Pr. 1.
The mythical Michigan Wolverine eats something once a day, either cheese, grapes, or lettuce. Its daily dietary habits follow the following rules:

- If it ate cheese today, tomorrow it will eat lettuce or grapes with equal probability.
- If it ate grapes today, tomorrow it will eat grapes with probability 1/10, cheese with probability 4/10, and lettuce with probability 5/10.
- If it ate lettuce today, it will not eat lettuce again tomorrow, but it will eat grapes with probability 4/10 or cheese with probability 6/10.

(a) What is the **long term average probability** of the Wolverine eating cheese, grapes, or lettuce on a particular day?

(b) Is your answer the unique soluion? Explain why or why not?

Pr. 2.
Show that when $\boldsymbol{A}$ has full column rank and $\boldsymbol{P}$ is symmetric positive definite:
$\text{eig}\big(\boldsymbol{P}^{1/2}\boldsymbol{A}'\boldsymbol{A}\boldsymbol{P}^{1/2}\big) < 2 \iff 2\boldsymbol{P}^{-1} \succ \boldsymbol{A}'\boldsymbol{A}$. This property is important for the **preconditioned gradient descent** algorithm for solving **least-squares** and related problems.

Pr. 3.
This problem constructs a simple and practical **diagonal majorizer** for use in iterative algorithms.
A $n \times n$ square matrix $\boldsymbol{A}$ is called **diagonally dominant** iff $|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$ for $i = 1, \ldots, n$.

(a) (Optional) Use the **Gershgorin disk theorem** to show that if $\boldsymbol{H}$ is a (Hermitian) symmetric matrix that is diagonally dominant with $h_{ii} \geq 0$ for all i, then $\boldsymbol{H} \succeq \boldsymbol{0}$.

(b) Show that if $\boldsymbol{B}$ is a $n \times n$ Hermitian symmetric matrix, then $\boldsymbol{D} \triangleq \text{diag}(|\boldsymbol{B}|\boldsymbol{1}_n) \succeq \boldsymbol{B}$ where $|\boldsymbol{B}|$ denotes the matrix consisting of the absolute values of the elements of $\boldsymbol{B}$, *i.e.*, `abs.(B)` and $\boldsymbol{1}_n$ is `ones(n)` in `Julia`.

Pr. 4.
Let $\boldsymbol{A}$ denote the $N \times N$ **Vandermonde** matrix corresponding to the roots of the polynomial $z^N = 1$.
Let $\boldsymbol{B}$ denote the $N \times N$ **Vandermonde** matrix corresponding to $\{\text{e}^{-\imath 2\pi(k+1/2)/N} \, : \, k = 0, \ldots, N-1\}$.
Define the matrix $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix}$.
Determine whether $\boldsymbol{X}$ is a **frame**, a **tight frame**, a **Parseval tight frame**, or none of the above.
If it is some type of frame, determine its **frame bound(s)**.

Pr. 5.
Let ψ denote the **Fair potential function**

$$\psi(t) \triangleq \delta^2 \left(\frac{|t|}{\delta} - \log\left(1 + \frac{|t|}{\delta}\right)\right),$$

with parameter $\delta > 0$.

(a) Show that

$$\psi'(t) = \frac{t}{1 + |t|/\delta}.$$

Hint: At first glance, it may look like ψ is not differentiable due to the absolute value, but try computing $\psi'(t)$ separately for $t \geq 0$ and $t < 0$.

(b) Show that the **Lipschitz constant** of ψ' is $\mathcal{L}_{\psi'} = 1$.

Hint: Recall that the Lipschitz constant $\mathcal{L}_f$ of a function $f : \mathbb{R} \to \mathbb{R}$ is the smallest constant such that $|f(z) - f(x)| \leq \mathcal{L}_f |z - x|$ for all x, z.

Hint: If f is differentiable with bounded derivative, then $\mathcal{L}_f = \sup_t |f'(t)|$.

Hint: Show that $\psi''(t) = \delta^2/(\delta + |t|)^2$. Again consider $t \geq 0$ and $t < 0$ separately.

(c) Determine the **proximal operator** for the (scaled) Fair potential on $\mathbb{R}^N$, *i.e.*, for $\boldsymbol{v} \in \mathbb{R}^N$ solve

$$\text{prox}_{\beta\psi}(\boldsymbol{v}) = \underset{\boldsymbol{x} \in \mathbb{R}^N}{\arg\min} \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{x}\|_2^2 + \beta \mathbf{1}' \psi.(\boldsymbol{x}).$$

Pr. 6.
Write a very short `Julia` function that computes the **nuclear norm** of a **circulant matrix** `C` without calling expensive functions like `eig` or `svd`.

Pr. 7.

(a) Give an example of a 4×4, rank-1 **transition matrix** $\boldsymbol{P}$ for which equilibrium distribution is $\boldsymbol{\pi} = \mathbf{1}_4/4$.

