Pr. 1.
Let $\boldsymbol{A}$ be an $M \times N$ matrix having elements: $a_{ij} = i\, 2^j$, for $i = 1, \ldots, M,\ j = 1, \ldots, N$.

(a) Express matrix $\boldsymbol{A}$ mathematically as an **outer product** of two appropriately defined vectors.

(b) Write (on paper) a one-line `Julia` expression for creating $\boldsymbol{A}$ when M and N are defined already. Test your `Julia` expression (just for yourself, not graded).

Pr. 2.
Rewrite the expression

$$y = \sum_{i=1}^{N} \sum_{j=1}^{N} x_i^* a_{ij} x_j$$

in terms of the $N \times N$ matrix $\boldsymbol{A}$ and the vector $\boldsymbol{x} \in \mathbb{F}^N$ whose ith entry is x_i.
Check your answer using `Julia` by trying some random examples.
Hint. Consider $\boldsymbol{u}'\boldsymbol{v} = \sum_i u_i^* v_i$. The left hand side is an expression involving vectors, whereas the RHS is an expression involving elements of those vectors. We often need to go back and forth between these two types of expressions.

Pr. 3.
Vectors $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ are all in $\mathbb{R}^N$ and we want to compute `z*u'*v*x'*y`

(a) Rewrite the code with parentheses to ensure the computation is as efficient as possible.

(b) Exactly how many scalar multiplication operations are needed? Explain.

Pr. 4.
Let $\boldsymbol{\Sigma}$ be an $M \times N$ diagonal matrix. Let the diagonal elements of $\boldsymbol{\Sigma}$ be denoted by $\sigma_1, \sigma_2, \ldots$.
Let $\boldsymbol{U}$ and $\boldsymbol{V}$ be $M \times M$ and $N \times N$ matrices, respectively. Define $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}'$.

(a) Suppose $M < N$: Express $\boldsymbol{A}$ as a sum of **outer product** matrices where each outer-product matrix is formed from the columns of $\boldsymbol{U}$ and $\boldsymbol{V}$. How many such outer-product matrices add up to form $\boldsymbol{A}$?

(b) Suppose $M > N$: Express $\boldsymbol{A}$ as a sum of outer-product matrices where each outer-product matrix is formed from the columns of $\boldsymbol{U}$ and $\boldsymbol{V}$. How many such outer-product matrices add up to form $\boldsymbol{A}$?

(c) Generalize the answer above: what is the maximum number of outer-product matrices formed from the columns of $\boldsymbol{U}$ and $\boldsymbol{V}$, for $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}'$, that can add up to form $\boldsymbol{A}$?

Pr. 5.
Let $\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots, \boldsymbol{U}_k \in \mathbb{F}^{N \times N}$ denote **unitary** matrices. Show that the product $\boldsymbol{U}_1 \boldsymbol{U}_2 \cdots \boldsymbol{U}_k$ is a unitary matrix. Because of this property, the set of $N \times N$ unitary matrices over a field $\mathbb{F}$ is a group called the **orthogonal group** denoted $\mathsf{O}(N, \mathbb{F})$.

Pr. 6.
Use the defining **determinant** properties $\det(\boldsymbol{A}) = \det(\boldsymbol{A}^\top)$ and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{N \times N} \Rightarrow \det(\boldsymbol{A}\boldsymbol{B}) = \det(\boldsymbol{A}) \det(\boldsymbol{B})$ to solve the following problem. If $\boldsymbol{A}$ is **orthogonal**, what are the possible values of $\det(\boldsymbol{A})$?

Pr. 7.
For 30 points, complete the "`Julia` 101" tutorial at `https://pathbird.com/courses/register`. (Use the registration code shown on the ECE 551 Canvas home page.) When you reach the end of the tutorial, you will be able to print a concise version (as a pdf file) that shows all your correct answers. Submit that pdf to gradescope. This is an "all or nothing" problem: you earn the credit if and only if you reach the end of the tutorial and submit properly.
The introduction you see there might mention "opportunities to work with Python;" that comment applies to ECE 505, not ECE 551.

Pr. 8.

