COGS 119
MATLAB for Experimental Research

Fall 2013 – Week 5

Writing into Files in Matlab

Writing into files

```
>> help fprintf
```

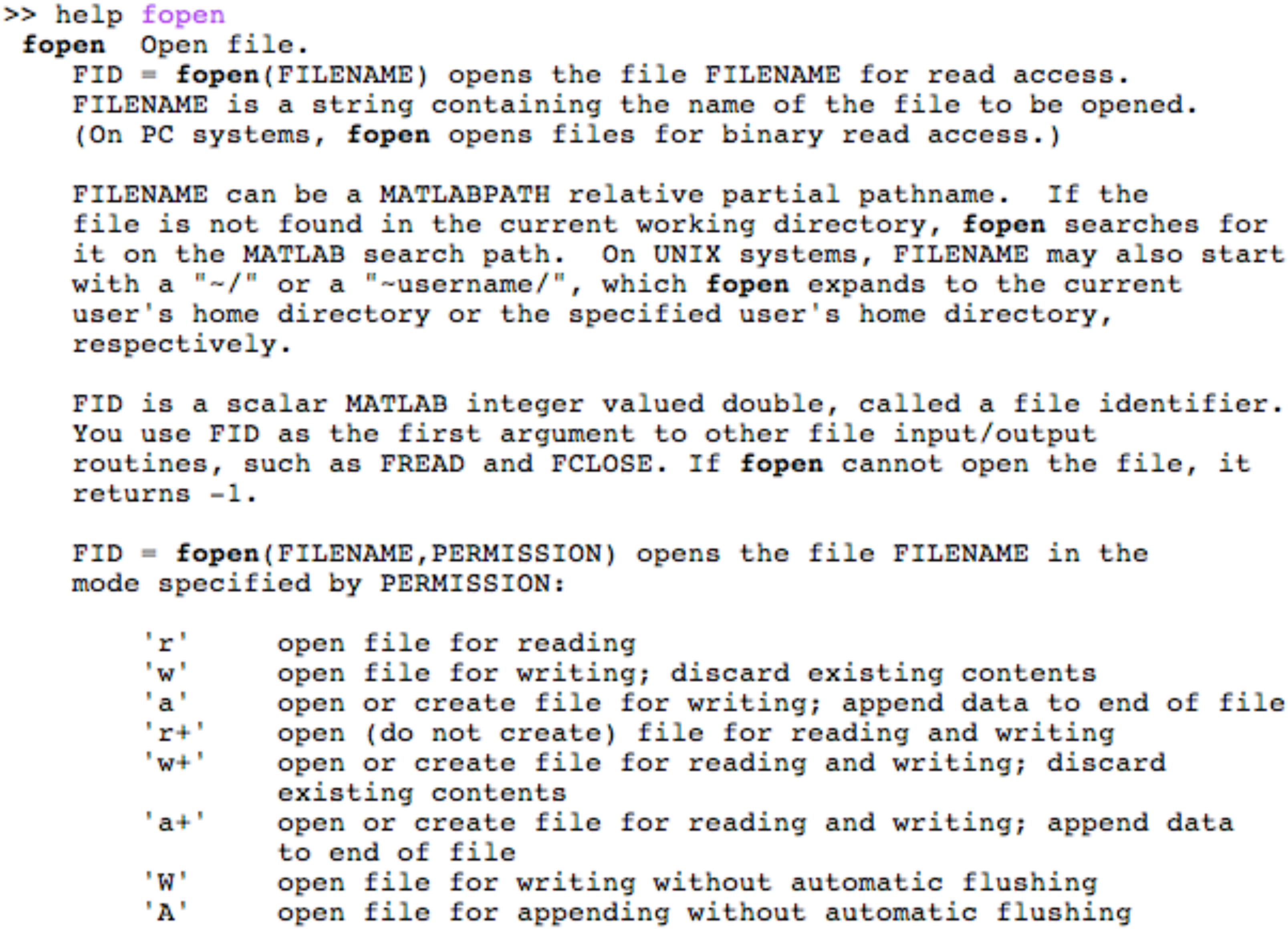
fprintf syntax

```
fprintf (fileID, format, A, ...)
```

- It applies the *format* to all elements of array *A* and any additional array arguments in column order, and writes the data to a text file.
- `fprintf` uses the encoding scheme specified in the call to `fopen`.

```
>> help fopen
```

fopen



```
>> help fopen
 fopen  Open file.
    FID = fopen(FILENAME) opens the file FILENAME for read access.
    FILENAME is a string containing the name of the file to be opened.
    (On PC systems, fopen opens files for binary read access.)

    FILENAME can be a MATLABPATH relative partial pathname.  If the
    file is not found in the current working directory, fopen searches for
    it on the MATLAB search path.  On UNIX systems, FILENAME may also start
    with a "~/" or a "~username/", which fopen expands to the current
    user's home directory or the specified user's home directory,
    respectively.

    FID is a scalar MATLAB integer valued double, called a file identifier.
    You use FID as the first argument to other file input/output
    routines, such as FREAD and FCLOSE. If fopen cannot open the file, it
    returns -1.

    FID = fopen(FILENAME,PERMISSION) opens the file FILENAME in the
    mode specified by PERMISSION:

        'r'     open file for reading
        'w'     open file for writing; discard existing contents
        'a'     open or create file for writing; append data to end of file
        'r+'    open (do not create) file for reading and writing
        'w+'    open or create file for reading and writing; discard
                existing contents
        'a+'    open or create file for reading and writing; append data
                to end of file
        'W'     open file for writing without automatic flushing
        'A'     open file for appending without automatic flushing
```

fprintf syntax

```
fprintf(format, A, ...)
```

- It formats data and displays the results on the screen.

fprintf syntax

```
count = fprintf(...)
```

- It returns the number of bytes that fprintf writes.

