IAM 550
Midterm Exam
Fall 2019

No books. No cheat sheets. No cell phones. No computers.
Only pen and paper. You should not need extra paper (but there is some, in case).
Make sure your answers can be understood. Write your name on every page as indicated.

Printed Student Name: SOLUTION

Grading: pts / 100 pts, 115 max possible (Grader:)

---

1. [7 pts] Let x = [x1, x2, … …, xN] and y = [y1, y2, … …, yN] be two equal length vectors. Write down the MATLAB statement that forms the product vector z = [x1*y1, x2*y2, … …, xN*yN].

```
z = x .* y;
```

2. [7 pts] With the same input as above, write MATLAB **loop** that forms the scalar product s of the vectors x and y.

```
s=0;
for i=1:N
s=s+x(i)*y(i);
end
```

3. [7 pts] The police clocks you driving at 60 mph, while your speedometer says 57 mph. The police radar gun is correct (it always is). What is your speedometer's

    a. True error? 60 mph – 57 mph = 3 mph

    b. Relative error? 3 mph / 60 mph = 0.05

    c. Percentage error? 5%

Name:

4. [16 pts] Write a section of MATLAB code that first fills a vector C such that it has 20 equally spaced points from 11 to 30. Then, in a subsequent section **use a for-loop** to calculate a new vector, D. D(1) is to contain the first element of C divided by the second element of C, D(2) should contain the second element of C divided by the third element of C, and so on up to the highest possible element.

```
C=11:30;
for i=1:19
D(i)=C(i)/C(i+1);
end
```

5. [30 pts] Suppose you know how to fit a straight line $y(x)=a_0+a_1*x$ to a set of data points and you have a function to calculate a_0 and a_1 from the data, defined as `function [a0, a1] = fit(x,y)` , where `x` and `y` are vectors of length `N` with your data. Your data are expected to follow a power law $y(x)=Ax^B$. Write a section of MATLAB code that determines A and B using the function `fit`, assuming that `x` and `y` are known. Hint: you will need the log10() function here as well.

y=A*x^B → log(y) = log(A) + B*log(x), so there is a linear relationship between log(x) and log(y)

code:

```
xx=log(x);
yy=log(y);
[a0,a1]=fit(xx,yy);    % ← function that returns intercept and slope
A=10^a0;   % ← since the intercept is log(A)
B=a1;
```

Name:

6. [24 pts] The following MATLAB code is a function that is supposed to calculate a root with a given precision. There are 8 errors in the code, some are syntax errors (which MATLAB will complain about before the code even runs), others are runtime errors (which will crash the code while it runs), and logical errors, which will just give you a bad result or might even result in an infinite loop. Carefully circle the offending parts of the code and write on the margin or with a footnote what the mistake is. [3 pts each correctly marked error, 3 points deducted if something correct is marked as error]

```
function [root, nit] = bisect(F,a,b,precision);

   %  function to find a root within the interval [a,b] with bisection.
   %  F:  file handle for the function
   %  a,b  the initial interval
   %  precision:  the desired absolute precision of the result
   %  root:  the best estimate of the root
   %  nit:  number of iterations used

fprintf('input: a,b,precision:  %f  %f\n',a,b,precision);
   x1=a;
   x2=b;
   f1=f(x1);  %  ←  function name is case sensitive
   f2=F(x2);
   if f1 = 0; root=x1; return;  %  <←  missing 'end'
   while x2 > x1;
     xm=(x1-x2)/2;  %  ←  the nidpoint is 0.5*(x1+x2)
     fm=F(xm);
     if fm < 0
       x2 = xm;
     elseif  %  ← two errors:  elseif must be followed by a logical
%            expression and a simple 'else' would be the proper logic here
       x1 = xm;
     end
     nit = nit + 1;  %  ←  nit is not defined yet
     d = x2 – x1;
     if d < precision;
       root == 0.5 * (x1 + x2);   %  ←  simple '=' for an assignment
       return;
%    'end'  missing
   end
end
```

Name:

7. [9 pts] Name at least 3 methods that can be employed using MATLAB to find an approximation to a root of a given function:

    a. First method: graphical solution

    b. Second Method: bisection

    c. Third method: Newton-Raphson (also secant method)

8. [8 pts] When using the bisection method to find the root of a function, your initial interval is of length 1. What is the least number of iterations it takes to bracket the root to within at most 10^{-3}?

    a. 3
    b. 8
    c. 10 because 1 / 2^10 < 0.001 < 1 / 2^9 (interval becomes smaller by ½ every iteration)
    d. 14

9. [7 pts] What will this MATLAB code print?

```
clear;
a=5;  b=10;
fprintf(' a=%d  b=%d \n',a,b);
c = func1(1,2)
fprintf(' a=%d  b=%d  c=%d\n',a,b,c);

function z = func1(x,y)
   a=x*y;
   b=x+y;
   z=a+b;
end
```

```
a=5   b=10
a=5   b=10   c=5
```