ではなくて位置にマッチする）		
^	行頭（複数行モードの場合）	^●●なら「行頭に●●がある場合」。この^は[ ]の外なので、行頭を指す
$	行末（複数行モードの場合）	●●$なら「行末に●●がある場合」
そのほか		
|	または	パイプ。日本語キーボードなら右上隅にある。太郎|次郎|三郎、は[太次三]郎、と同じ意味。 James|Jack、はJa(mes|ck)とも書ける。包含関係にあるものを並べる際は、長い方を前に書くほうが安全。社長秘書|社長、はOKだが、社長|社長秘書、はNG
（まとめる）	( )内をまとめ、かつ、後で引用できるよう番号を振って一時保存する	番号は外から中、左から右の順に振られる
\1、\2、\3…	( )で保存された内容	(.)りに\1る、なら「走りに走る」や「語りに語る」など。(.{3})\1と、なら、「ぽつんぽつんと」や「ひらりひらりと」「どすんどすんと」など。$1、$2、$3…と書く場合もある
\	直後の文字を、その文字本来の意味に戻す	エスケープ。特別な意味合いを解除する。\.なら「ピリオドそのもの」。*なら「スター文字そのもの」。\\なら「バックスラッシュそのもの」
/i	大文字小文字を区別しないモード	tokyo/i、ならば[tT][oO][kK][yY][oO]と同じ

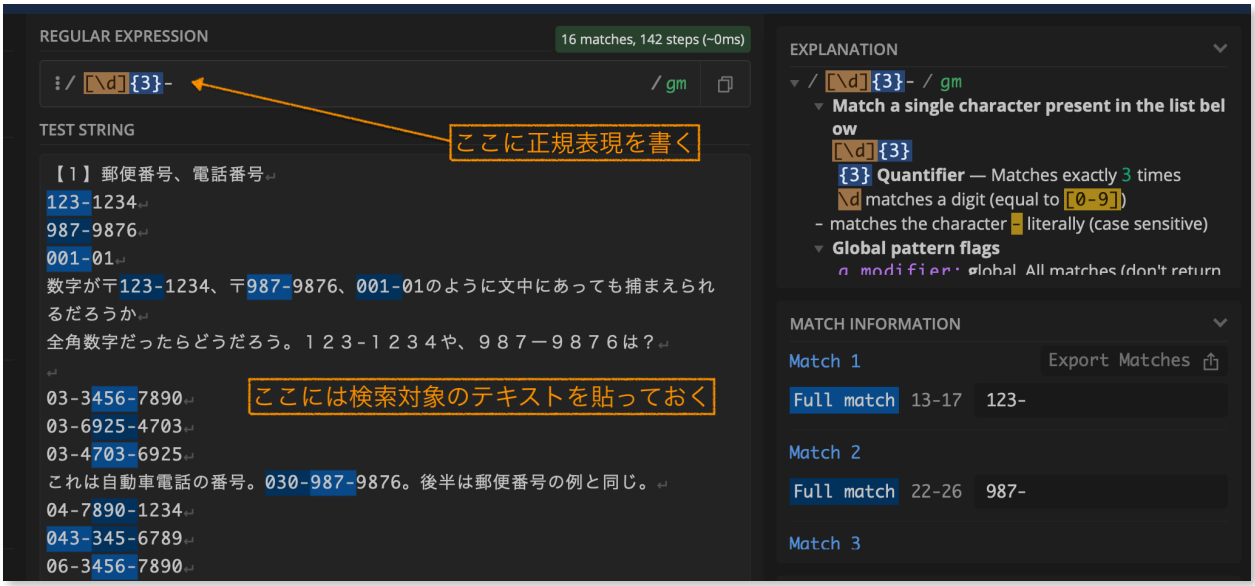
※細かいことを言うと、環境によって、動作の仕方には、部分一致（該当するパターンが含まれていれば、前後に余分な文字があっても見つけ出してくれる）と、完全一致（文字列の最初から最後まで、ぴたりと一致する場合しか見つけ出してくれない）の２通りある。部分一致がデフォルトの状態で、完全一致として動作させたいなら、行頭と行末の位置指定をつけ「**^ここに正規表現を書く$**」とする。完全一致がデフォルトの状態で、部分一致として動作させたいなら、前後にドットとスターを補って「**.* ここに正規表現を書く.***」のようにすればよい