Input arguments of fprintf

fileID

One of the following:

- An integer file identifier obtained from fopen.
- 1 for standard output (the screen).
- 2 for standard error.
- **Default:** 1 (the screen)

Examples of printing to screen

```
>> fprintf(‘a’)
a>>fprintf(‘a\n’)
a
>> fprintf(‘a\n\n’)
a

```

Examples of bytes

```
>> c = fprintf(‘a’);   → c = 1
>> c = fprintf(‘a\n’);   → c = 2
>> c = fprintf('a\tj,s\n'); → c = 6
a      j,s
```

Input arguments of fprintf

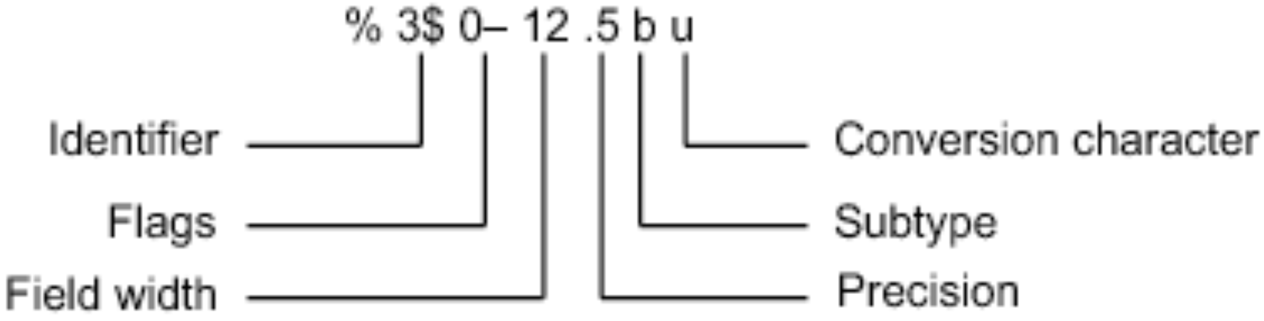
format

String in single quotation marks that describes the format of the output fields. Can include combinations of the following:

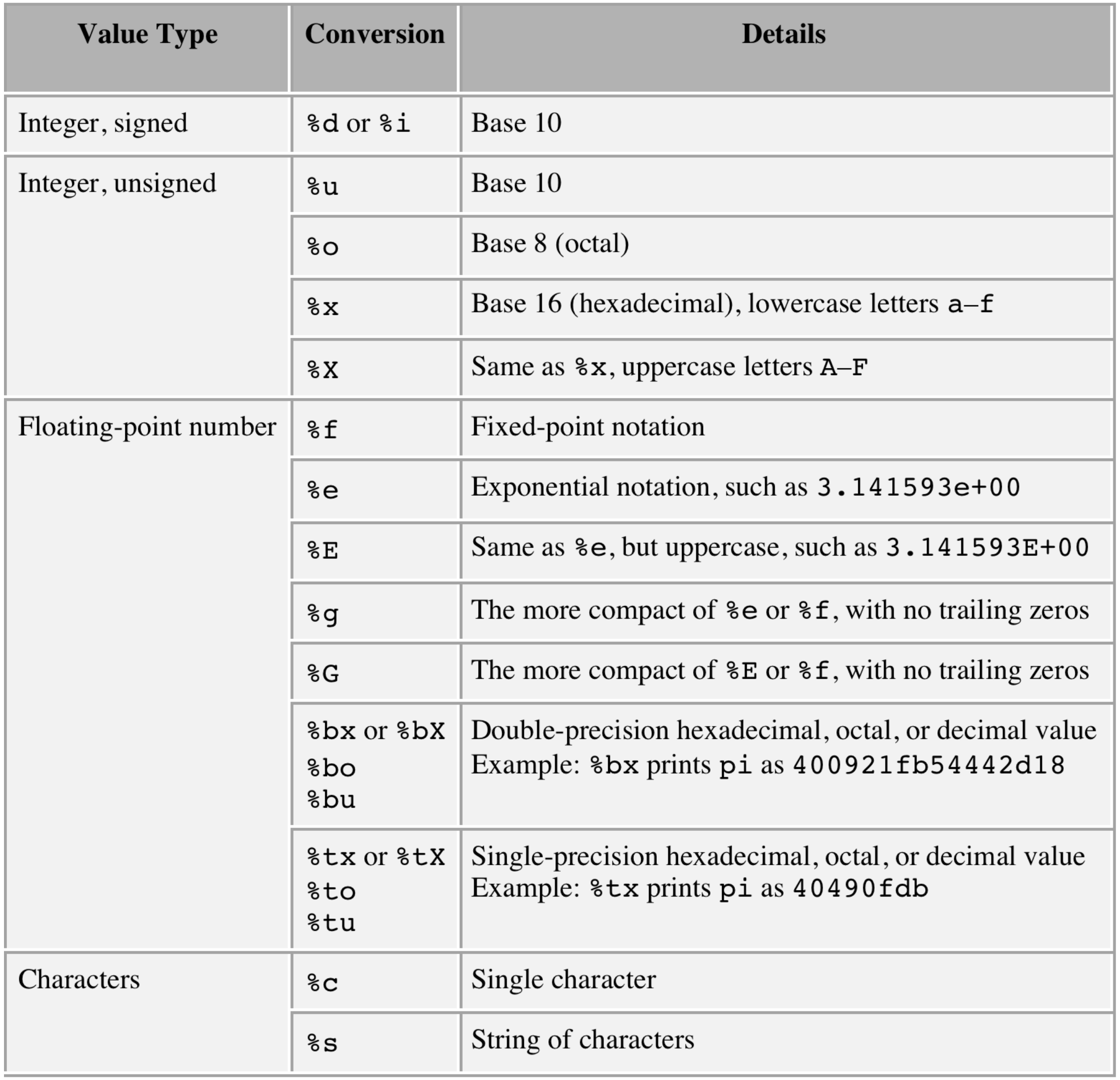
- Percent sign followed by a conversion character, such as '%s' for strings.
- Operators that describe field width, precision, and other options.
- Literal text to print.
- Escape characters, including:

