PRA1004 Scientific Computing — Example report 1

Your names and email

April 19, 2012

Note: this is an example report showing what I would like to see based on the new guidelines. As such, the filenames for the scripts are different from the ones indicates in the document with assignments.

This lab session gives an introduction to Matlab, a high level programming environment.

Assignment 1

We familiarized ourselves with Matlab, no questions need to be answered.

Assignment 2

Parts (1)–(3). We created `script2.m` in an appropriate directory.[1] It is important to change to the directory (using the `cd` command) the script is in, otherwise Matlab can't find it. (Actually, it would also be possible to edit the so-called 'path' but this is not required.)

Part (4). We implemented the script to perform the required computations. The important parts are the following:

```
% script2.m

% the following are in meters
length = 5;
heigth = 2;
depth = 7;

volume = length * heigth * depth %<- m^3
```

[1] Note that this is called 'script1' in the assignment. However, to keep things clear. From now on, the script for assignment X will be called scriptX.

```
n = 23; %<- number of people
app = volume / 23; % air per person

% Note: app is still in m^3, convert to liters
app_l = app * 1000
minutes = app_l / 8
```

Part (6). We implemented the required tasks (still in `script1.m`) . The important parts are as follows:

(a) print 1-25.

```
for i = [1:25]
    disp(i)
end
```

(b) print the powers of 2.

```
n=2;
while(n<100)
    disp(n);
    n = n * 2;
end
```

(c) print numbers divisible by 3 or 7.

```
for n = [1:50]
    divisible_by_3 = mod(n,3) == 0;
    divisible_by_7 = mod(n,7) == 0;
    divisible_by_3_or_7 = divisible_by_3 | divisible_by_7;
    if divisible_by_3_or_7 %<- of course we could have done all the testing here!
        disp(n)
    end
end
```

(d) print prime numbers.

```
for n = [1:50]
    f = factor(n);
    if numel(f) == 1 %<- 'numel' gives number of elements
        %(could also have used 'size')
        n_is_prime = 1; %<- n is prime if it has 1 factor
    else
        n_is_prime = 0;
    end
```

```
    if n_is_prime
        disp(n)
    end
end
```

Assignment 3

3.1 Functions in an M-file

We created `script3.m`. It demonstrates the use of the 'print_primes' function. The latter function is itself based on a new function that we called 'is_prime'.

This is the 'print_primes' function:

```
function print_primes(a, b)
% function print_primes(a, b) - NOTE: no output argument!
%
% this function print all primes between a and b
for n = [a:b]
    if is_prime(n)
        disp(n)
    end
end
```

This is the is_prime function:

```
function bool = is_prime(n)
%function bool = is_prime(n) - NOTE: 'bool' is the output argument!
%
% this function tests if n is a prime by making use of the
% matlab 'factor' command
f = factor(n);
if numel(f) == 1 %<- 'numel' gives number of elements
    bool = 1;
else
    bool = 0;
end

% return from the function (this is not crucial when putting each function
% in a seperate file)
return;
```

3.2 Anonymous Functions

Parts (1–3) The following was also implemented in script3:

```
% 3.2.1 defining the functions
```

```
sub = @(a,b) a-b;
mul = @(a,b) a*b;

% 3.2.2 copying
copy = sub;

% 3.2.3. swapping:
sub = mul;
mul = copy;

sub_test = sub(3,3)
mul_test = mul(3,3)
```

(4) The feval function can be used to call a anonymous function. This is especially useful when you are calling a anomous function of which you do not know in advance what it is going to be. For instance, in the Newton method algorithm described in the text book, two anonymous functions is given as an argument (for both f and f'). In this case feval can be used to evaluate the functions.

Assignment 4 (BONUS)

Omitted from example report, since it is part of the new assignment.