６日目：データをチェックする

今日は，読み込んだデータのチェックをしてみましょう。また，４日目にやったように，特定のデータだけを抽出して別のデータ（変数）を作ることもやってみたいと思います。これが，あとで結構役に立つのです。

R が起動したら，前回のように作業ディレクトリを変更し，保存した R エディタを開いてください。そこにファイルを読み込む命令が残っているはずなので，それを使ってデータファイルを読み込んでください。

では，読み込んだデータ（x という変数名にしたとします）を簡単にチェックするいくつかのコマンドを紹介します。

・すべてを表示させる

```
x
```

・表の上から３行分だけを表示させる（head(x)にすれば自動的に５行分）

```
head(x,3)
```

・表の下（最後）から３行分だけを表示させる（tail(x)にすれば自動的に５行分）

```
tail(x,3)
```

・変数名（列に付けられた名前）を表示させる

```
colnames(x)
```

・列数を求める（表示させる）

```
ncol(x)
```

・行数を求める（表示させる）

```
nrow(x)
```

全体的なチェックに使えるのはこのようなものでしょう。

次に，部分的にチェックするのに使えるコマンドを紹介します。引き続き，データのファイル名は x です。

・データ（x）の中の，ある変数名のデータだけを求める（表示させる）（※「年齢」の場合）

```
x$年齢
```

・特定のセル入っているデータを求める（表示させる）（たとえば，３行３列目なら）

```
x[3, 3]
```

・特定の行に入っているデータを求める（表示させる）（たとえば，２行目なら）

```
x[2, ]
```

・特定の列に入っているデータを求める（表示させる）（たとえば，３列目なら）その１

```
x[ , 3]
```

・特定の列に入っているデータを求める（表示させる）（たとえば，３列目なら）その２

```
x[3]
```

なお，上のコマンドは，見やすくなるように半角空白を入れている部分もあります。この半角空白は入れなくても問題はありません。

特定のセルの表示，特定の行，特定の列と見比べると，列番号，行番号をはぶくと，その行，もしくは列全体が表示されるわけです。少し違うのは，x[3]という列表示です。x[, 3]と両方やってみるとわかりますが，x[, 3]は数値を横に表示し，列名は表示されません。x[3]は数値を縦に表示し，変数名が表示されます。

```
> x[2, ]
    no 性別 年齢 b1 b2 b3 b4 b5
2 1002    2   20  2  2  4  3  2
> x[ ,3]
 [1] 20 20 19 21 20 19 20 19 19 19 19 19 19 19 19 19 19 19 19 20
> x[3]
   年齢
1    20
2    20
3    19
4    21
5    20
6    19
7    20
8    19
9    19
10   19
11   19
12   19
13   19
14   19
15   19
16   19
17   19
18   19
19   19
20   20
```

次に，複数の行や列，セルを抽出して，そこだけをチェックすることを考えてみます。４日目にやった，あるベクトルから任意のものを抽出したときのことを思い出してください。その時のやり方を援用すれば，いくつかの方法が考えられます。

まずひとつは，抽出する複数の行や列を「:」でつないで入力することが考えられます。以下のような指示は計算してくれるようです。

```
x[3, 3:5]
x[2:4, ]
x[ , 3:5]
x[3:5]
```

実行した結果を眺めてもらえれば一目瞭然ですが，上から，「３行目の３列目から５列目までのデータ」，「2 行目から４行目までのデータ」「３列目から５列目までのデータ」「３列目から５列目までのデータ」が表示されます。また「３列目から５列目までのデータ」である x[, 3:5]と x[3:5]は，複数列指定の場合は全く同じ出力結果となります。

ところが，このような行列の番号ではなく，変数名を使って複数列をチェックしたい，というのは，ちょっと面倒です。ある変数のみであれば，x$年齢でよいのですが（これは，ファイル名と変数名を「$」で結ぶことで指定している形式），この形式では x$年齢:b2 といった指定の仕方は受け付けてくれません。以下のように，c でくくってズラズラと並べないとダメなようです。なお，文字列をデータとして認識させるためには，「"」でくくってやります。

```
x[c("年齢","b1","b2")]
```

