# Table of Contents



```
% Aero 300 - lecuter HW 1 - Joshua Oates

% this script is both the writen questions and the matlab questions
% it includes tests to confirm that my answers are not incorrect
```

# section 0 - cleanup

```
close all;
clear all;
clc;
```

# section 1 - problem 1

```
% 1. (6 points each) Rewrite the following polynomials in nested form:
    %(a)
        a = @(x) 6*x.^3 -2*x.^2 -3*x + 1;
    %(b)
        b = @(x) 8*x.^5 -x.^4 -3*x.^3 + x.^2 -3*x + 1;
    %(c)
        c = @(x) 4*x.^6 -2*x.^4 -2*x + 4;

% answers
%(a)
    A = @(x) 1 + x.*( -3 + x.*( -2 + x.* 6 ));
%(b)
    B = @(x) 1 + x.*( -3 + x.*( 1 + x.*( -3 + x.*( -1 + x.*8 ))));
%(c)
    C = @(x) 4 + x.*( -2 + x.*( 0 + x.*( 0 + x.*( -2 + x.*( 0 + x.*(4))))));
% ^^ C(x) = 4 + x*( -2 + (x^3)*( -2 + (x^2)*4)) ^^

% test cases
x = rand(1,10)*10;
TestA = round(A(x),5) == round(a(x),5);
TestB = round(B(x),5) == round(b(x),5);
TestC = round(C(x),5) == round(c(x),5);
disp("TestA"+TestA)
disp("TestB"+TestB)
disp("TestC"+TestC)

% clean
clear all;
```

```
% 2. (12 points) Explain how to evaluate the polynomial below for a given
 input x, using as few operations
% as possible. How many multiplications and additions are required?
%     p(x) = a0 + a6*x^6 + a12*x^12 + a18*x^18 .

p = @(x,a0,a6,a12,a18) a0 + a6 * x^6 + a12 * x^12 + a18 * x^18;

% answer
% use storage method and make X = x^6
% use X in nested method

x6 = @(x,i) x(i)^6 ;% = x*x*x*x*x*x ( six multiplication )
P = @(x6,a0,a6,a12,a18) a0 + x6 * (a6 + x6 * (a12 + x6 * (a18))); % ( three
 addition , three multiplication )

% the minimum number of operations is twelve ( three addition , nine
 multiplication )

% test cases
a = round(rand(4,10),4);
x = round(rand(1,10),4);

for i=1:1:length(a)
    TestP(i) = (  round( p(x(i),a(1,i),a(2,i),a(3,i),a(4,i)), 4) ==
 round( P(x6(x,i),a(1,i),a(2,i),a(3,i),a(4,i)) , 4)  );
end
disp("TestP"+TestP)

clear all;
```

```
  Columns 1 through 5

    "TestAtrue"    "TestAtrue"    "TestAtrue"    "TestAtrue"    "TestAtrue"

  Columns 6 through 10

    "TestAtrue"    "TestAtrue"    "TestAtrue"    "TestAtrue"    "TestAtrue"

  Columns 1 through 5

    "TestBtrue"    "TestBtrue"    "TestBtrue"    "TestBtrue"    "TestBtrue"

  Columns 6 through 10

    "TestBtrue"    "TestBtrue"    "TestBtrue"    "TestBtrue"    "TestBtrue"

  Columns 1 through 5

    "TestCtrue"    "TestCtrue"    "TestCtrue"    "TestCtrue"    "TestCtrue"

  Columns 6 through 10
```

```
    "TestCtrue"    "TestCtrue"    "TestCtrue"    "TestCtrue"    "TestCtrue"

  Columns 1 through 5

    "TestPtrue"    "TestPtrue"    "TestPtrue"    "TestPtrue"    "TestPtrue"

  Columns 6 through 10

    "TestPtrue"    "TestPtrue"    "TestPtrue"    "TestPtrue"    "TestPtrue"
```

