

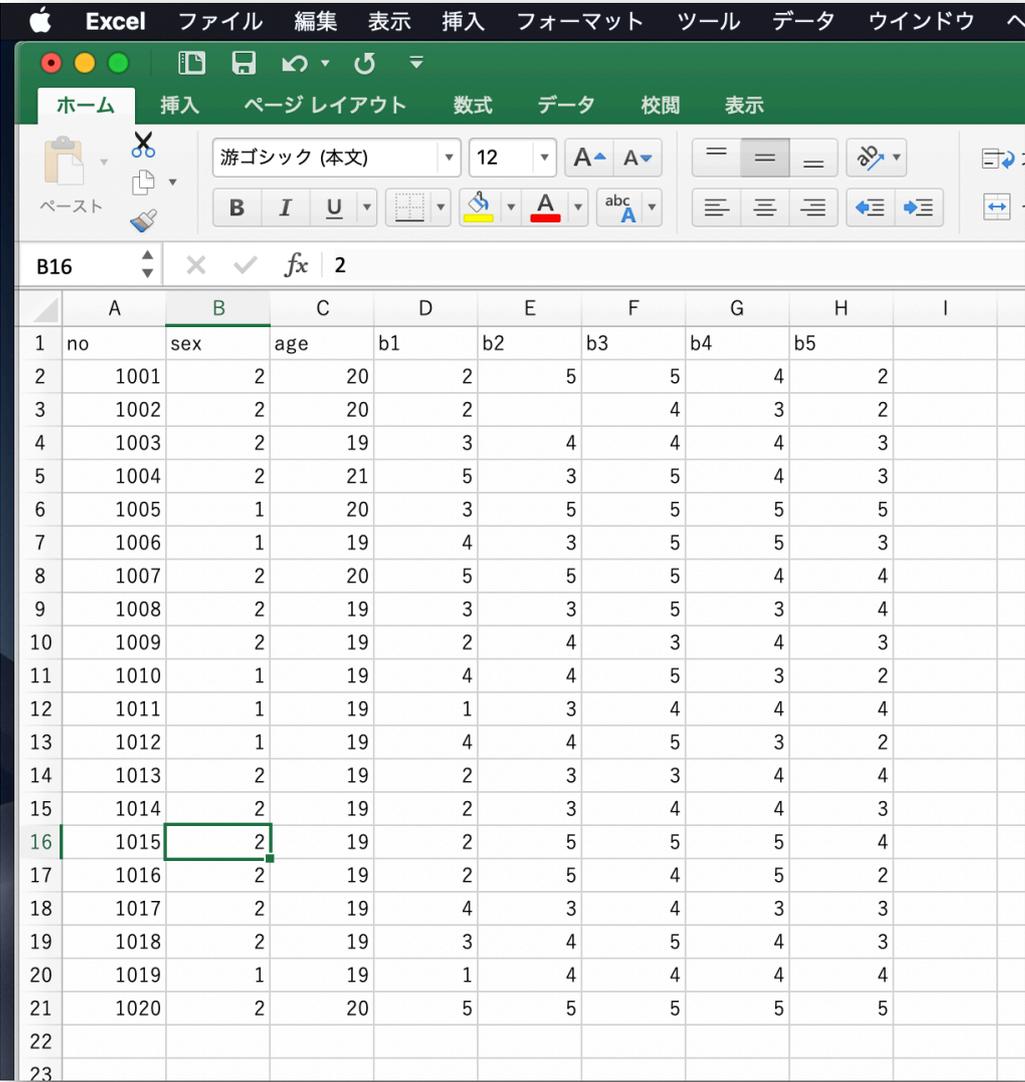
## 5日目：エクセルでデータを準備し、読み込む

これまでは、データ分析とほど遠い感じの内容だったかもしれませんが、今日からは、データ分析のイメージに近い内容になってきます。今日は、エクセルで入力したデータをRに読み込ませてみたいと思います。

もちろん、集めたデータを直接Rに入力していくこともできますが、現実的ではないと思います。まずはエクセル(などの表計算ソフト)に入力して、それをRに読み込ませるのが適当でしょう。