しかし，私とかだと，このようなものをいちいち書くのは面倒だ…と思ってしまうわけです。何とか，もっと簡単にできないか…

そこでこれを，３列目のデータを表示させる x[3]という極めて簡単なコマンドと比較してみます。すると，3 のかわりに，c("年齢","b1","b2")がはいっているということがわかります。逆に言えば，c()で変数名のリスト（ベクトル）を作り，たとえばｖという名前で保存しておけば，x[v]でそのリスト内のデータが表示されるということになります（要は，x[3]の列番号3の部分を，作成した変数リストｖに置きかえたものがx[v]というわけです）。

このような変数名を抽出してリストにしておくことは，今後いろいろな時にかなり役立ちますので，覚えておいてください。

ところが，v <- c("no","性別"…などと一から入力するのは，変数が多くなるととても手間…　なので，簡単に作る方法を考えます。

先に，colnames(x)で変数名が出力されることを紹介しました。その際，変数名は"no"というように表示されたはずです。この出力を使えば，変数名を書き，さらに「"」でくくるといった作業をしなくてすみます。

そこでまず colnames(x)で変数名を出力させます。その結果を，コピー&ペーストで R エディタに貼り付けます。こうしておいて，後はそれを加工すればよいわけです（メニュー

バーの「編集」の中にある，検索，置換をうまく使えば，さらに簡単！）。

```
> #"no" "性別" "年齢" "b1"   "b2"   "b3"   "b4"   "b5"
> #これをもとに，空白を消してカンマを入れ，c()でくくると…
> v <- c("no","性別","年齢","b1","b2","b3","b4","b5")
> #このような変数リストを簡単につくることができます。
>
> #さらに，これを使って必要な変数を追加したり残したりすれば，いろいろな変数リストができます。
> v2 <- c("b1","b2","b3","b4","b5")
> v3 <- c("性別","b2","b4","no")
```

上の図のvだけを残しておけば，いつでも簡単に必要な変数だけを取り出した新しいファイルを作成することができます。

ちなみに，ここでx[v3]と命令してみると…

```
> x[v3]
   性別 b2 b4   no
1     2  5  4 1001
2     2  2  3 1002
3     2  4  4 1003
4     2  3  4 1004
5     1  5  5 1005
6     1  3  5 1006
7     2  5  4 1007
8     2  3  3 1008
9     2  4  4 1009
10    1  4  3 1010
11    1  3  4 1011
12    1  4  3 1012
13    2  3  4 1013
14    2  3  4 1014
15    2  5  5 1015
16    2  5  5 1016
17    2  3  3 1017
18    2  4  4 1018
19    1  4  4 1019
20    2  5  5 1020
```

当然ですが，このような出力となります。

さて，このv3ですが，もちろんその中身は，変数名（だけ）です。それらの列のデータを伴っているものではありません。x[v3]というシンプルな命令を見ていると，x[v3]という名前のデータがあるかのようにも思えてきますが，これは「xからv3にある変数名の行を取り出せ」という命令なのです。v3で指定された変数名だけからなる新しいデータファイルを作るには，もう一手間かけておく必要があります。一手間といっても，以下のように新しい変数名を用意し，x[v3]の結果を代入するだけですが。

```
x3 <- x[v3]
```

こうしておいて x3 の中身を見ると，次の図のようなデータファイルになっていることが確認できるはずです。

これで6日目は終了です。Rエディタの中身は保存しておいてください。
（今日使ったコマンドでたくさん遊んでみるといいと思います。特に行や列に馴染みのない人は，これらのコマンドに慣れるのに少し時間がかかると思うので…）

```
> v3
[1] "性別" "b2"   "b4"   "no"
> x3 <- x[v3]
> x3
   性別 b2 b4   no
1     2  5  4 1001
2     2  2  3 1002
3     2  4  4 1003
4     2  3  4 1004
5     1  5  5 1005
6     1  3  5 1006
7     2  5  4 1007
8     2  3  3 1008
9     2  4  4 1009
10    1  4  3 1010
11    1  3  4 1011
12    1  4  3 1012
13    2  3  4 1013
14    2  3  4 1014
15    2  5  5 1015
16    2  5  5 1016
17    2  3  3 1017
18    2  4  4 1018
19    1  4  4 1019
20    2  5  5 1020
```