チートシートは、手元において、いつでも眺められるようにしてご参加を

実際に「regex101」で試してみよう

サイトを開き、真ん中あたりにある「TEST STRING」の部分に、検索対象にする文字列を入れる。きょうのところは、こちらで用意した「test.txt」というファイルの内容を、練習用にコピー＆ペーストして下さい。お手元の貴重な取材成果をアップしないように。

試したい正規表現は、上にある横長の窓みたいな「REGULAR EXPRESSION」の欄にタイプする。面倒ならば、このPDFファイルから貼り付ければよい。でも、考えながら打ち足していくと、それに応じて「マッチ」の状況が変わっていくのが楽しいし、うまく行かないときの試行錯誤を助けてもくれる。さて、ターゲットを上手に捕まえられるだろうか？



色がついた部分が、指定したパターンに当てはまる箇所。正規表現を入力していくにつれ、マッチング状況が刻々と変わる

数字、と指示したいときは、[0-9]または[\d]、と書く。これを３個並べてもよし、×３の意味になる{3}としてもよい。ハイフンは、素の文字だから、そのままハイフンを書く。後半は数字の個数のみが違う。なお、個数はぴったりだけでなく、以上や以下、といった指定もできる。以下、はちょっと工夫が必要だけど。

なので、

① [0-9][0-9][0-9]-[0-9][0-9][0-9][0-9]
② [0-9]{3}-[0-9]{4}
③ [\d][\d][\d]-[\d][\d][\d][\d]
④ [\d]{3}-[\d]{4}

などが、郵便番号を見つけ出す正規表現として考えられる。〒マークを目印にできないと、ほかのものもひっかかってくるかもしれないが、絞りすぎて漏れるより、広めに緩く、が安全。

質問２：自分で使うなら①〜④のどれにしますか

しかし、試してみたところ、郵便番号よりも多く数字が並んでいる「1234-56789」にもマッチしてしまった……。これを除外することができないか？　全角数字だったらどうなる？　見逃してしまうのか？　もっともなニーズだが、30分しかないので、先を急ぎます。

文字は「××以外」という指定も可能

`[ ]`で囲むと、その中の文字どれか１つ、を表す。具体的な文字でなく、今使った数字の「`\d`」のように、「こんな種類の文字」という書き方も用意されている。これが便利。

単に列挙するだけでなく、文字コードの並び順を使って`[A-Z]`（英大文字）や`[ぁ-んー]`とも書ける。後者は平仮名をだいたい捕まえようと意図したもので、先頭は「あ」ではなく拗音の「ぁ」であることに注意。横棒に見えているもののうち、短い方はハイフン（字の範囲を指定）で、長い方は長音符だ。でも、漢数字は洋数字と違って文字コードの並びが連続していないので、通常の環境では、`[〇一二三四五六七八九]`のように使われている字を列挙しないといけない。十・百・千・万・億や、壱弐参のような大字(だいじ)が使われているなら、それも書く。大変。

`[^ABC]`とすると、ABC以外の文字、を指す。`[ ]`内の先頭にある「`^`」は否定の働きをする。数字「`\d`」に対して、数字以外「`\D`」、スペースやタブなど見えない文字「`\s`」に対して通常の文字「`\S`」など、小文字を大文字に変えると否定の意味になるものもある。

環境によっては、文字種の指定に`[\p{Hiragana}]`で平仮名、`[\p{Han}]`で漢字、とか、`[:upper:]`で英大文字、`[:alnum:]`で英字と数字（怪しい）、といった書き方も用意されているが、きょうは触れない。基本をはみ出る部分は、使うときに慎重に調べたほうがよい。