# section 2 - problem 2

```
% (a) (15 points) Use the function nest.m from class (and now on Canvas ) to
 evaluate the polynomial
    % p(x) = 1 #x + x^2 #x^3 + . . . + x^98 #x^99
% at the point x0 = 1.00001 . Call your result r1 .
x0 = 1.00001;

for i = 1:1:100
    if mod(i,2) == 1
        coeffs(i) = 1;
    else
        coeffs(i) = -1;
    end
end
r1 = nest(99,coeffs,x0);

% (b) (7 points) It can be shown that a simpler, equivalent expression for the
 polynomial p is given by
p = @(x) (1 - x^100)/(1 + x);
% Use Equation (1) to evaluate p at the same x0 as above. Call your result
 r2 .
r2 = p(x0);

% (c) (8 points) Estimate the difference err = |r1 #r2| between your answers
 from parts (a) and (b)
% and comment on your result.
err = abs(r1 -r2);
disp (err);

% the err is ~1.7130e-16 which is in the same order of magnitude as Epsilon
 Mach
% this result is not supprising as any function that deals with small
% numbers will experience an error around  10^-16


%%section 3 - problem 3
clear all;

% 1. (6 points each) Find the binary representation of the following base 10
 numbers:
% (a) 35/16 = 2.1875;
```

```
% R is remainder

% iPart = 2
% 2/2 = 1 R 0
% 1/2 = 0 R 1

% fPart = .1875
% .1875 * 2 = .375 + 0
% .375 * 2 = .75 + 0
% .75 * 2 = .5 + 1
% .5 * 2 = 0 + 1

%N in binary representation N2 = 10.0011


% (b) 79.6
% iPart = 79
% 79/2 = 39 R 1
% 39/2 = 19 R 1
% 19/2 = 9 R 1
% 9/2 = 4 R 1
% 4/2 = 2 R 0
% 2/2 = 1 R 0
% 1/2 = 0 R 1

% fPart = .6
% .6 * 2 = .2 + 1
% .2 * 2 = .4 + 0
% .4 * 2 = .8 + 0
% .8 * 2 = .6 + 1
% ...

%N in binary representation N2 = 1001111.1001100110011001...


% 2. (8 points) Obtain the binary representation of the mathematical constant
 e .
% Hint : You may take e #2.718 .

% iPart = 2
% 2/2 = 1 R 0
% 1/2 = 0 R 1

% fPart = .718
% .718 * 2 = .436 + 1
% .436 * 2 = .872 + 0
% .872 * 2 = .744 + 1
% .744 * 2 = .488 + 1
% .488 * 2 = .976 + 0
% .976 * 2 = .952 + 1
% .952 * 2 = .904 + 1
% .904 * 2 = .808 + 1
% .808 * 2 = .616 + 1
% .616 * 2 = .232 + 1
```

```
% .232 * 2 = .464 + 0
% .928 * 2 = .856 + 1
% .856 * 2 = .712 + 1
% .712 * 2 = .424 + 1
% .424 * 2 = .848 + 0
% ...

% N in binary representation N2 ~= 10.101101111101110


% 3. (10 points each) Using the Rounding to Nearest Rule, determine the
 normalized floating point
% representation for each of the following decimal numbers and give their
 corresponding machine
% representation.
% (a) 9.5 ; (b) #3.6 .

% (a)
% iPart = 9 =>
% 1 0 0 1

% fPart = .5
% .5 * 2 = 0 + 1
% N in binary representation N2 = 1001.1 = 1.0011 * 2^-3
% p = -3 , p' = 1020
% 1020/2 = 510 R 0 ...
% 255 R 0
% 127 R 1
% 63 R 1
% 31 R 1
% 15 R 1
% 7 R 1
% 3 R 1
% 1 R 1
% 0 R 1

% e = [01111111100]
% m = [0011000...0] (48 zeros)
% s = 0
% N in machine repesentation is 0|01111111100|0011000...0 (48 zeros)

% (b)
% iPart = 3 =>
% 1 1
% fPart = .6 =>
% 1 0 0 1 | 1 0 0 1 | 1 0 0 1 ...

% N in binary representation N2 = -11.1001100110011001... =
% -1.11001100110011001... * 2^-1
% p = -1 , p' = 1022
% 1022/2 = 511 R 0 ...
% 255 R 1
% 127 R 1
% 63 R 1
```

```
% 31 R 1
% 15 R 1
% 7 R 1
% 3 R 1
% 1 R 1
% 0 R 1

% e = [01111111110]
% m = [1100|1100|1100...1100]1100 =>[1100|1100|1100...1101]
% s = 1
% N in machine repesentation is 1|01111111110|[1100|1100|1100...1101] (13 sets
 of 4 bits)
```

```
    1.7130394325270 97e-16
```

*Published with MATLAB® R2022a*