(b) Is your example the unique solution? Explain why or why not.

Pr. 8.
Consider the **regularized LS** cost function $f(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \delta^2 \frac{1}{2} \|\boldsymbol{x}\|_2^2$.

(a) (0 points) Determine the **gradient** of this cost function.

(b) Determine an easily computed upper bound on the **Lipschitz constant** of the gradient of this cost function. Your final expression must not involve any singular values, because the SVD is too expensive to compute for large-scale problems. Your bound should be one that is reasonably tight (see below); if your bound is too loose (too large), then using its reciprocal as the step size in gradient descent (or related algorithms) would lead to undesirably slow convergence.

(c) Determine (analytically) the numerical value of your upper bound when $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I}_5 \\ \mathbf{1}_5 \mathbf{1}_5' \end{bmatrix}$ and $\delta = 2$.

(d) Determine (analytically) the exact value of the Lipschitz constant of the gradient of the cost function for the $\boldsymbol{A}$ in the previous part. If your bound is more than 10% larger than the exact Lipschitz constant for this case, then you have not found a good enough bound and you should return to part (b) and improve your answer.

Pr. 9.
(Steepest descent for least squares problems)
The gradient descent method for LS problems requires selecting a step size parameter. In contrast, the **steepest descent** method determines the **step size** automatically (for quadratic problems), as discussed in the course notes and analyzed in a previous HW problem.

(a) Write a function called `lssd` that implements the steepest descent method for iteratively finding the minimizer over $\boldsymbol{x}$ of $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2$.

For full credit, your final version of the code should should use only one multiplication by $\boldsymbol{A}$ and one multiply by $\boldsymbol{A}'$ each iteration. (For debugging you might initially write a version that uses more multiplies.)

Your function should use `Julia`'s function keyword arguments capability so that the initial estimate $\boldsymbol{x}_0$ is $\mathbf{0}$ by default and so that the number of iterations is 10 by default.

In `Julia`, your file should be named `lssd.jl` and should contain the following function:

```
"""
    x = lssd(A, b ; x0=zeros(size(A,2)), nIters::Int=10)

Perform steepest descent to solve the least squares problem
`\\min_x \\|b - A x\\|_2`

In:
* `A` a `m x n` matrix
* `b` a vector of length `m`

Option:
* `x0` is the initial starting vector (of length `n`) to use; default `zeros`
* `nIters` number of iterations to perform; default `10`

Out:
* `x` a vector of length `n` containing the approximate solution

Notes:
Because this is a quadratic cost function, there is a closed-form solution
for the step size each iteration, so no "line search" procedure is needed.

A full-credit solution uses only *one* multiply by `A` and one by `A'` per iteration.
"""
function lssd(A, b ; x0=zeros(size(A,2)), nIters::Int=10)
```

Submit your solution to the autograder by emailing it as an attachment to `eecs551@autograder.eecs.umich.edu`.

(b) After your code passes, use it to generate a plot of $\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}\|$ as a function of k for $\boldsymbol{A}$ and $\boldsymbol{b}$ generated as follows.

```
using Random: seed!
seed!(0) # seed random number generator
m = 100; n = 50; sigma = 0.1
A = randn(m, n); xtrue = rand(n)
b = A * xtrue + sigma * randn(m)
```

Does $\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}\|$ **decrease monotonically** with k in the plots?

Optional: compare to GD on the same plot using your `lsgd` function. Does SD converge faster than GD here?

(c) Submit your code (a screen capture is fine) to gradescope so that the grader can verify that you used only one multiply by $\boldsymbol{A}$ and $\boldsymbol{A}'$ each iteration.

Hint: If $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{d}$ then $(\boldsymbol{A}\boldsymbol{x}_{k+1}) = (\boldsymbol{A}\boldsymbol{x}_k) + (\boldsymbol{A}\boldsymbol{d})$, so think about using a `+=` operation.

Optional problem(s) below
(not graded, but solutions will be provided for self check; do not submit to gradescope)

Pr. 10.
You showed in a previous HW problem that if $\boldsymbol{A}$ is a $M \times N$ matrix, then $\|\boldsymbol{A}\|_2 \leq \sqrt{N}\,\|\boldsymbol{A}\|_1$.

(a) Find a simple (but nonzero) 1×2 or 2×1 matrix where this inequality is equality. (When an inequality is achievable, we call it a **tight** bound.)

(b) This bound is not tight for the **normal** matrices. If $\boldsymbol{A}$ is normal and $N \times N$, then $\|\boldsymbol{A}\|_2 \leq c\,\|\boldsymbol{A}\|_1$ for a constant c that is much smaller than N in general. Determine (and prove) the best value for c.

(c) Give a nonzero 2×2 matrix where this inequality is equality, so it is also tight.

Hint. Think about **spectral radius**.