”	Single quotation mark
%%	Percent character
\\	Backslash
\a	Alarm
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\x*N*	Hexadecimal number, *N*
\N	Octal number, *N*

- Conversion characters and optional operators appear in the following order (includes spaces for clarity):



List of the available conversion characters and subtypes



Value Type	Conversion	Details
Integer, signed	`%d` or `%i`	Base 10
Integer, unsigned	`%u`	Base 10
	`%o`	Base 8 (octal)
	`%x`	Base 16 (hexadecimal), lowercase letters `a`–`f`
	`%X`	Same as `%x`, uppercase letters `A`–`F`
Floating-point number	`%f`	Fixed-point notation
	`%e`	Exponential notation, such as `3.141593e+00`
	`%E`	Same as `%e`, but uppercase, such as `3.141593E+00`
	`%g`	The more compact of `%e` or `%f`, with no trailing zeros
	`%G`	The more compact of `%E` or `%f`, with no trailing zeros

	`%bx` or `%bX`
	`%bo`
	`%bu`
		Double-precision hexadecimal, octal, or decimal value
		Example: `%bx` prints `pi` as `400921fb54442d18`

	`%tx` or `%tX`
	`%to`
	`%tu`
		Single-precision hexadecimal, octal, or decimal value
		Example: `%tx` prints `pi` as `40490fdb`

Characters	`%c`	Single character
	`%s`	String of characters

Additional operators

Field width:

- Minimum number of characters to print.
- Can be a number, or an asterisk (*) to refer to an argument in the input list.
- For example, the input list `('%12d', intmax)` is equivalent to `('%*d', 12, intmax)`

Additional operators

Precision:

- For %f, %e, or %E:

  Number of digits to the right of the decimal point.
  Example: '%6.4f' prints pi as '3.1416'

- For %g or %G:

  Number of significant digits.
  Example: '%6.4g' prints pi as ' 3.142’

- Can be a number, or an asterisk (*) to refer to an argument in the input list.

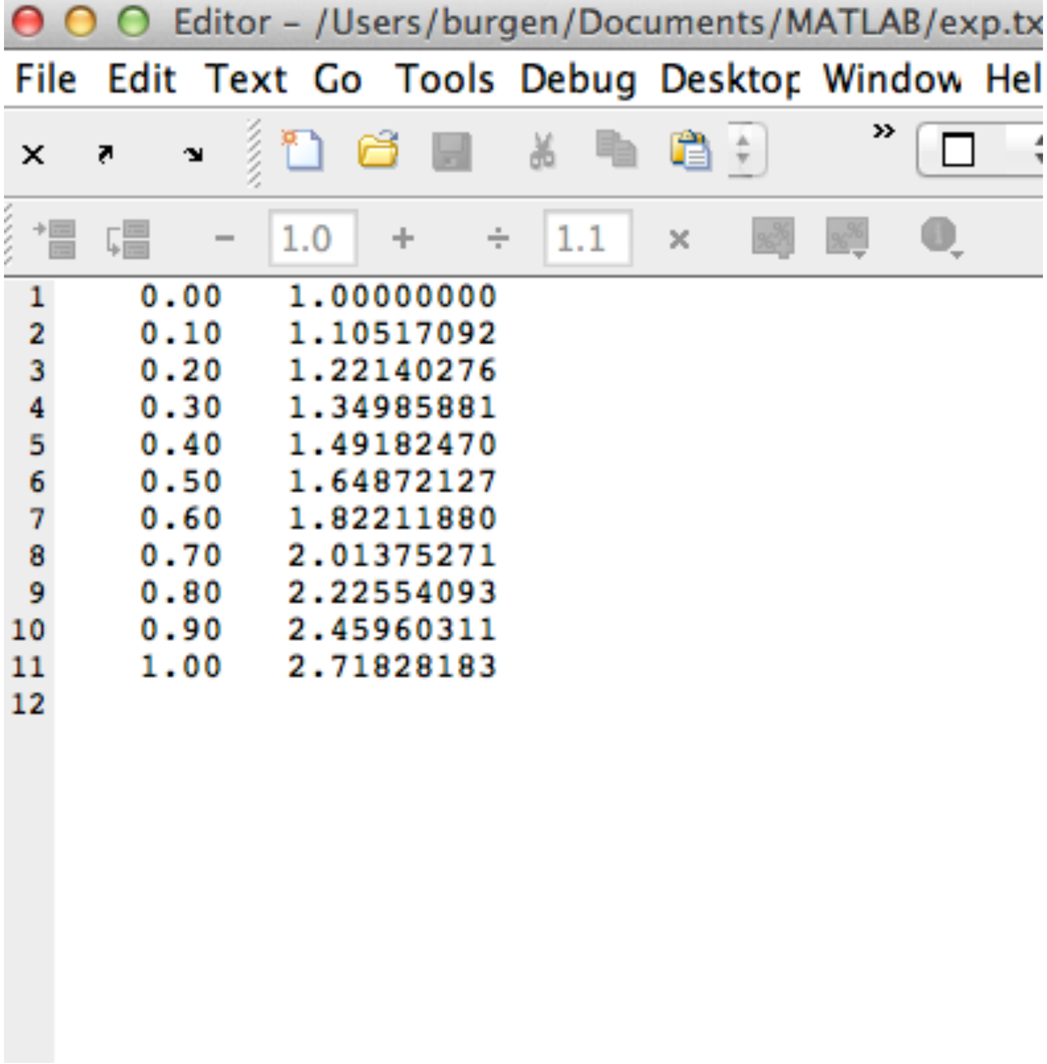
- For example, the input list ('%6.4f', pi) is equivalent to ('%*.*f', 6, 4, pi).