＊や？は「ワイルドカード」と違う使い方。ドット「.」も大事

自分のパソコンでファイルを検索するとき、「`*.jpg`」（jpg形式のファイル全部）とか、「`image-?.png`」（？部分に１字入る名前のファイル全部）などと入力したことがあるのでは？　これで見つけられるのは、OSに備わっているワイルドカード展開機能のためだ。

ただ、展開では、おなじみ「`*`」は「任意の文字列＝どの文字でもいくつでも」の意味だし、「`?`」は「任意の文字＝どれでも１文字だけ」を意味している。これに対し、正規表現では、「`*`」は「直前に指定した文字が０個以上」と、数のみを指定する働きをする。「`?`」も、正規表現では、「直前の文字が1個あるか、ないか、そのどちらか」の意味。やはり、文字の種類ではなく、数だけを指定する機能を持つ。同じ記号を使うため、迷うかもしれない。きょうはワイルドカード展開のことはいったん忘れて、チートシートを眺めてほしい。

正規表現では、任意の文字列を意味するのは「`.*`」か「`.+`」になる。「`.`」がどの文字でも、という種類の指定の部分。その後に数の指定をつけて、「`.*`」は０個以上（つまり、文字がない場合も該当する！）、「`.+`」は１個以上（必ず１個はある）ということになる。

「*」や「.」や「\」の特別な機能を解除し、素の文字として使う場合は「\」をつける。これを「エスケープする」という。スターそのものは「*」、バックスラッシュそのものは「\\」とする。

表記の揺れに対応できる ｜または

正規表現を使ったパターン検索が役に立つのは、素の文字だけの検索に比べ、強力だから。

情報公開請求をして、サイトウさんに関する文書がどさっと出てきたとしよう。スキャンした文書をOCRにかけた場合や、最初から電子データだったとしても作成時期や入力者がまちまちだった場合には、「斎藤」「齋藤」「斉藤」「齊藤」など表記の揺れがあるかもしれない。

数値データでも同様。ゼロが英字のオーになってしまっていることだってある。数字の 1 だと思ったら、英字のI（大文字のアイ）やl（小文字のエル）が混ざっているかもしれない。こんな場合も、パターンを使えば、工夫次第で検索できる。

このように「または」と指示したいときは、「|（パイプ）」記号を使う。日本語キーボードの場合は、右上のあたりにある¥マークのキーを、「SHIFT」を押しながら一緒に押すと出る。

練習３：上記の４種類の「サイトウ」さん全部が検索できるパターンを作って下さい

出題しておきながら、すぐ答えが見えてしまっては申し訳ないので、以下の解答例は白い字にした。もちろん、範囲指定で選択してコピーし、どこかに貼り付ければ、読める。

①

②

③

質問４：自分で使うとしたら、どれにしますか

字形が似ているために起きること

表記の揺れは、悲しいぐらいにある。社内むけの記事データベースを検索してみるとよい。OCRや手入力の場合もそう。

一見、どれも同じ「ヘリコプター」に見えても、以下のようなことがある。

・カタカナの「へ」ではなくて平仮名の「へ」になっている
・カタカナの「リ」ではなくて平仮名の「り」になっている
・半濁点の「プ」ではなく、濁点の「ブ」になっている
・カタカナの「タ」ではなく、漢字の「夕」になっている
・音引き「ー」ではなく、マイナス記号「－」になっている

練習５：以上５種の各種「ヘリコプター」を全部検索できるパターンを考えて下さい

解答例
[へヘ][りり]コ[プブ][タタ][ーー]?

　見た目で区別がつかないので、なかなか説明しにくいが、
・最初の[　]の中はカタカナと平仮名の「へ」
・次もカタカナと平仮名の「リ」
・コはそのまま
・半濁点と濁点つきの「フ」
・カタカナの「た」と、漢字の夕方の「ゆう」
・音引きがあってもなくても捕まえるために、「?」。0個か1個、という意味
という構成になっている。

