CS4 Final Exam (Practice)

Contents

Introduction

The purpose of this exam is help get you used to the format. Topics covered in this practice exam are not comprehensive. Additional practice problems will be covered in the 5/4 and 5/5 Review Sessions.

Target completion time for this exam is 90 minutes. But you will have twice that amount of time.

Practice problems from ITC and D.Y. Fan and C.F. Van Loan. Used by permission.

1) Basics: Code Understanding

What is the output when the following script is executed?

```
x = 10;
y = 20;
NumberOfTimesThruLoop = 1;
while x+y<=100
    z = x+y;
    x = y;
    y = z + NumberOfTimesThruLoop; fprintf('%3d \n',y)
end
```

Solution:

```
31
52
84
```

2) Basics: For and While

Assume that the value of the variable n is a positive integer. Rewrite the following fragment so that it uses a while-loop and produces exactly the same output.

```
for k=3:4:n
    fprintf('k = %3d\n',k)
end
```

Solution:

```
k = 3;
while k<=n
    fprintf('k = %3d\n',k)
    k = k+4;
end
```

3) Basics: Functions

Consider the following function:

```
function [u,v] = AddAndSubtract(x,y)
v = x+y;
u = y-x;
fprintf('u = %2d v = %2d x = %2d y = %2d\n',u,v,x,y)
```

What is the output when following script is executed? Show your work.

```
u = 30;
v = 10;
[y,x] = AddAndSubtract(u-v,u+v);
fprintf('u = %2d v = %2d x = %2d y = %2d\n',u,v,x,y);
```

Solution:

The script calls AddAndSubtract with input variables x = 20 and y = 40. Local variable v is assigned the value 60 and u gets 20. The first line of output is therefore

```
20 60 20 40
```

In the script, the function call assigns 20 to y and 60 to x. Thus, the second line of output is

```
30 10 60 20
```

4) Recursion

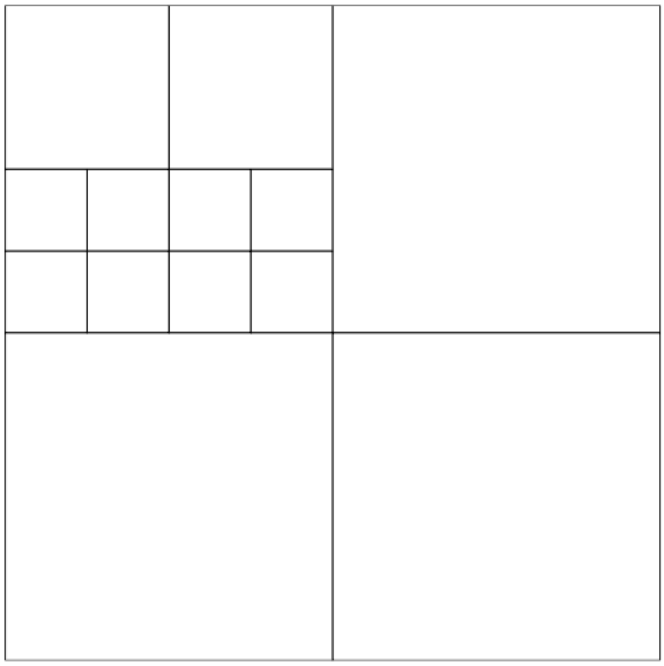
Consider the following function

```
function DrawOrPartition(a,b,s)
disp('Calling Function')
r = rand(1);
if s<=.125 || r<.5
   plot([a a+s a+s a a],[b b b+s b+s b],'k')
else
   DrawOrPartition(a,b,s/2);
   DrawOrPartition(a+s/2,b,s/2);
   DrawOrPartition(a,b+s/2,s/2);
   DrawOrPartition(a+s/2,b+s/2,s/2);
end
```

Notice that the message 'Calling Function' is printed in the Command Window every time DrawOrPartition is called. Now suppose that the script

```
figure
axis equal off
hold on DrawOrPartition(0,0,1)
```

is run and that it produces the following figure:



How many times is the message Calling Function? printed in the Command Window? You must justify your answer to receive full credit. Hint: Draw a tree that depicts the subdivision process.

Solution:

17 is the answer = total number of squares in the image, reasonable divide-and-conquer chart

5) Iterative Calculation

The area of the largest regular polygon with n sides that fits inside the unit circle is given by

$$A_n = n\frac{c}{\sqrt{1-c^2}}$$

where

$$c = \cos(\frac{\pi}{n})$$

Since $\cos(\pi/3) = 1/2$, we see that

$$A_3 = \frac{3\sqrt{3}}{4}$$

is the area of the largest equilateral triangle that fits inside the unit circle. Using the formula

$$\cos(\frac{\theta}{2}) = \sqrt{\frac{1 + cos(\theta)}{2}}$$

write a script that assigns to `A` the value of An where $n = 3 \cdot 2^{10}$. To receive full credit, your solution must make effective use of a for-loop and not make use of any built-in Matlab function.