Example 1

Let's write a short table of the exponential function to a text file called exp.txt:

```
x = 0:.1:1;
y = [x; exp(x)];

% open the file with write permission
fid = fopen('exp.txt', 'w');

fprintf(fid, '%6.2f %12.8f\n', y);

fclose(fid);
% view the contents of the file type exp.txt
```

exp.txt


Editor – /Users/burgen/Documents/MATLAB/exp.tx
File Edit Text Go Tools Debug Desktop Window Hel
1.0
1.1

```
    0.00    1.00000000
    0.10    1.10517092
    0.20    1.22140276
    0.30    1.34985881
    0.40    1.49182470
    0.50    1.64872127
    0.60    1.82211880
    0.70    2.01375271
    0.80    2.22554093
    0.90    2.45960311
    1.00    2.71828183
```

Note:

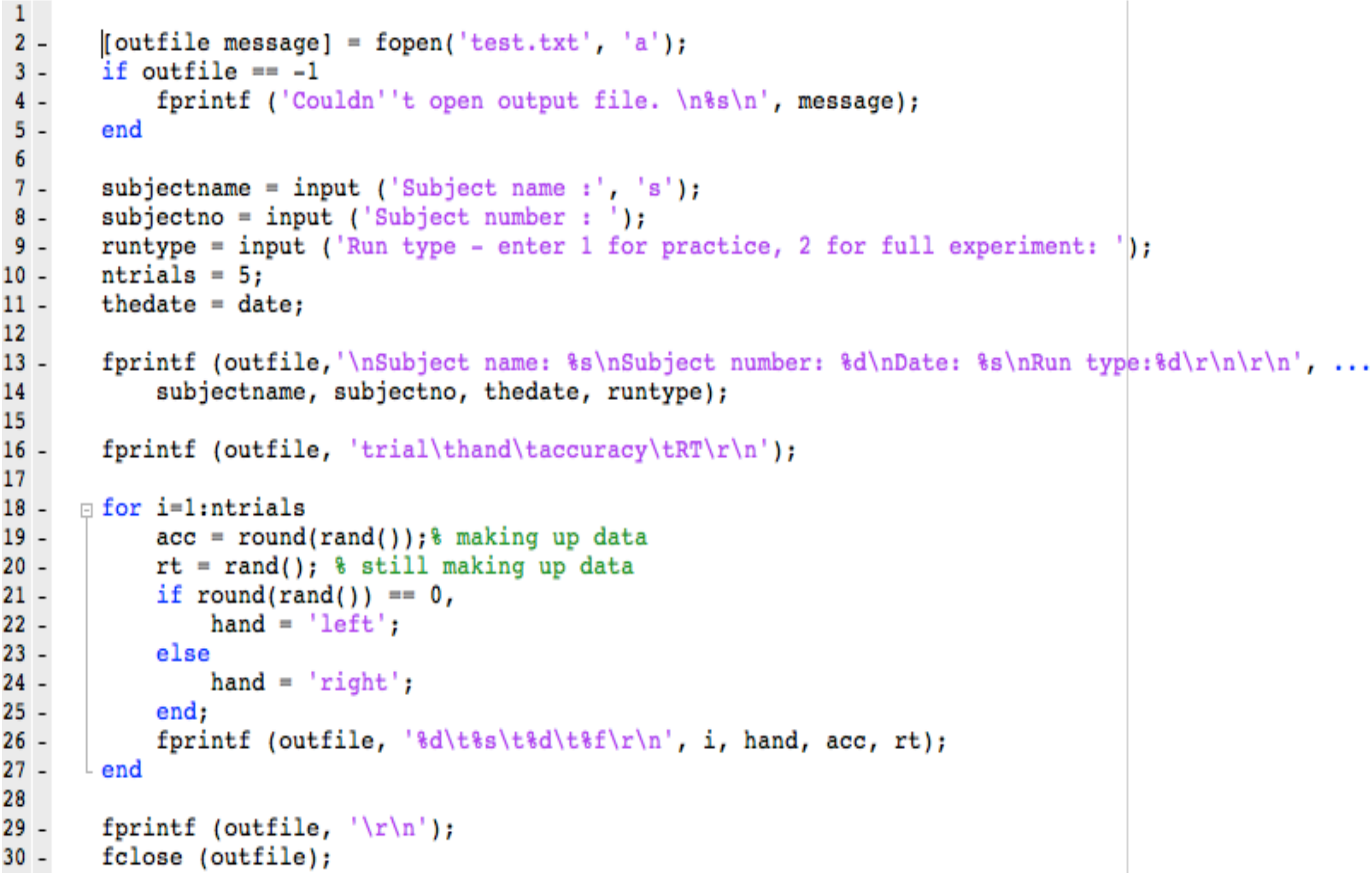
- MATLAB import functions, all UNIX applications, and Microsoft Word and WordPad recognize '\n' as a newline indicator.

