Comments to exercises (appendix B)

Juul, S. and M. Frydenberg 2010. An Introduction to Stata for Health Researchers, 3rd Edition.

Stata Press: College Station, TX.

Exercise B.1 and B.2 require no comments.

B.3 Calculations

The following exercises use datasets downloaded from the book's web site. Choose the directory where you stored them; in the book's examples, the command to do that is

```
. cd "C:\docs\ishr3"
```

B.3-1

```
. cd C:\docs\ishr3
. use smoke.dta
. codebook, compact
```

Variable	Obs	Unique	Mean	Min	Max	Label
id	230	230	134.0696	1	250	ID number
sex	230	2	1.73913	1	2	
age	230	54	55.80435	21	84	Age in years
weight	227	45	64.08811	43	110	Weight, kg
height	227	40	166.9736	150	194	Height, cm
smoker	230	3	.9217391	0	2	Smoker?
cigaret	230	21	4.730435	0	40	Cigarettes/day
cheroot	230	9	.2086957	0	10	Cigars, cheroots/day
pipe	230	3	.0347826	0	2	Pipe, packs/week

For `weight` and `height`, three observations have missing values, so there are only 227 observations with nonmissing values.

Study the minimum and maximum value for each variable. Did we expect participants to be between 20 and 85 years old? Is it plausible that an adult person weighs 43 kg? To us, the minimum and maximum values for all variables give no rise to concern. Also look at the `Unique` column; it tells how many unique, i.e., different, values the variables have. Relief: there were only two sexes.

The output from `summarize` also displays the standard deviation but not a `Unique` column and not the variable labels:

```
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
id	230	134.0696	68.99164	1	250
sex	230	1.73913	.4400666	1	2
age	230	55.80435	14.27799	21	84
weight	227	64.08811	11.85798	43	110
height	227	166.9736	8.370788	150	194
smoker	230	.9217391	.7435055	0	2
cigaret	230	4.730435	7.186015	0	40
cheroot	230	.2086957	1.105727	0	10
pipe	230	.0347826	.2262446	0	2

B.3-2

```
. generate weightlb = weight/0.454
(3 missing values generated)

. generate heightin = height/2.54
(3 missing values generated)

. codebook, compact
```

Variable	Obs	Unique	Mean	Min	Max	Label
id	230	230	134.0696	1	250	ID number
sex	230	2	1.73913	1	2	
age	230	54	55.80435	21	84	Age in years
weight	227	45	64.08811	43	110	Weight, kg
height	227	40	166.9736	150	194	Height, cm
smoker	230	3	.9217391	0	2	Smoker?
cigaret	230	21	4.730435	0	40	Cigarettes/day
cheroot	230	9	.2086957	0	10	Cigars, cheroots/day
pipe	230	3	.0347826	0	2	Pipe, packs/week
weightlb	227	45	141.1632	94.71365	242.2908	
heightin	227	40	65.73763	59.05512	76.37795	

The results look right. The three missing values were expected. Check whether 43 kg. is the same as 94.71 lb. Is 76.38 in. the same as 194 cm? Use your pocket calculator or Stata's `display` command:

```
. display 94.71365*0.454
42.999997

. display 194/2.54
76.377953
```

Obviously, the new variables should be furnished with variable labels (next exercise).

B.3-3–B.3-5 The following do-file, `gen_smoke1.do`, includes the actions related to three questions.

gen_smoke1.do

```
// gen_smoke1.do

cd "C:\docs\ishr3"
use smoke.dta

generate weightlb = weight/0.454
label variable weightlb "Weight, lb."

generate heightin = height/2.54
label variable heightin "Height, in."