Solution:

```
c = 1/2;
for k=1:10
    c = sqrt((1+c)/2);
end
A = 3*2^10 *c/sqrt(1-c*c);
```

6) Geometry and use of rand

The following script displays 100 randomly positioned unit squares in the figure window:

```
close all figure
axis equal off hold on
a = rand(1);
b = rand(1);
for k=1:1000
    x = rand(1);
    y = rand(1);
    plot([x x+1 x+1 x x],[y y y+1 y+1 y],?k?)
end
```

Modify the loop body so that it only displays squares that include the point (a,b). The displayed square should be red with probability .2, green with probability .5, and blue with probability .1.

Solution:

```
x = rand(1);
y = rand(1);
if x<=a && a<=x+1 && y<=b && b<=y+1
    rn = rand(1)
    if rn<.2
        c = 'r'; elseif rn<=.7
        c = 'g'; elseif rn<=.8
        c = 'b'; end
    plot([x x+1 x+1 x x],[y y y+1 y+1 y],c)
end
```

7) Use of Struct, Unique Combinations

Assume the availability of the following function

```
function P = MakePoint(x,y)
% x and y are real numbers and P represents the point
% with Cartesian coordinates (x,y).
% P.x is the x-coordinate and P.y is the y-coordinate
P = struct(?x?,x,?y?,y);

function T = MakeTriangle(A,B,C)
% T represents a triangle with vertices A, B, and C
% that are distinct points.
T = struct(?A?,A,?B?,B,?C?,C);
```

(a) Suppose F is a triangle. Complete the following command so that F is displayed as a colored blue triangle:

```
fill([____________________], [____________________],'b')
```

Solution:

```
fill([F.A.x F.B.x F.C.x ], [F.A.y F.B.y F.C.y],'b')
```

(b) Complete the following function so that it performs as specified: function T = ListOfTriangles(P) % P is a length-n structure array of points. It is assumed that % P(1),...,P(n) are distinct and situated on the unit circle. % T is a

structure array consisting of all possible triangles that have % vertices chosen from P(1),...,P(n). Each triangle in T must have % positive area and no two triangles in T should be the same.

Solution:

```
n = length(P);
q = 0;
for i=1:n
    for j=i+1:n for k=j+1:n
            q = q+1;
            T(q) = MakeTriangle(P(i),P(j),P(k));
        end
    end
end
```

8) Use of Struct, Unique Combinations

(a) Implement the following function: function z = overlap(diskA, diskB) % diskA and diskB are each a disk structure with the following fields: % x: x-coordinate of center of disk % y: y-coordinate of center of disk % radius: radius of disk

Solution:

```
dis = sqrt((diskA.x - diskB.x)^2 + (diskA.y - diskB.y)^2);
if dis < diskA.radius + diskB.radius
    z = 1;
else
    z = 0;
end
```

(b) Implement the following function to return the indices of disk triplets that overlap. Three disks form a triplet if every disk overlaps with each of the other two. Make effective use of function overlap from part (a). Your code should be effcient - avoid unnecessary iterations. State your runtume as a function length(D).

```
function idx = diskTriplets(D)
% D is a 1-d array of disk structures; each structure has fields as
% defined in part (a). Assume D has a length greater than 3. idx is a
% vector of indices indicating all triplet overlap combinations. For
% example, if disks 2, 4, and 5 form a triplet and disks 3, 4, and 6
% form a triplet, idx should be the vector [2 4 5 3 4 6]. Other
% orderings of triplets are acceptable, however each triplet should
% only appear once.
```

Solution:

```
n = length(D);
idx = [];
for i = 1:n - 2
    for j = i + 1:n - 1
        for k = j + 1:n
            if overlap(D(i), D(j)) && ...
                    overlap(D(j), D(k)) && overlap(D(i), D(k))
                idx = [idx, i, j, k];
            end
        end
    end
end
```

Runtime is C(length(D),3).

9) Index Understanding

Complete the following function so that it performs as specified:

```
function [vMax,j] = maxV(v)
% v is a vector with distinct values.
% vMax is the maximum value in v and j is an integer with the
% property that the value of v(j) is vMax.
```

You are NOT allowed to use the built-in function max.

Solution:

```
n = length(v);
vMax = v(1);
j = 1;
for i=2:n
    if v(i)>vMax
        vMax = v(i); j = i;
    end
end
```

(b) Assume that vMax is correctly implemented and available. Complete the following function so that it performs as specified:

```
function [AMax,p,q] = maxA(A)
% A is a matrix with distinct values.
% AMax is the maximum value in A and p and q are integers with the
% property that the value of A(p,q) is AMax.
```

Your implementation must make effective use vMax and must NOT use the built-in function max.

Solution:

```
[m,n] = size(A);
[Amax,p] = vMax(A(:,1);
for j=2:n
    % Can we do better in column j? [vMax,i] = vMax(A(:,j));
    if vMax >AMax
        % Remember the new maximum AMax = vMax;
        q = j;
        p = i;
    end
end
```