- However, if you plan to read the file with Microsoft Notepad, use '\r\n' to move to a new line when writing.

- For example, replace the previous call to fprintf with the following:

```
    fprintf(fid, '%6.2f %12.8f\r\n', y);
```

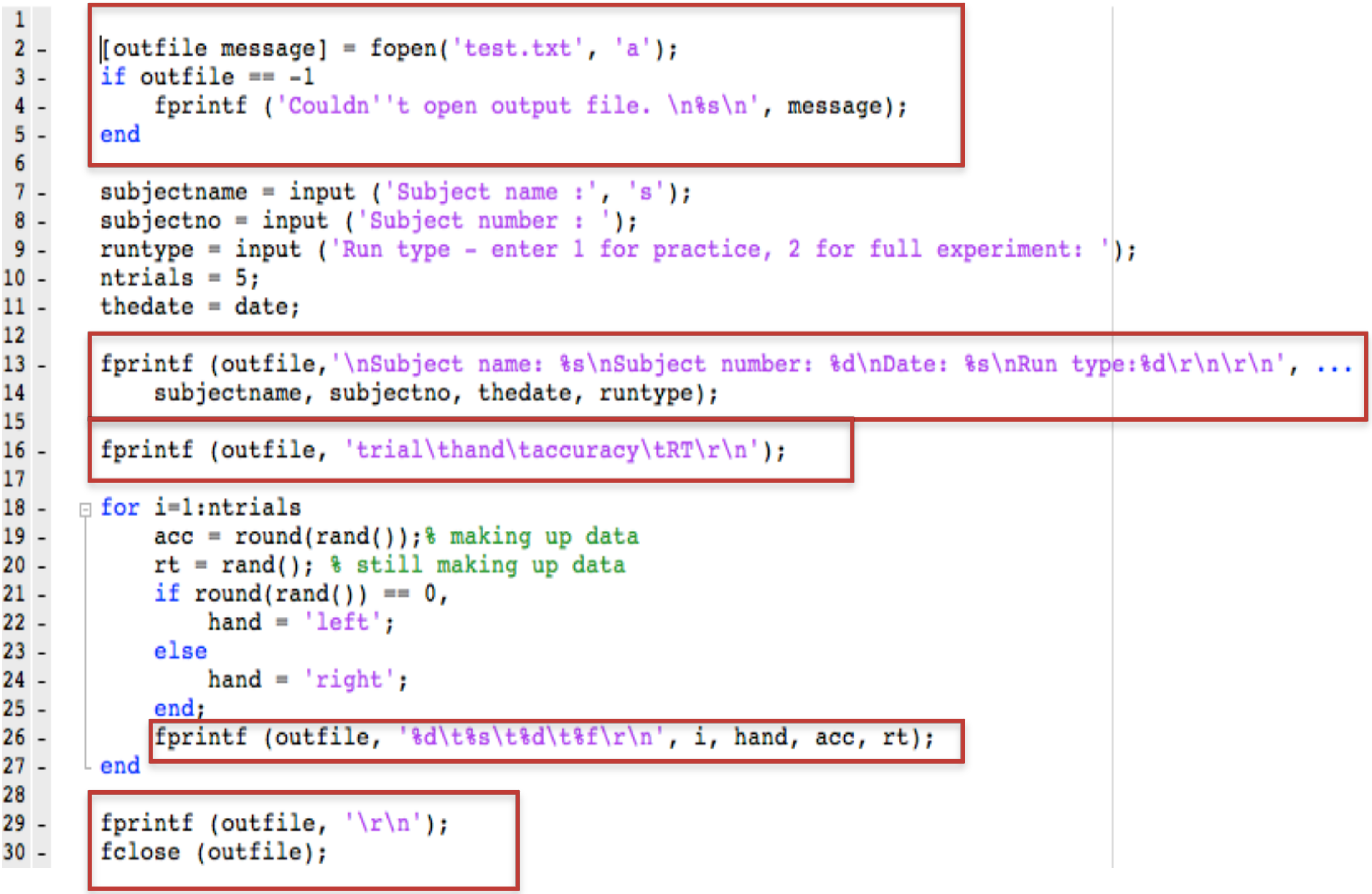
Example 2



```
[outfile message] = fopen('test.txt', 'a');
if outfile == -1
    fprintf ('Couldn''t open output file. \n%s\n', message);
end

subjectname = input ('Subject name :', 's');
subjectno = input ('Subject number : ');
runtype = input ('Run type - enter 1 for practice, 2 for full experiment: ');
ntrials = 5;
thedate = date;

fprintf (outfile,'\nSubject name: %s\nSubject number: %d\nDate: %s\nRun type:%d\r\n\r\n', ...
    subjectname, subjectno, thedate, runtype);

fprintf (outfile, 'trial\thand\taccuracy\tRT\r\n');

for i=1:ntrials
    acc = round(rand());% making up data
    rt = rand(); % still making up data
    if round(rand()) == 0,
        hand = 'left';
    else
        hand = 'right';
    end;
    fprintf (outfile, '%d\t%s\t%d\t%f\r\n', i, hand, acc, rt);
end

fprintf (outfile, '\r\n');
fclose (outfile);
```