検索したら、置換も削除もできる

　では、一括して修整してみよう。平仮名をカタカナに、へ、り、と 1 文字ずつ置換していけばいいんでしょ、と思うかもしれない（思わないかもしれない）が、そうすると、**へり**くつも「**へり**くつ」になってしまう。文書中の全部の助詞「へ」がカタカナになってしまったら大ごとだ。やはり、ヘリコプターを意図して書かれた語だけを捕まえるのが、よい。
　実際にはエディターなどで作業するだろうが、今はこのまま **regex101** で試してみる。
　上の検索窓には、上記の正規表現を入れたままにしておく。そして、左のほうにある **FUNCTION** の欄は、 **Match** にチェックがついているはずだが、一つ下の **Substitution** にチェックをつけてみよう。置換の窓が開く。そこに、正しく「ヘリコプター」と入力してやると、置き換えた結果が下に表示される。

SUBSTITUTIONの窓が開いた。ここに正しく「ヘリコプター」と入れてみると……

置換の際、窓に何も入れず、「空文字（何もなし）に置き換える」ように指示を出せば、それは「削除」ということになる。不要な部分、邪魔になる部分を検索で捕まえてこうすれば、たとえば、ある県内の住所と分かっている場合に県名と郡名は削る（市町村から始める）とか、単位付きの数字から数値のみを抜き出すような場合に使えるだろう。

注意点は３つ。①一度に片付けようとしないこと。何回かに分けてやるほうが、無理がなくて楽なことが多い。②逆転の発想を。必要な部分を探すパターンより、その裏返しで、不要な部分にマッチさせて削るほうが容易なときもある。③すべて正規表現でやろうとしないこと。エディターや表計算ソフト、コマンドライン（前回やりましたね）の機能も、遠慮せず合わせ技で使えばよい。正規表現はあくまで脇役。使えるナイフが１本増えた、という感覚で。

^位置指定。行頭にあるか行末か$

行頭にそのパターンが来る場合には「^」、行末なら「$」で位置を指定することができる。行頭の斎藤さんなら「^斎藤」（この場合は[ ]の外なので、否定ではない）、行末に数字が2つ以上なら「\d{2,}$」とする。位置指定で絞り込むことで、検索の精度が上がる場合には、使ってみよう。もし、もう時間切れだったら、後ほど自分でやってみて下さい。

(まとめて再利用する)括弧の機能

( )には、普通の数式の計算と同様、まずそこを優先する・まとめる、という機能がある。

また、( )で囲んだ場合には、パターンに一致したものに内部的に番号が振られて記憶され、それを正規表現中で使うことができる。

① (.)りに\1る
② (.){3}\1と
③ (ぐ[りら])と\1

で検索してみると、①はたとえば、「喋りに喋る」「滑りに滑る」、②は「ゆらりゆらりと」「ぽたんぽたんと」などにマッチする。③は「ぐりとぐり」「ぐらとぐら」にはマッチするが、「ぐりとぐら」にはマッチしない。「ぐ[りら]とぐ[りら]」や「ぐ.とぐ.」であれば、マッチする（「ぐらとぐり」にも）。

重要。(ここだけほしい)ときも括弧を使う

順番に説明してきたので、終盤になって申し訳ないが、重要なポイントなのでご辛抱を。探すものを特定するためにパターンを考えたが、取り出して使いたいのは実はその一部分だけ、というケースにも、( ) は役に立つ。

選手データの身長の項目だけを使いたい、というときに、「身長[\d.]+cm」（この場合の「.」は文字通りの小数点。律儀にエスケープして「身長[\d\.]+cm」でもよい）とすると、確

かに身長が検索できる。でも、「身長」や「cm」は、これが体重や胸囲ではなく身長のデータだ、と特定するのには有効だったけれど、間違いなく身長でありさえすれば、本当は数値だけでいい。数値だけがいい。こういうときは、そこだけを( )で囲む。以下のようになる。

```
身長([\d\.])+cm
```

身長の数値だけが、別の色でハイライトされたはず。

ここで、確認のために、左のFUNCTION のチェックを、List に変更してみよう。そうすると LIST の窓が開くので、「$1」と入力する。これで、いま捕まえたもの＝身長の数値だけ、がずらりと並ぶ。