generate bmi = weight/((height/100)^2)
label variable bmi "Body mass index (kg/m^2)"

save smoke1.dta
```

Note these points:

1. We give the do-file a name that tells what it does (`gen_smoke1.do` generates `smoke1.dta`).
2. We start the do-file with a comment stating the do-file's own name. Comments mean nothing to Stata, but this one is useful to the user because the do-file's name would be included in a printout.
3. The `cd` command ensures that the datasets are read from and written to the correct folder.
4. The do-file starts with a `use` command and ends with a `save` command, making sure that both the input and the output file are defined.
5. We modify the data in `smoke.dta`, and we therefore save a file with a new name. Overwriting the primary dataset is outright dangerous.
6. We define the variable label immediately after generating a new variable.

It is likely that you needed to modify the do-file, and issuing the `save` command a second time leads to an error message:

```
. save smoke1.dta
file smoke1.dta already exists
r(602);
```

Stata wants to prevent unintended overwriting of data on your hard disk, but here you want to overwrite. Allow overwriting by adding the `replace` option to the last line in the do-file:

```
. save smoke1.dta, replace
```

This is the typical situation that justifies the `replace` option, but it should be used with care. Above all, make sure that you do not replace the input file (`smoke.dta` in this case).

You may see this error message after the `use` command:

```
. use smoke.dta
no; data in memory would be lost
r(4);
```

This happens if there are unsaved modifications to the data in memory. You must evaluate the situation: Would it be harmful to overwrite the data in memory? If it is okay to delete them, type

```
. clear
```

and run the do-file again. If you think you need to save the data in memory, think again: All preservation-worthy modifications to your data should be made with do-files ending with a `save` command, like the above do-file (`gen_smoke1.do`). Modifying and saving data interactively leaves things undocumented, and this should be avoided.

The dataset now includes:

```
. codebook, compact
```

Variable	Obs	Unique	Mean	Min	Max	Label
id	230	230	134.0696	1	250	ID number
sex	230	2	1.73913	1	2	
age	230	54	55.80435	21	84	Age in years
weight	227	45	64.08811	43	110	Weight, kg
height	227	40	166.9736	150	194	Height, cm
smoker	230	3	.9217391	0	2	Smoker?
cigaret	230	21	4.730435	0	40	Cigarettes/day
cheroot	230	9	.2086957	0	10	Cigars, cheroots/day
pipe	230	3	.0347826	0	2	Pipe, packs/week
weightlb	227	45	141.1632	94.71365	242.2908	Weight, lbs.
heightin	227	40	65.73763	59.05512	76.37795	Height, in.
bmi	226	194	22.98144	15.46522	38.06228	Body mass index (kg/m^2)

B.3-6, B.3-7 We modify `gen_smoke1.do` to include generation of the new variables:

gen_smoke1.do

```
// gen_smoke1.do

cd "C:\docs\ishr3"
use smoke.dta

generate weightlb = weight/0.454
label variable weightlb "Weight, lbs."

generate heightin = height/2.54
label variable heightin "Height, in."

generate bmi = weight/((height/100)^2)
label variable bmi "Body mass index, kg/m^2"

recode age (min/44=1)(45/64=2)(65/max=3), generate(agegrp)
label variable agegrp "Age, three groups"
label define agelabel 1 "-44" 2 "45-64" 3 "65+"
label values agegrp agelabel
numlabel, add

generate tobacco = cigaret + cheroot*2 + pipe*40/7
label variable tobacco "Tobacco, grams/day"

save smoke1.dta, replace
```

Again define labels at once for new variables. The `numlabel, add` command includes the code (1, 2, 3) in the value label. Instead of the `label define`, `label values` construct, the `value label` definitions can be made within the `recode` command:

```
. recode age (min/44=1 "-44")(45/64=2 "45-64")(65/max=3 "65+"),
> generate(agegrp)
```

To check for correctness of the `recode`, let `tabstat` display the correspondence between `age` and `agegrp`:

```
. tabstat age, by(agegrp) stat(min max)

Summary for variables: age
     by categories of: agegrp (Age, three groups)
```

agegrp	min	max
1. -44	21	44
2. 45-64	45	64
3. 65+	65	84
Total	21	84

The minimum and maximum values are as expected; the recoding went right.

B.3-8

