Quiz # 3 **Name:** Solutions
Date: 2/3/2022
Math 124: Spring 2022

Matrix Multiply (20 Points)

In this problem you will implement matrix-matrix multiply two ways, using iteration and recursion. For all the problems we will assume that the matrices are of size $2^k \times 2^k$ for some integer $k \geq 0$. Recall also that matrix-matrix multiplication is defined asas

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Problem 1 (7 points)

Write a Julia function that takes in two input matrices A and B and returns the product $C = AB$ using for loops. You may only do scalar multiplication and addition, and may not use any built in functionality that performs matrix-matrix or matrix-vector multiplication.

```
function matmatmul(A,B)
  n= size(A)[1]
  C=zeros(n,n)
  for i=1:n
      for j=1:n
        for k=1:n
          C[i,j] += A[i,k] * B[k,j]
        end
      end
  end

  return C
end
```

FUN FACT:

This takes n^3 operations

Now you will implement matrix-matrix multiply recursively. Do this by splitting the matrices as

$$C = AB = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$$

where A is size $n \times n$ and each of the four submatrices is size $\frac{n}{2} \times \frac{n}{2}$.

Problem 2 - 2 Points

What is the base case?

When $n=1$, then $C=AB$ is just scaler multiplication.

Problem 3 - 3 Points

Write out the expression for each submatrix C_{ij} in terms of the submatrices A_{ij}, B_{ij}.

$C_{11} = A_{11}B_{11} + A_{12}B_{21}$ $\quad$ $C_{12} = A_{11}B_{12} + A_{12}B_{22}$

$C_{21} = A_{21}B_{11} + A_{22}B_{21}$ $\quad$ $C_{22} = A_{21}C_{12} + A_{22}C_{22}$

Problem 4 - 8 Points

Write Julia code to implement matrix-matrix multiply recursively using this splitting method.

```
func recur(A,B)
    n=size(A)[1]
    C=zeros(n,n)
```

Base Case

```
    if n==1
        return A*B
    end
```

Generate Submatrices

```
    A11 = A[1:n/2, 1:n/2]; B11 = B[1:n/2, 1:n/2]
    A12 = A[1:n/2, n/2+1:end]; B12 = B[1:n/2, n/2+1:end]
    A21 = A[n/2+1:end, 1:n/2]; B12 = B[n/2+1:end, 1:n/2]
    A22 = A[n/2+1:end, n/2+1:end]; B22 = B[n/2+1:end, n/2+1:end]
```

Recursive calls

```
    C[1:n/2, 1:n/2] = recur(A11,B11) + recur(A12,B21)
    C[1:n/2, n/2+1:end] = recur(A11,B12) + recur(A12,B22)
    C[n/2+1:end, 1:n/2] = recur(A21,B11) + recur(A22,B21)
    C[n/2+1:end, n/2+1:end] = recur(A21,B12) + recur(A22,B22)
    return C
end
```