(a) For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}^N$, prove that $\det(\boldsymbol{I} - \boldsymbol{x}\boldsymbol{y}') = 1 - \boldsymbol{y}'\boldsymbol{x}$.

Hint. Use properties in §2.6, especially §2.6.4.

(b) Express $\det(\lambda \boldsymbol{I}_N - \boldsymbol{x}\boldsymbol{y}')$ in terms of λ, $\boldsymbol{y}'\boldsymbol{x}$, and N.

Hint. Consider writing $\lambda \boldsymbol{I}_N - \boldsymbol{x}\boldsymbol{y}' = (\lambda \boldsymbol{I}_N)\left(\boldsymbol{I}_N - \frac{\boldsymbol{x}\boldsymbol{y}'}{\lambda}\right)$ when $\lambda \neq 0$. Also consider the $\lambda = 0$ case.

(c) Use the previous step to find *all* **eigenvalues** of the matrix $\boldsymbol{x}\boldsymbol{y}'$.

(d) When are the eigenvalues of the matrix $\boldsymbol{x}\boldsymbol{y}'$ all equal to 0 even when the matrix is not equal to zero?

Pr. 9.

For $\boldsymbol{A} \in \mathbb{F}^{N \times N}$, the **trace** of $\boldsymbol{A}$, denoted by $\mathrm{Tr}(\boldsymbol{A})$, is defined as the sum of its diagonal elements: $\mathrm{Tr}(\boldsymbol{A}) = \sum_{i=1}^{N} a_{ii}$.

(a) Show that the trace is a linear function; i.e., if $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{F}^{N \times N}$ and $\alpha, \beta \in \mathbb{F}$, then $\mathrm{Tr}(\alpha \boldsymbol{A} + \beta \boldsymbol{B}) = \alpha \,\mathrm{Tr}(\boldsymbol{A}) + \beta \,\mathrm{Tr}(\boldsymbol{B})$.

(b) Show that $\mathrm{Tr}(\boldsymbol{A}\boldsymbol{B}) = \mathrm{Tr}(\boldsymbol{B}\boldsymbol{A})$, even though in general $\boldsymbol{A}\boldsymbol{B} \neq \boldsymbol{B}\boldsymbol{A}$, when both $\boldsymbol{A}\boldsymbol{B}$ and $\boldsymbol{B}\boldsymbol{A}$ are square. Your work should be general enough to handle cases where $\boldsymbol{A}$ and $\boldsymbol{B}$ are rectangular.

(c) Let $\boldsymbol{S} \in \mathbb{R}^{N \times N}$ be **skew-symmetric**, i.e., $\boldsymbol{S}^T = -\boldsymbol{S}$. Show that $\mathrm{Tr}(\boldsymbol{S}) = 0$.

(d) Prove the converse of the previous part, or provide a counterexample.

Pr. 10.

A matrix $\boldsymbol{A} \in \mathbb{F}^{N \times N}$ is called **idempotent** iff $\boldsymbol{A}^2 = \boldsymbol{A}$.

Show that the matrix $\boldsymbol{A} = \frac{1}{2}\begin{bmatrix} 2\cos^2(\theta) & \sin(2\theta) \\ \sin(2\theta) & 2\sin^2(\theta) \end{bmatrix}$ is idempotent for all θ.

Hint. Verify that $\boldsymbol{A} = \boldsymbol{v}\boldsymbol{v}'$ where $\boldsymbol{v} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$.

Pr. 11.

(a) Determine by hand the **eigenvalues** of $\boldsymbol{A} = \begin{bmatrix} 6 & 16 \\ -1 & -4 \end{bmatrix}$.

(b) Compute the **determinant** and **trace** of this matrix $\boldsymbol{A}$.
Compute the product of the eigenvalues of $\boldsymbol{A}$ and the sum of its eigenvalues.
How do these compare to the determinant and the trace?

(c) Check your answers by invoking:

```
using LinearAlgebra
lambda, V = eigen(A)
```

Are the eigenvectors (columns of $\boldsymbol{V}$) orthogonal?
Display the matrix $\boldsymbol{V}'\boldsymbol{V}$ and submit a screenshot of the answer.

(d) (not graded)
Some software sorts the eigenvalues in a certain order. What order does `Julia` return for symmetric matrices?