```
. use smoke1.dta
. tab1 tobacco cigaret
-> tabulation of tobacco
```

Tobacco, grams/day	Freq.	Percent	Cum.
0	128	55.65	55.65
3	6	2.61	58.26
4	2	0.87	59.13
5	7	3.04	62.17
5.714286	2	0.87	63.04
6	6	2.61	65.65
7	2	0.87	66.52
8	4	1.74	68.26
9	2	0.87	69.13
10	21	9.13	78.26
10.71429	1	0.43	78.70
11	1	0.43	79.13
11.42857	1	0.43	79.57
12	4	1.74	81.30
13	8	3.48	84.78
14	2	0.87	85.65
15	14	6.09	91.74
15.71429	1	0.43	92.17
17	2	0.87	93.04
20	11	4.78	97.83
25	1	0.43	98.26
29.42857	1	0.43	98.70
30	2	0.87	99.57
40	1	0.43	100.00
Total	230	100.00	

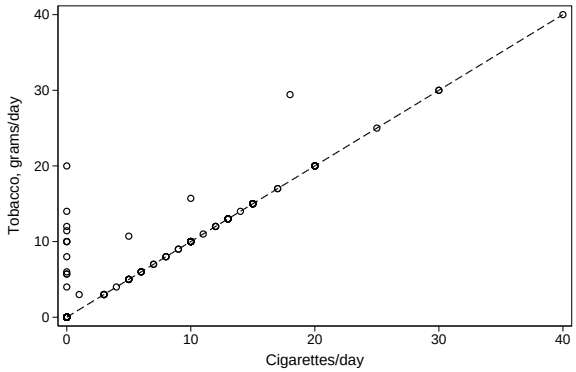
(*Continued on next page*)

```
-> tabulation of cigaret
```

Cigarettes/ day	Freq.	Percent	Cum.
0	140	60.87	60.87
1	1	0.43	61.30
3	5	2.17	63.48
4	1	0.43	63.91
5	8	3.48	67.39
6	5	2.17	69.57
7	2	0.87	70.43
8	3	1.30	71.74
9	2	0.87	72.61
10	19	8.26	80.87
11	1	0.43	81.30
12	3	1.30	82.61
13	8	3.48	86.09
14	1	0.43	86.52
15	14	6.09	92.61
17	2	0.87	93.48
18	1	0.43	93.91
20	10	4.35	98.26
25	1	0.43	98.70
30	2	0.87	99.57
40	1	0.43	100.00
Total	230	100.00	

Most smokers smoke cigarettes only, so the two tables are not very different. In this case, a graph is illustrative. People who smoke nothing but cigarettes should be on the identity line, other smokers should be above the line, and nobody should be below it:

```
. twoway (scatter tobacco cigaret)(function y=x, range(cigaret)),
> legend(off) ytitle("Tobacco, grams/day") xtitle("Cigarettes/day")
```



To check that the calculations were made correctly, list a sample of observations, including both the source variables and the target variable:

```
. list cigaret cheroot pipe tobacco in 1/30
```

	cigaret	cheroot	pipe	tobacco
1.	0	0	0	0
2.	0	0	0	0
3.	18	0	2	29.42857
4.	0	0	0	0
5.	0	0	0	0
6.	6	0	0	6
7.	0	0	0	0

(*output omitted*)

Pipe smokers get some strange values because the tobacco amount was given in packs per week, and one pack corresponds to $40/7 = 5.71429$ grams per day. This degree of precision is nonsense, of course, and we could safely round to the nearest gram:

```
. replace tobacco = round(tobacco)
. tab1 tobacco
-> tabulation of tobacco
```

Tobacco, grams/day	Freq.	Percent	Cum.
0	128	55.65	55.65
3	6	2.61	58.26
4	2	0.87	59.13
5	7	3.04	62.17
6	8	3.48	65.65

(*output omitted*)

B.3-9 The output will look something like this:

```
. clear
. set obs 1
obs was 0, now 1
. generate x1 = 1/5
. generate double x2 = 1/5
. list if x1==0.2, clean
. list if x2==0.2, clean
     x1   x2
  1.  .2   .2
. list, clean
     x1   x2
  1.  .2   .2
```

The second and the third `list` commands display the expected output, but the first `list` command did not because the expression `x1==0.2` was false. Despite appearances, Stata has not made a mistake. Stata (and other computer software) stores numbers internally in binary form, and the number 0.2 has no exact binary representation. This means that the

`float` variable `x1` cannot have exactly the same value as the `double` variable `x2`. During evaluation of the expression `x1==0.2`, 0.2 is stored in double precision, so it has the same value as `x2` but not as `x1`.

B.4 Working with missing values

B.4-1 The list from question B.4-1 might look like:

```
. list id weight height bmi if bmi>30
```

	id	weight	height	bmi
4.	4	85	166	30.84628
13.	14	87	167	31.1951
48.	68	90	172	30.42185
63.	83	.	175	.
71.	91	.	.	.
73.	93	.	.	.
75.	95	85	163	31.99217
81.	101	91	172	30.75987
84.	104	81	157	32.86137
164.	184	50	.	.
196.	216	110	170	38.06228
211.	231	90	173	30.07117

You may wonder why the list includes four observations with a missing `bmi`. Stata considers missing values to be very large numbers, so a missing `bmi` is evaluated as larger than 30; see section 5.3. You may want to modify the command to

```
. list id weight height bmi if bmi>30 & bmi<.
```

or

```
. list id weight height bmi if bmi>30 & !missing(bmi)
```

	id	weight	height	bmi
4.	4	85	166	30.84628
13.	14	87	167	31.1951
48.	68	90	172	30.42185
75.	95	85	163	31.99217
81.	101	91	172	30.75987
84.	104	81	157	32.86137
196.	216	110	170	38.06228
211.	231	90	173	30.07117

B.4-2 The values of all five newly generated variables were missing:

```
. list
```

	a	b	c	d	e
1.	.	.	.	.	.

The reasons are presented here:

Command	Reason for missingness
`generate a = 1/0`	division by 0
`generate b = sqrt(-4)`	the square root of a negative number
`generate c = ln(0)`	the log of 0
`generate d = mdy(2,29,2001)`	29 February 2001 is a nonexisting date
`generate e = a+5`	missing + anything = missing

B.5 Working with date variables

B.5-1 Generate the modified dataset with a do-file:

gen_dates2.do

```
* gen_dates2.do

cd C:\docs\ishr3
use dates.dta

generate bdate = mdy(bm,bd,by)
label variable bdate "Date of birth"

generate adate = date(adate_s,"DMY")
label variable adate "Date of admission"
format bdate adate %td

generate admage = (adate-bdate)/365.25
label variable admage "Age at admission, years"

generate admyr = year(adate)
label variable admyr "Year of admission"

save dates2.dta
```

```
. list
```

	bd	bm	by	adate_s	bdate	adate	admage	admyr
1.	12	3	1955	15072004	12mar1955	15jul2004	49.34428	2004
2.	24	12	1964	20112003	24dec1964	20nov2003	38.90486	2003
3.	3	12	1961	14012004	03dec1961	14jan2004	42.11362	2004

B.5-2

```
. format bdate %tdNN/DD/CCYY
. list
```

	bd	bm	by	adate_s	bdate	adate	admage	admyr
1.	12	3	1955	15072004	03/12/1955	15jul2004	49.34428	2004
2.	24	12	1964	20112003	12/24/1964	20nov2003	38.90486	2003
3.	3	12	1961	14012004	12/03/1961	14jan2004	42.11362	2004

B.5-3