BasicFile.m on the course website

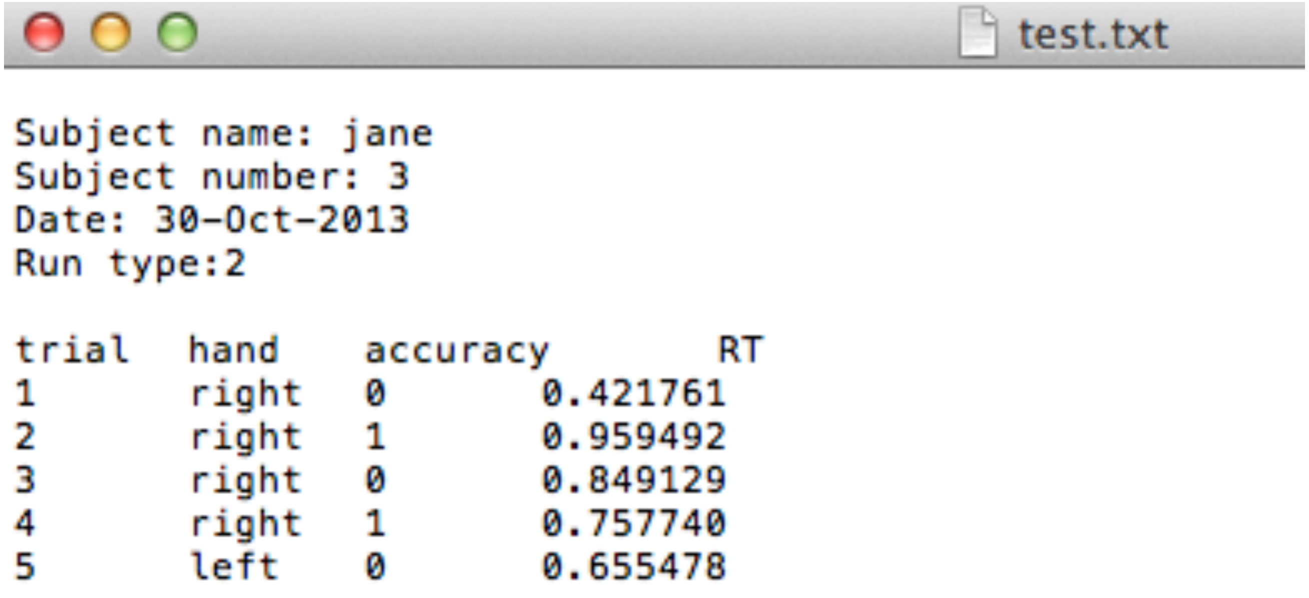
Example 2



```
[outfile message] = fopen('test.txt', 'a');
if outfile == -1
    fprintf ('Couldn''t open output file. \n%s\n', message);
end

subjectname = input ('Subject name :', 's');
subjectno = input ('Subject number : ');
runtype = input ('Run type - enter 1 for practice, 2 for full experiment: ');
ntrials = 5;
thedate = date;

fprintf (outfile,'\nSubject name: %s\nSubject number: %d\nDate: %s\nRun type:%d\r\n\r\n', ...
    subjectname, subjectno, thedate, runtype);

fprintf (outfile, 'trial\thand\taccuracy\tRT\r\n');

for i=1:ntrials
    acc = round(rand());% making up data
    rt = rand(); % still making up data
    if round(rand()) == 0,
        hand = 'left';
    else
        hand = 'right';
    end;
    fprintf (outfile, '%d\t%s\t%d\t%f\r\n', i, hand, acc, rt);
end

fprintf (outfile, '\r\n');
fclose (outfile);
```