Pr. 12.
Discrete time **convolution** (a DSP term) of an input signal $x[n]$ and a filter $h[n]$ is defined in textbooks as:

$$y[n] = (h * x)[n] = (x * h)[n] \Rightarrow y[m] = \sum_{k=-\infty}^{\infty} h[k]x[m-k] = \sum_{n=-\infty}^{\infty} x[n]h[m-n],$$

where the "$*$" above denotes convolution, not ordinary multiplication.
In practice we often use finite-length signals and filters, and not all textbooks define this case clearly. The convolution of an input signal $(x[0], \ldots, x[N-1])$ with a finite impulse response (FIR) filter $(h[0], \ldots, h[K-1])$ becomes:

$$y[m] = \sum_{k=\max(0,m-N+1)}^{\min(K-1,m)} h[k]x[m-k] = \sum_{n=\max(0,m-K+1)}^{\min(N-1,m)} x[n]h[m-n], \quad m = 0, \ldots, M-1.$$

This is the type of convolution used in **convolutional neural network** (CNN) models for machine learning.
We will use the notation $\boldsymbol{y}, \boldsymbol{h}, \boldsymbol{x}$ to denote the (finite) vectors of length M, K, and N, respectively.

(a) For the specific case where the input $\boldsymbol{x}$ is [3 3 3] and the filter $\boldsymbol{h}$ is [2 2], determine what the output $\boldsymbol{y}$ is. (You may do this by hand or with any software you want to use.) Hint: `(` sum(y) = 36).

(b) Determine M, the length of the (possibly) nonzero part of $\boldsymbol{y}$, for the general case, in terms of $N \geq 1$ and $K \geq 1$.

(c) Convolution is a linear operation, so we can express it as a **matrix-vector operation** of the form $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$ where $\boldsymbol{H}$ is a matrix that you must determine and implement in this problem.
With some recycling of notation, you must implement a matrix $\boldsymbol{H}$ such that $\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x} = \boldsymbol{H}\boldsymbol{x}$
Caution: DSP notation uses signals that start at $n = 0$, whereas the first element of a vector in `Julia` (and `MATLAB`) is indexed by 1. This is something you must get used to handling properly.
Hint: It may be helpful to think through carefully the most reasonable size of $\boldsymbol{H}$ first. You should not augment $\boldsymbol{x}$, nor have any rows that are identically zero in $\boldsymbol{H}$.
Write a function called `convolution` that takes as its input the vectors $\boldsymbol{h}$ and $\boldsymbol{x}$ and returns as output both the matrix $\boldsymbol{H}$ and the column vector $\boldsymbol{y}$ (the convolved signal, implemented without `conv`).
Hint: Compare your answer with a built-in convolution function: `Julia`'s `conv` function (in the `DSP` package).
In `Julia`, your file should be named `convolution.jl` and should contain the following function:

```
"""
    H, y = convolution(h, x)

Compute discrete convolution of the input vectors `h` and `x`
via matrix multiplication,
returning both the matrix `H` and result `y`.

# In:
- `h` vector of length `K`
- `x` vector of length `N`

# Out:
- `H` `M × N` convolution matrix defined by `h`
- `y` vector of length `M` containing the discrete convolution
  of `h` and `x` computed using `H`.
"""
function convolution(h, x)
```

Email your solution as an attachment to `eecs551@autograder.eecs.umich.edu`.

The material between the triple quotation marks `"""` is a `Julia` **docstring** and is the preferred way to document code. It supports **markdown** syntax. With this approach, you can type `?convolution` at the REPL or in a Jupyter notebook and see nicely formatted documentation. You are not required to include documentation when submitting code to the autograder, but making clear documentation is important in practice.

Autograder instructions:
Throughout the semester, all coding problems will be graded via an automated system that verifies the correctness of your code by evaluating it on a suite of test cases. To submit your code, simply attach your file to an email (from your `@umich.edu` account) and send it to: `eecs551@autograder.eecs.umich.edu`. The system will send an email response within a few seconds saying whether your code passes or fails its suite of (typically randomly generated) test cases.
If your code fails: Edit and resubmit your code via the same process. You have unlimited retries.
It is much more intelligent to first write your own test cases rather than trying to use the autograder as a debugger!
Once your code passes: You are done! You will receive full credit for the problem. Do not submit anything to gradescope for this problem! The system already saved an electronic copy for our records. The "pass" email also contains a confirmation code. If we accidentally fail to give you credit for the problem, forward to us the confirmation code and we will record your credit. Points will be posted on Canvas at the end of the term.