```
. display mdy(7,1,2008)
17714
. display %td 2345
03jun1966
```

B.6 Description and simple analysis

Use do-files for all important analyses. And always start an analysis by specifying the path and filename:

analysis1.do

```
// analysis1.do
// Initial overview
cd C:\docs\ishr3
use smoke1.dta
```

Note the comments stating the do-file's own name and its purpose. The name of this do-file does not start with `gen_`. It does not generate a modified dataset, and the purpose of the `gen_` prefix is to distinguish between data-management do-files and analysis do-files.

B.6-1

```
. describe

Contains data from smoke1.dta
  obs:           230
 vars:            14                          16 Aug 2007 14:03
 size:        11,730 (99.9% of memory free)

-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
id              float  %9.0g                  ID number
sex             byte   %17.0g      sex
age             float  %9.0g                  Age in years
weight          float  %9.0g                  Weight, kg
height          float  %9.0g                  Height, cm
smoker          byte   %17.0g      smoker     Smoker?
cigaret         float  %9.0g                  Cigarettes/day
cheroot         float  %9.0g                  Cigars, cheroots/day
pipe            byte   %8.0g                  Pipe, packs/week
weightlb        float  %9.0g                  Weight, lbs.
heightin        float  %9.0g                  Height, in.
bmi             float  %9.0g                  Body mass index, kg/m^2
agegrp          float  %9.0g       agelabel   Age, three groups
tobacco         float  %9.0g                  Tobacco, grams/day
-------------------------------------------------------------------------------
Sorted by:
```

`describe` tells about the variables: their names, storage types, display formats, and labels, if any, and the sorting status of the dataset. This dataset was not sorted. Three variables had value labels; use `label list` to see them:

```
. label list
agelabel:
           1 1. -44
           2 2. 45-64
           3 3. 65+
sex:
           1 1. male
           2 2. female
           9 9. no information
smoker:
           0 0. no
           1 1. current
           2 2. former
           9 9. no information
```

```
. codebook, compact
```

Variable	Obs	Unique	Mean	Min	Max	Label
id	230	230	134.0696	1	250	ID number
sex	230	2	1.73913	1	2	
age	230	54	55.80435	21	84	Age in years
weight	227	45	64.08811	43	110	Weight, kg
height	227	40	166.9736	150	194	Height, cm
smoker	230	3	.9217391	0	2	Smoker?
cigaret	230	21	4.730435	0	40	Cigarettes/day
cheroot	230	9	.2086957	0	10	Cigars, cheroots/day
pipe	230	3	.0347826	0	2	Pipe, packs/week
weightlb	227	45	141.1632	94.71365	242.2908	Weight, lbs.
heightin	227	40	65.73763	59.05512	76.37795	Height, in.
bmi	226	194	22.98144	15.46522	38.06228	Body mass index, kg/m^2
agegrp	230	3	2.113043	1	3	Age, three groups
tobacco	230	24	5.346584	0	40	Tobacco, grams/day

```
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
id	230	134.0696	68.99164	1	250
sex	230	1.73913	.4400666	1	2
age	230	55.80435	14.27799	21	84
weight	227	64.08811	11.85798	43	110
height	227	166.9736	8.370788	150	194
smoker	230	.9217391	.7435055	0	2
cigaret	230	4.730435	7.186015	0	40
cheroot	230	.2086957	1.105727	0	10
pipe	230	.0347826	.2262446	0	2
weightlb	227	141.1632	26.1189	94.71365	242.2908
heightin	227	65.73763	3.295586	59.05512	76.37795
bmi	226	22.98144	3.649023	15.46522	38.06228
agegrp	230	2.113043	.7448832	1	3
tobacco	230	5.346584	7.398785	0	40

The main advantage of `codebook, compact` is the display of variable labels, but for large datasets, it can be quite slow.