BasicFile.m on the course website

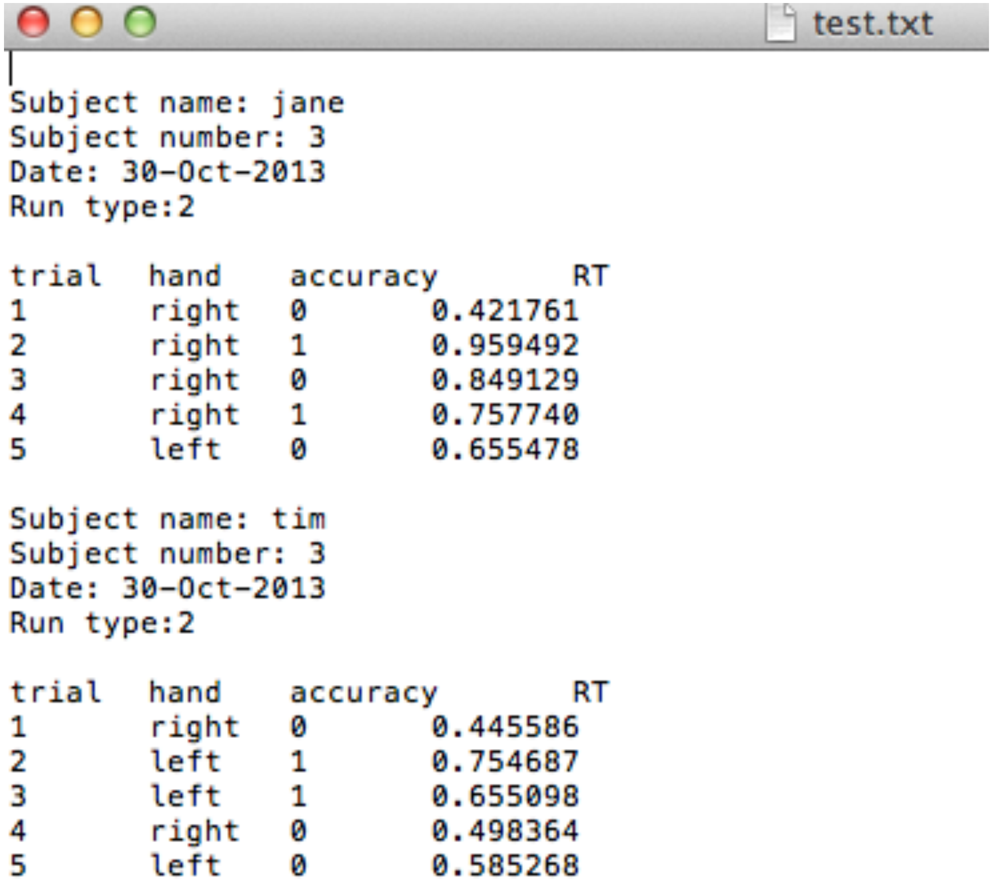
Output of BasicFile.m



```
Subject name: jane
Subject number: 3
Date: 30-Oct-2013
Run type:2

trial   hand    accuracy       RT
1       right   0       0.421761
2       right   1       0.959492
3       right   0       0.849129
4       right   1       0.757740
5       left    0       0.655478
```

Exercises:
Can you print only 2 decimal points for RT?
Can you move RT values for trials to the right?
Can you write only RT to a second file called test2.txt?

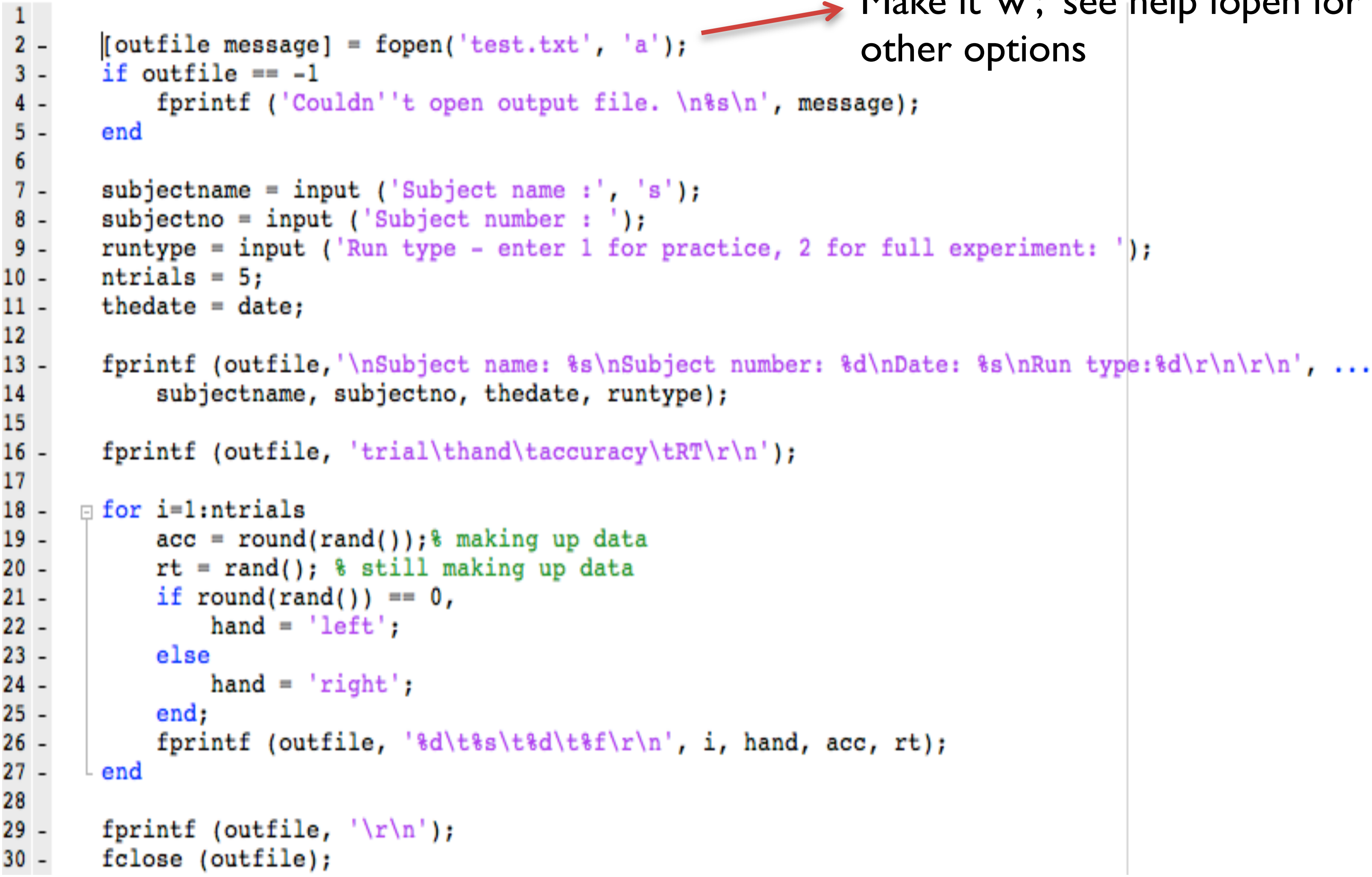
Run BasicFile.m for second time



```
Subject name: jane
Subject number: 3
Date: 30-Oct-2013
Run type:2

trial   hand    accuracy        RT
1       right   0       0.421761
2       right   1       0.959492
3       right   0       0.849129
4       right   1       0.757740
5       left    0       0.655478

Subject name: tim
Subject number: 3
Date: 30-Oct-2013
Run type:2

trial   hand    accuracy        RT
1       right   0       0.445586
2       left    1       0.754687
3       left    1       0.655098
4       right   0       0.498364
5       left    0       0.585268
```