Pr. 13.
To see how to use the convolution matrix from the previous problem in an image processing example, see:
`https://web.eecs.umich.edu/~fessler/course/551/julia/demo/hw1_show_conv_2d.html`
`https://web.eecs.umich.edu/~fessler/course/551/julia/demo/hw1_show_conv_2d.ipynb`
Modify that notebook to call your own `convolution` function and see how it works on the 2D image in that notebook.
To use this notebook, you will have to first install the `MIRTjim.jl` and `ImagePhantoms.jl` packages by typing

```
] add MIRTjim
```

```
] add ImagePhantoms
```

Submit a screen shot of your filtered 2D image (the results of 2D convolution) to gradescope.
Your screenshot must include a figure title that shows your name or username.
Caution
When you submit scans of your work to gradescope, be sure to follow the instructions to properly associate submitted work to problems and subproblems. *Only work that is submitted properly will be graded!* For a class this size it is infeasible to make exceptions. Start the uploading process early to ensure that you have time to submit properly.

Non-graded problem(s) below
(Solutions will be provided for self check; do not submit to gradescope.)

Pr. 14.
Evaluate the following **block matrix** operations by simplifying the result as much as possible. (For simplicity, you may assume $\boldsymbol{A}, \boldsymbol{B}, \ldots, \boldsymbol{H}$ below are all square matrices of the same size. Most of the equalities generalize to rectangular matrices of appropriate sizes.) Below, $\boldsymbol{I}_2$ denotes the 2×2 identity matrix.

(a) $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} & \boldsymbol{C} \end{bmatrix} \begin{bmatrix} \boldsymbol{D} \\ \boldsymbol{E} \\ \boldsymbol{F} \end{bmatrix} =$

(b) $\begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{B} \end{bmatrix} \begin{bmatrix} \boldsymbol{C} & \boldsymbol{D} \end{bmatrix} =$

(c) $\boldsymbol{A} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} & \boldsymbol{D} \end{bmatrix} =$

(d) $\begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{B} \\ \boldsymbol{C} \end{bmatrix} \boldsymbol{D} =$

(e) $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix} \begin{bmatrix} \boldsymbol{E} & \boldsymbol{F} \\ \boldsymbol{G} & \boldsymbol{H} \end{bmatrix} =$

(f) $(\boldsymbol{I}_2 \otimes \boldsymbol{A}) \begin{bmatrix} \boldsymbol{E} & \boldsymbol{F} \\ \boldsymbol{G} & \boldsymbol{H} \end{bmatrix} =$

(g) $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix} \begin{bmatrix} \boldsymbol{C} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D} \end{bmatrix} \begin{bmatrix} \boldsymbol{E} & \boldsymbol{F} \end{bmatrix}' =$

(h) $\operatorname{trace}\left(\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix} \right) =$

(i) $\det\left(\begin{bmatrix} \boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{B} & \boldsymbol{C} \end{bmatrix} \right) =$

(j) $\boldsymbol{A} \operatorname{Diag}(\lambda_1, \ldots, \lambda_N) =$

Even though this problem is not graded, keep in mind that we will use block matrix operations like these very frequently throughout the semester, and block matrices are likely to appear on exams and in your future work.

Pr. 15.
Show that if $\boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{M \times N}$, and if $\boldsymbol{A} \in \mathbb{F}^{N \times N}$ and $\boldsymbol{D} \in \mathbb{F}^{M \times M}$ are invertible, then

$$\det(\boldsymbol{D} + \boldsymbol{C}\boldsymbol{A}\boldsymbol{B}') = \det(\boldsymbol{A}^{-1} + \boldsymbol{B}'\boldsymbol{D}^{-1}\boldsymbol{C}) \det(\boldsymbol{A}) \det(\boldsymbol{D}).$$

You may use properties stated or shown in previous problems.