B.6-2 `tab1` gives a simple distribution of a categorical variable. If value labels are defined, they will be displayed, not the codes:

```
. tab1 smoker
-> tabulation of smoker
```

Smoker?	Freq.	Percent	Cum.
0. no	73	31.74	31.74
1. current	102	44.35	76.09
2. former	55	23.91	100.00
Total	230	100.00	

To see the codes, use the `nolabel` option:

```
. tab1 smoker, nolabel
-> tabulation of smoker
```

Smoker?	Freq.	Percent	Cum.
0	73	31.74	31.74
1	102	44.35	76.09
2	55	23.91	100.00
Total	230	100.00	

To see both the code and the value label, use `numlabel` to add the codes to the labels:

```
. numlabel, add
(no value label to be modified)
. tab1 smoker
-> tabulation of smoker
```

Smoker?	Freq.	Percent	Cum.
0. no	73	31.74	31.74
1. current	102	44.35	76.09
2. former	55	23.91	100.00
Total	230	100.00	

B.6-3 Obviously, we should use the grouped age; the ungrouped age has 54 unique values, so we would get a huge and useless table:

```
. tab2 agegrp sex, col chi2 exact
-> tabulation of agegrp by sex
```

Key
frequency *column percentage*

(*output omitted*)

Age, three groups	sex 1. male	2. female	Total
1. -44	14	38	52
	23.33	22.35	22.61
2. 45-64	29	71	100
	48.33	41.76	43.48
3. 65+	17	61	78
	28.33	35.88	33.91
Total	60	170	230
	100.00	100.00	100.00

```
          Pearson chi2(2) =   1.2042   Pr = 0.548
           Fisher's exact =                 0.546
```

B.6-4 There are several possible commands when you want to compare means. The `tabstat` command is quite flexible:

```
. tabstat bmi, by(sex)

Summary for variables: bmi
     by categories of: sex
```

sex	mean
1. male	23.85571
2. female	22.66544
Total	22.98144

```
. tabstat bmi, by(agegrp)

Summary for variables: bmi
     by categories of: agegrp (Age, three groups)
```

agegrp	mean
1. -44	21.72467
2. 45-64	23.07006
3. 65+	23.72707
Total	22.98144

No surprise here: BMI is higher for men, and it increases with age. Use `ttest` to compare sexes:

```
. ttest bmi, by(sex)

Two-sample t test with equal variances
```

Group	Obs	Mean	Std. Err.	Std. Dev.	[95% Conf.	Interval]
1. male	60	23.85571	.4578328	3.546357	22.93959	24.77183
2. femal	166	22.66544	.282862	3.644422	22.10694	23.22394
combined	226	22.98144	.2427294	3.649023	22.50313	23.45975
diff		1.190271	.5451239		.1160439	2.264498

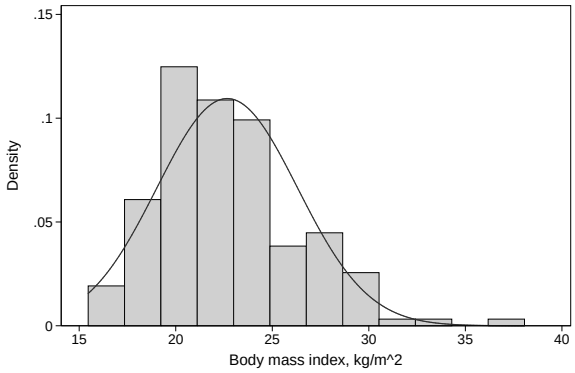
```
    diff = mean(1. male) - mean(2. femal)                         t =   2.1835
Ho: diff = 0                                     degrees of freedom =      224

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.9850         Pr(|T| > |t|) = 0.0300          Pr(T > t) = 0.0150
```

In the bottom line, ignore the leftmost and rightmost results. The middle Pr $= 0.03$ is the relevant two-sided test.