先の「ぐりとぐり」のように、( )で捕まえた部分をその正規表現の中で再利用する場合は「\1」と書いたが、別の場所で使う際には「$1」とする記法のようだ。このあたりは、使うときに改めて調べてみてほしい。

さて、全員分がくっついて 1 行になってしまっているので、これでは役に立たない。そこで、改行を意味する「\n」を添えて、LIST の窓に

D選手って誰です？

```
$1\n
```

と入れてやると、きれいに整形される。

Substitution なら、「$1」とだけ入れれば、行末の改行コード（という文字）はそのまま残っているので、身長の入っている行だけが、この LIST 同様に数値だけに変換される。ほかの行はそのままの状態で残る。

順番を入れ替える

この応用で、( ) を複数組使って順番の入れ替えもできる。「2021年2月27日」というスタイルの日付を「27/02/2021」のように日月年に並び替える（ついでにスラッシュ区切りにして、月と日は２桁に揃える）ためにはどうすればよいか、考えてみよう。FUNCTION はまた、Substitution に戻しておく。検索の窓は、数字を捕まえるために、こう入力する。

```
(\d*)年(\d*)月(\d*)日
```

( )で捕まえて一時保存されたものは、外から内、左から右、の順に内部的に番号が振られて、年が１、月が２、日が３、となっている。これを活用して、置換の窓には３２１の順に並べて

やればよい。区切り文字もスラッシュに変えたいのだから

```
$3/$2/$1
```

とする。

ここで、下の **SUBSTITUTION** の出力結果を、上の **TEST STRING** の窓に張り直して、２段階目に取りかかる。

今度は、月と日の数字が１桁だったら２桁にする部分。条件をほぐしながら、日本語で考えていこう。まず、１桁の数字とは「すぐ後にスラッシュがある」数字。月と日だったら、は「行頭（日）かスラッシュの直後（月）」ということだ。２桁に揃える、とは「前に0をくっつける」に相当する。

月と日で場合分けして、まず行頭（日）からやる。**regex101** の検索窓ではスラッシュもエスケープしないといけないようだから、検索窓に入れるのは

```
^\d\/
```

とし、置換窓に入れるものは

```
0$0
```

とする。ここで初めて出てきた「`$0`」は、検索パターンに合致したもの全体をそのまま受け取る器で、`(  )`を使っていない場合にも有効だ。行頭に数字が１文字だけあって、その次がスラッシュの場合は、頭にゼロをつける、と指示していることになる。

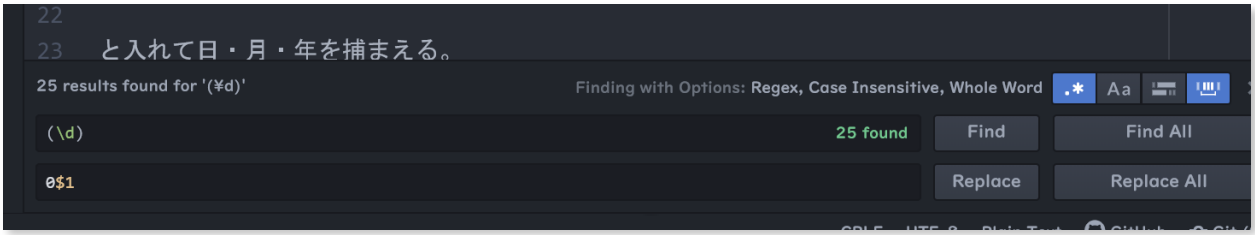
次に、月を処理する。今度は行頭ではなく、途中に0を補うので、`(  )`を使う。これまでの置換結果をテスト文字列の欄に張り直した上で、１桁数字を捕まえるため、検索窓には

```
\/(\d\/)
```

と入れる。置換窓には

```
/0$1
```

と入れる。月で１桁、と絞り込むために使った前のスラッシュを入れ直している。完了。



ATOMの検索・置換画面。「.*」を押して青く反転させると、正規表現がオンになる。さらにその右、FIND ALLの上の青いボタンが、WHOLE WORD（単語単位での検索）モード