エクセル上でのデータ入力は、特にRのために留意する点はないように思います。1行目冒は変数名を入力し、次の行からをケースとします。項目名を付ける際には、4日目でも触れた入れ物の名前の注意事項を参照してください。欠損値は空欄にしておけばよいです。

今回は、下図のようなデータを例にしましょう。エクセルに入力してください。



	A	B	C	D	E	F	G	H	I
1	no	sex	age	b1	b2	b3	b4	b5	
2	1001	2	20	2	5	5	4	2	
3	1002	2	20	2		4	3	2	
4	1003	2	19	3	4	4	4	3	
5	1004	2	21	5	3	5	4	3	
6	1005	1	20	3	5	5	5	5	
7	1006	1	19	4	3	5	5	3	
8	1007	2	20	5	5	5	4	4	
9	1008	2	19	3	3	5	3	4	
10	1009	2	19	2	4	3	4	3	
11	1010	1	19	4	4	5	3	2	
12	1011	1	19	1	3	4	4	4	
13	1012	1	19	4	4	5	3	2	
14	1013	2	19	2	3	3	4	4	
15	1014	2	19	2	3	4	4	3	
16	1015	2	19	2	5	5	5	4	
17	1016	2	19	2	5	4	5	2	
18	1017	2	19	4	3	4	3	3	
19	1018	2	19	3	4	5	4	3	
20	1019	1	19	1	4	4	4	4	
21	1020	2	20	5	5	5	5	5	
22									
23									

エクセルでこのようなデータを入力できたら、とりあえず保存しておきます。次にこれをCSV形式で保存します。エクセルのメニューバーから「ファイル」を選び、「名前を付けて保存」を選びます。

保存画面の下部に、「ファイル形式」の選択がありますので、ここでCSV形式を選択します。CSVの形式に「CSV (コンマ区切り) (.csv)」と「CSV UTF-8 (コンマ区切り) (.csv)」がある場合は、今回は「CSV (コンマ区切り) (.csv)」を選んでください。そして、3日目に作成したRの練習のための専用フォルダに保存してください。

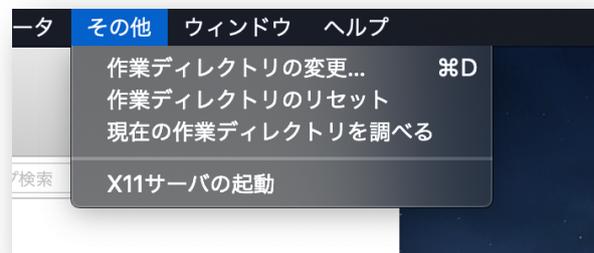
名前は適当につけてかまいませんが、とりあえず今回は「sample.csv」にしておきたいと思います。

CSVという形式はエクセルの標準フォーマットよりも情報量が少ないので、注意喚起の質問が出てきますが、「はい」(もしくは「続行」)をクリックしてください。

保存ができれば、今度はRでこのファイルを読み込んでみます。

Rを起動してください。起動したら、まずは作業ディレクトリを変更します。ディレクトリとは、フォルダと同じようなものだと思ってください。Rにデータが置かれている場所を伝え、作業をする場所を設定するというイメージです。

Rのメニューバーから「その他」を選びます。すると右図のような選択肢が出てくるので、「作業ディレクトリの変更」を選びます。



すると、フォルダを指定する画面が出てきますので、先にデータファイルを保存した、Rの練習用のフォルダを選び「開く」をクリックします。

これで作業ディレクトリの変更は終わりですが、Rコンソールには何のメッセージも出ません。そこで、ちゃんと作業ディレクトリがデータファイルの置いてあるフォルダに設定されているかを確認してみます。

先と同様に、Rのメニューバーから「その他」、そして「現在の作業ディレクトリを調べる」を選びます。するとRコンソールにメッセージが出てくるでしょう。私の場合だと、以下のようになります。

```
> getwd()
```

```
[1] "/Users/urakami*****/Desktop/R/データ"
```

getwd()というのは、現在の作業ディレクトリを調べるための命令です。メニューバーから選択しなくても、このコマンドを打ち込めば同じことをやってくれます。

2行目がディレクトリ（フォルダ）の位置になります。私はデスクトップに「R」というフォルダを作り、さらにその中に「データ」というフォルダを作って、そこにデータファイルを置きました。確かに、合っています。