Do you see why the output of the second run was appended to the first?

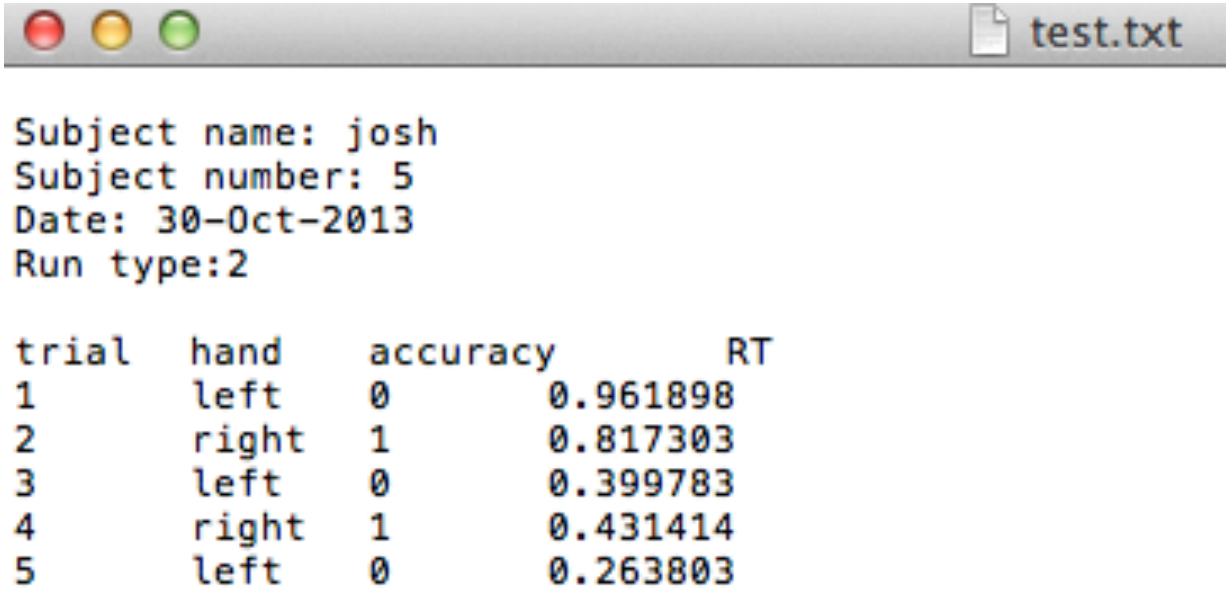
Now, let’s change the second parameter of fopen in line 2 in BasicFile.m



```
1
2 -    [outfile message] = fopen('test.txt', 'a');
3 -    if outfile == -1
4 -        fprintf ('Couldn''t open output file. \n%s\n', message);
5 -    end
6
7 -    subjectname = input ('Subject name :', 's');
8 -    subjectno = input ('Subject number : ');
9 -    runtype = input ('Run type - enter 1 for practice, 2 for full experiment: ');
10 -   ntrials = 5;
11 -   thedate = date;
12
13 -   fprintf (outfile,'\nSubject name: %s\nSubject number: %d\nDate: %s\nRun type:%d\r\n\r\n', ...
14          subjectname, subjectno, thedate, runtype);
15
16 -   fprintf (outfile, 'trial\thand\taccuracy\tRT\r\n');
17
18 -   for i=1:ntrials
19 -       acc = round(rand());% making up data
20 -       rt = rand(); % still making up data
21 -       if round(rand()) == 0,
22 -           hand = 'left';
23 -       else
24 -           hand = 'right';
25 -       end;
26 -       fprintf (outfile, '%d\t%s\t%d\t%f\r\n', i, hand, acc, rt);
27 -   end
28
29 -   fprintf (outfile, '\r\n');
30 -   fclose (outfile);
```

Make it ‘w’; see help fopen for other options

Now, look at the output file, test.txt



```
Subject name: josh
Subject number: 5
Date: 30-Oct-2013
Run type:2

trial   hand     accuracy        RT
1       left     0       0.961898
2       right    1       0.817303
3       left     0       0.399783
4       right    1       0.431414
5       left     0       0.263803
```

Do you see what happened? Why?