深入りはしないが、単語境界「`\b`」と、先読み（**lookahead**）が使える場合は、後段の月日のゼロ埋め処理が簡単になる。検索「`\b(\d)(?=\/)`」または「`\b(\d)(?!\d)`」、置換

「0$1」で、一度に済ませることができる。日か月で1桁、と特定するためのパターンが重なり合わないので、場合分けせずに済むからだ。**regex101** でも試せる。**ATOM**のように、検索・置換時に単語単位（**Whole Word**）モードが用意されているエディターもある。

郵便番号ふたたび

先ほど考えたうち、郵便番号の④の

```
\d{3}-\d{4}
```

をベースに、どう改良すればよいか、試行錯誤してみることにする。

きっちり3桁+4件になっているものだけを捕まえるようにしたい、のだった。なので、数字の前後には「数字以外のもの」が来ている、という条件をつけてみよう。数字の否定なので、「^\d」または[^0-9]。大文字にした「\D」でもよい。ここまでで、

```
\D\d{3}-\d{4}\D
```

となる。ただ、試してみると、行末の場合はこれで対応できる（改行コードがあるから。文末を改行していない場合はダメ）のに対し、行頭にある場合は、この書き方ではマッチしないことが分かる。

そこで、行頭にあるか、または数字以外の文字に続く、としてみよう。これは(^|[\D])と書けばよいから、ここまでで

```
(^|\D)\d{3}-\d{4}\D
```

となった。

加えて、前後にさらにハイフンが続く場合は郵便番号以外のもの（たとえば電話番号）なので、これも除外するため、数字とともに否定する。否定の対象に「-」が加わるので、大文字を使った\Dではなく、キャレット「^」を使った方が書きやすい。(^|[^\d-])とすればよいだろう。ここで、一番目のキャレットは行頭を意味するアンカーなのに対し、二番目は否定の意味であることによく注意してほしい。この部分は、「行頭にあるか、直前の文字が数字でもハイフンでもない場合のどちらか」ということになる。

そして最後、郵便番号だと特定するために、前後に何があるかを条件に検索したが、そこは本当は邪魔。なので、**XXX-XXXX**の部分だけを取り出すために、( )で囲もう。

……という具合に、**regex101** で段階を追って試しながら書いた正規表現が、下記。

```
(^|[^\d-])([\d]{3}-[\d]{4})[^\d-]
```

見てもなんだか分からん、と感じるかもしれないが、サイトで実際に試しながら考えていくと、パズルを解くような感じで、意外となんとかなる。あとは、置換窓に「$1」。

なお、この種のサイトは動作確認のテスト用。当たり前だが、秘匿すべき資料を貼って処理するのは厳禁だ。それは、自分の手元なり、社内のパソコンでやること。

全角と半角

これ以上はしつこくなるのでやめておくが、郵便番号の例で全角で書かれたものにもマッチさせるなら、両方を併記して、数字は「(\d|[０-９])」、ハイフンは「[-ー]」とすればよい。でも、最初に別のツールで、すべて半角に変換しておくほうが合理的かもしれない。

なお、**regex101**のサイトでは、「**REGULAR EXPRESSION**」の窓のオプションに、小文字の**u**も足して「**/gmu**」（UNICODEのu）とすると、数字**\d**やスペース**\s**が全角にもマッチするようになる。

補足。「社長|社長秘書」か「社長秘書|社長」か

マッチングは、左から右へなるべく長く、と働く。複数のパターンを組み合わせた場合、気をつけないといけない点だ。また、たとえグローバル（１行に何回出てきてもマッチさせる。**gm**の**g**はこの意味）モードでも、パターンが重なってしまっていると、意図したとおりにならない。A<—>Aと、b<—>bに合致するパターンが、入れ子になっていなくて、A<—b<……>A—b>のように続いている場合、……の部分はAが取る。なので、bのパターンにはマッチしない、と判定される。東京都は東京・京都の両方にはマッチせず（京の字が重なり合っている）、東京にしかマッチしてくれない。１桁の日付のゼロ埋めで、場合分けしたのは、このためだ。