作業ディレクトリが、確かにデータファイルを置いたフォルダになっていることを確認できたら、新しいRエディタを開いてください。そして以下のような命令を書き込んでください。これがCSVファイルを読み込む命令になります。

### # ファイル読み込み

```
x <- read.csv("sample.csv", header=TRUE, fileEncoding="CP932")
```

先にこの命令に関する解説をしておきましょう。まず1行目は「#」で始まっています。コメントなどとも呼ばれますが、Rは「#」で始まる部分無視します。単にRコンソールに表示するだけであり、計算などには影響しません。そのため、自分用のメモ、解説などに使うと便利です。

2行目の `read.csv` が、CSV形式のファイルを読みなさいという指示になります。続くカッコの中には、まずファイル名（" " でくくります）を指定し、続く `header=` の部分は、データの1行目が変数名である場合には `header=TRUE` と入力しておきます。変数名でない場合は `header=FALSE` と入力します。TRUE と FALSE は、その通りに、「真」と「偽」を意味します。これもよく出てくるので覚えておくとよいでしょう。ちなみに、TRUE は T、FALSE は F と略記することができます。しかし、いずれにしても小文字にしてはダメです。Rは大文字と小文字を区別します。

`fileEncoding="CP932"` の部分は、ある意味で Mac ユーザーならば必ず覚えておかなければならない部分かもしれません。Windows 版の R では、この部分がなくてもほとんど問題は生じないようですが、Mac で、特に変数名や変数に日本語が使われている場合だと、これを入れないとエラーになってしまう場合が多いと思います（「不正なマルチバイト文字があります」といったメッセージが出てきます）。`fileEncoding="CP932"` を入れるとエラーが出ず、うまくいくと思います。変数名や変数に日本語を使っていない場合に、これがなくても問題ない場合がほとんどだと思いますが、書いたからといって問題が生じるわけでもないようです。ひとつのおまじないだと思って入れておくとよいと思います。

ちなみに"CP932"は、Windows の Shift\_JIS を読む場合ですが、これでうまくいかなければ"UTF-8"（エクセルでのファイル保存に、「CSV UTF-8（コンマ区切り）(.csv)」を選んだ場合）または"UCS-2LE"なども試してみてください。他にも多く種類があるので、うまくいかない時は調べてみてください。

`x <- ...`という部分は、昨日やったとおりです。以上のような `read.csv` で読み込んだデータを、`x` という名前の入れ物に代入しています。つまり、今後は `x` という名前で、このデータすべてを操作できることとなります（これもひとつのベクトルです）。

本日の最後に、確かに `x` の中身が読み込んだデータ通りになっているか確認しておきましょう。中身を見るのはもちろん

**x**  
です。

右図のように出力されると思います。1002 番の `b2` は欠損値でした。R に読み込んでみると、ここが「NA」に変わっています。R は欠損値を「NA」と示すのです。また変数名の行には一番左の番号（行番号）がふられていないことにも注意しておいてください。番号がついていないということは、ここはデータの行ではないことを意味しています。

（まだ時間があれば、`header=FALSE` としてデータを読み込んで、違いを確認しておいてください）

これで5日目は終了です。R エディタに適切な名前を付けて、作業ディレクトリに保存しておいてください。

```
> x
  no sex age b1 b2 b3 b4 b5
1 1001  2 20  2  5  5  4  2
2 1002  2 20  2 NA  4  3  2
3 1003  2 19  3  4  4  4  3
4 1004  2 21  5  3  5  4  3
5 1005  1 20  3  5  5  5  5
6 1006  1 19  4  3  5  5  3
7 1007  2 20  5  5  5  4  4
8 1008  2 19  3  3  5  3  4
9 1009  2 19  2  4  3  4  3
10 1010  1 19  4  4  5  3  2
11 1011  1 19  1  3  4  4  4
12 1012  1 19  4  4  5  3  2
13 1013  2 19  2  3  3  4  4
14 1014  2 19  2  3  4  4  3
15 1015  2 19  2  5  5  5  4
16 1016  2 19  2  5  4  5  2
17 1017  2 19  4  3  4  3  3
18 1018  2 19  3  4  5  4  3
19 1019  1 19  1  4  4  4  4
20 1020  2 20  5  5  5  5  5
>
```