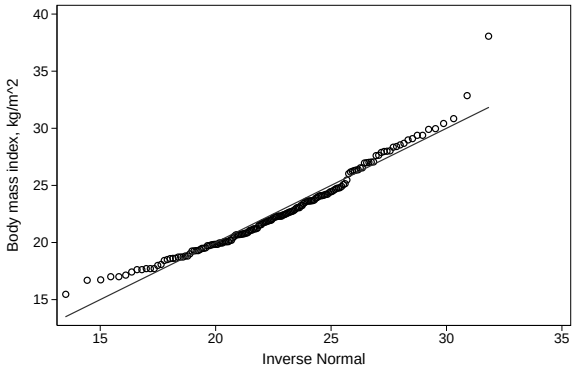
B.6-5 To assess normality, a histogram is useful. We look at women only:

```
. histogram bmi if sex==2, normal
```



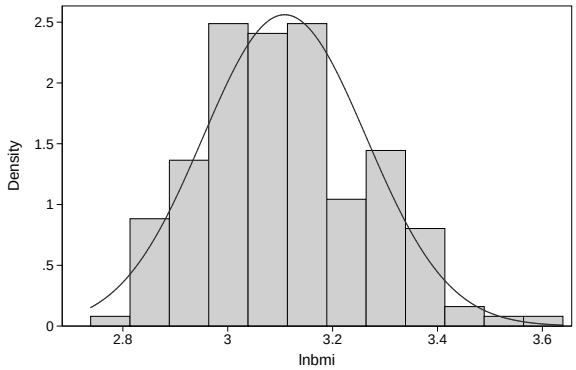
As expected, this distribution is right-skewed. This is also illustrated by the Q–Q plot:

```
. qnorm bmi if sex==2
```



The right-skewed distribution can be normalized with a log transformation:

```
. generate lnbmi = ln(bmi)
. histogram lnbmi if sex==2, normal
```



Obviously, the distribution is less skewed. We make a `ttest` with the log-transformed BMI:

```
. ttest lnbmi, by(sex)
```
(*output omitted*)

B.6-6

```
. sdtest lnbmi, by(sex)
Variance ratio test
```

Group	Obs	Mean	Std. Err.	Std. Dev.	[95% Conf.	Interval]
1. male	60	3.16103	.0193994	.1502667	3.122212	3.199848
2. femal	166	3.108575	.0120878	.1557407	3.084709	3.132442
combined	226	3.122501	.010358	.1557146	3.10209	3.142913

```
    ratio = sd(1. male) / sd(2. femal)                     f =   0.9309
Ho: ratio = 1                                 degrees of freedom =  59, 165

    Ha: ratio < 1               Ha: ratio != 1                 Ha: ratio > 1
  Pr(F < f) = 0.3832         2*Pr(F < f) = 0.7665          Pr(F > f) = 0.6168
```

Again ignore the leftmost and rightmost p-values. The standard deviations are quite similar, and they are not significantly different (Pr $= 0.77$).

B.6-7 `ci` is a useful command:

```
. ci bmi
```

Variable	Obs	Mean	Std. Err.	[95% Conf.	Interval]
bmi	226	22.98144	.2427294	22.50313	23.45975

```
. recode sex (2=0), generate(male)
(170 differences between sex and male)
```

```
. ci male, binomial level(90)
```

Variable	Obs	Mean	Std. Err.	Binomial Exact [90% Conf.	Interval]
male	230	.2608696	.028954	.2135872	.3128254

The mean BMI was 22.98 (95% CI: 22.50 to 23.46). The proportion males was 0.26 (90% CI: 0.21 to 0.31).