なお、**gm**の**m**はマルチライン（複数行）。本来は文書全体の先頭と最後を指す「^」と「$」を、各行の行頭と行末にマッチさせる設定だ。記者が取材ツールとして使う状況を想定して、きょうはこのモードで練習した。もしも、あなたが開発担当部門に転勤になり、入力された文字列のバリデーションなどに正規表現を使うときには、セキュリティ上の問題を考えて、「**\A**」「**\z**」を選ぶこと。あと、そうなったら、記者ゼミの講師に来て下さい。

一方、「|」（または）で並べたものが包含関係にある場合は、長いものから順に書いておかないと、全体を拾ってくれない、ということも起こる。「社長|社長室|社長室付」と書いてしまうと、「社長室付になったよ」という文字列のうち、最短の「社長」にしかマッチしない。**OR**というロジック処理を、できるだけサボって効率を上げるのが、コンピューターの基本思想だからだ。最初のもので真偽判定が済んだ（マッチした＝**TRUE**だ）なら、そこから先を調べる必要はありませんよね、というお作法。そういうものだと心得て、この場合には正規表現を「社長室付|社長室|社長」と書くようにする。エライ順ではない。

見出しのケースなら、「社長秘書|社長」が正解。「社長(秘書)?」と書いても正しい。

お疲れさまでした！

はるばる遠くまで来た感じですね。手を動かしてみても分からなかったところのフィードバックは、今後の皆の財産になります。お問い合わせ、後日のフォローなどは、ご遠慮なく
yoshito.nishio+JNPC@gmail.com
までお願いします。もっとシンプルな解法を見つけた方もぜひ、教えて下さい。

最後の質問：最初のページにある、このテキストのタイトルは、何にマッチするでしょうか

参考資料

［１］「改訂新版 正規表現ポケットリファレンス」
（宮前竜也 著、技術評論社）
［２］「正規表現 技術入門 最新エンジン実装と理論的背景」
（新屋良磨・鈴木勇介・高田謙 著、技術評論社）
［３］「はじめての正規表現とベストプラクティス」
（https://techracho.bpsinc.jp/hachi8833/2020_11_12/62948）
［４］「オレンジ工房 ORANGE-FACTORY　UTF-8の文字コード表」
（http://orange-factory.com/dnf/utf-8.html）

［１］は基本的な内容の書籍。処理系による違いが整理されている。
［２］は網羅的な内容の書籍。正規表現の歴史や内部的な仕組み、数学的背景にも触れている。何でも載っている頼もしい本だが、その分、かなり難しい。
［３］は親切に書かれているサイトだが、内容は高度。Rubyを使わない人でも、参考になる。ただ、正規表現を自分で使ってみてからでないと、難しい。
［４］は見やすい文字コード表を載せている。平仮名は３バイト文字のところにある。

改版履歴

p.11　下から５行目の正規表現、ハイフンの位置が不統一だったので整理
p.12　上から２行目の文中の正規表現、「\d」まで全角になってしまっていたので半角に修正
(21/02/27講義終了後)

p.8　上から６行目、「逆転の発想」の誤字を修正
(21/02/28)

p.12　補足を書き直し。パターンのオーバーラップとして東京都の例を足し、包含関係の例も直した。PCREやATOMの場合、後ろに文字が増えていく例はこの通りの挙動になったが、社長|支社長のように前に増えた場合には、問題なく補足されることに気づいたため。なお、最左最長はDFA型、包含関係はバックトラックVM型のエンジンに特有なのであれば、処理系によってくっきり差が出る部分ということになるのだが、確かめ切れていない
(21/03/02)

p.7　「検索したら」の小見出しから下に４行目、ヘリコプターを「想定して」は変なので「意図して」に直す
p.8　「まとめて再利用」の小見出しから下に９行目、ぐりとぐらにマッチする正規表現の例から( )を抜いた。直前に出てくる例との対比のためだったが、まとめたり、キャプチャーしたりしておらず、本来は不要なため。あってもマッチはする
(21/03/30)