B.7 Taking good care of your data

In the examples below, we use these fonts to show what changes should be made:

`cd "C:\docs\project xyz"`	No change
`use xyz.dta`	Add this command
~~`save xyz1.dta`~~	Delete this command

B.7-1

gen_xyz1.do

```
* gen_xyz1.do
cd "C:\docs\project xyz"
use xyz.dta
generate age1 = (date1 - bdate)/365.25
label variable age1 "Age at entry"
generate bmi = weight/(height^2)
label variable bmi "Body mass index (kg/m^2)"
save xyz1.dta
* This do-file generated two new variables.
```

Two new variables were generated, but we do not know what the data input was nor if any dataset was saved to disk; we need an initial `use` command and a final `save` command. The do-file name suggests that we wanted to save `xyz1.dta`, but did we do that?

The two new variables were not furnished with labels. Add labels right after a new variable is generated. It is not easier, only more difficult to postpone it. (In several of the following examples, labels are lacking as well).

B.7-2

~~gen_xyz1.do~~
gen_xyz2.do

```
~~* gen_xyz1.do~~
** gen_xyz2.do
cd "C:\docs\project xyz"
use xyz1.dta
generate age1 = (date1 - bdate)/365.25
generate bmi = weight/(height^2)
keep if !missing(bmi)
~~save xyz1.dta, replace~~
save xyz2.dta
```

We overwrote `xyz1.dta` with a modified version of the dataset, thus destroying the information in the original `xyz1.dta`. We might regret that we dropped some observations; now the risk is that they can never be restored.

B.7-3

~~gen_xyz1.do~~
gen_xyz2.do

```
~~* gen_xyz1.do~~
** gen_xyz2.do
cd "C:\docs\project xyz"
use xyz1.dta
generate age1 = (date1 - bdate)/365.25
generate bmi = weight/(height^2)
save xyz2.dta
```

The do-file generates `xyz2.dta`, but its name suggested something else. That was a design for confusion.

B.7-4

gen_xyz1.do

```
* gen_xyz1.do
cd "C:\docs\project xyz"
use xyz.dta
generate age1 = (date1 - bdate)/365.25
recode age1 (min/20=1)(20/40=2)(40/max=3), generate(age1grp)
generate bmi = weight/(height^2)
save xyz1.dta
```

The `recode` command destroyed the original `age1`. Use the `generate()` option. Another problem is the lack of variable labels and value labels for the new variables (see later examples).

B.7-5

gen_xyz2.do

```
* gen_xyz2.do
cd "C:\docs\project xyz"
use xyz1.dta
replace age1 = age1-5 if sex==2
generate age1_f = age1-5 if sex==2
replace bmi = bmi-2 if sex==2
generate bmi_f = bmi-2 if sex==2
save xyz2.dta
```

We made changes to two variables, overwriting their original values. This is risky; generate new variables instead.

B.7-6

gen_xyz1.do

```
* gen_xyz1.do
cd "C:\docs\project xyz"
use xyz.dta
generate age1 = (date1 - bdate)/365.25
label variable age1 "Age at admission"
recode age1 (40/max=3)(20/40=2)(min/20=1), generate(age1grp)
label variable age1grp "Age (3 groups) at admission"
label define agelab 1 "-19" 2 "20-39" 3 "40+"
label values age1grp agelab
generate bmi = weight/(height^2)
label variable bmi "Body mass index (kg/m^2)"
save xyz1.dta
```

This looks better than most of the preceding examples. We missed, however, defining value labels for the new categorical variable `age1grp` (see the alternative method in the next example).

B.7-7

gen_xyz1.do

```
~~* gen_xyz2.do~~
* gen_xyz1.do

cd "C:\docs\project xyz"

use xyz.dta

generate age1 = (date1 - bdate)/365.25
label variable age1 "Age at admission"

recode age1 (40/max=3 "40+")(20/40=2 "20-39") (min/20=1 "-19"), ///
     generate(age1grp)
label variable age1grp "Age (grouped) at admission"

generate bmi = weight/(height^2)
label variable bmi "Body mass index (kg/m^2)"

save xyz1.dta
```

This do-file is nice, but note the initial comment that should display the do-file's own name. It displayed a different name, which may have lead to confusion.