A. Images for web
    a. Vector images use lines, points, and polygons to represent an image
        i. essentially giant math equations, and each dot, line and shape is represented by its own equation
        ii. ideal for images that consist of geometric shapes
        iii. SVG is an abbreviation that stands for Scalable Vector Graphics
            1. It is the primary vector image format for the web
            2. XML-based markup language for describing two dimensional based vector graphics
            3. SVG is to graphics what HTML is to text
            4. defined in XML text files which can be searched, indexed, scripted and compressed
            5. can be created and edited with any text editor and with drawing software
            6. SVG editor: https://svg-edit.github.io/svgedit/releases/latest/editor/svg-editor.html
            7. SVG can be included in documents in several ways
                a. as img or background-image: if you don't need to modify with CSS
                b. as inline: if you want to access inner paths/shapes with CSS/scripting/filters
                c. as data URI in CSS or in document: if you don't need to modify the SVG programmatically and you want to save additional data file download
    b. Raster images represent an image by encoding the individual values of each pixel within a 2-dimensional grid of individual "pixels"
        i. 100x100 pixel image is a sequence of 10,000 pixels
        ii. each pixel stores the "RGBA" values: (R) red channel, (G) green channel, (B) blue channel, and (A) alpha (transparency) channel
        iii. complex scenes with lots of irregular shapes and details
        iv. Scaling up a raster image = jagged and blurry graphics
B. Zoom and resolution
    a. https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/image-optimization
    b. Different images are appropriate for different resolution screens, especially as we move towards higher resolution screens for our desktop monitors and our mobile devices
    c. High resolution screens have multiple device pixels per CSS pixel
        i. Remember CSS pixels are "magic" units and represent a ratio that maps to an actual device pixel, not the actual device pixel itself
        ii. Single CSS pixel may contain multiple device pixels
        iii. The more device pixels there are, the finer the detail of the displayed content on the screen
    d. High resolution images require significantly higher number of pixels and bytes
        i. High DPI screens (dots per inch = pixels fit into an inch)
        ii. image assets need more detail in order to use higher device pixel counts
            1. vector images = rendered at any resolution with sharp results
            2. raster images = data on a per-pixel basis
            3. larger the number of pixels = the larger the filesize of a raster image
            4. double the resolution of the physical screen = quadruples the number of required pixels
        iii. high resolution screens also require high-resolution images

1. prefer vector images whenever possible
    a. Scaling up a vector = smooth, crisp graphics
    b. ideal format for high-resolution screens and assets that need to be displayed at varying sizes
2. deliver and optimize multiple variants of each image (will talk about responsive images with responsive layouts)

C. Lossless vs lossy image compression
    a. For certain data like source code or executable, it is critical that a compressor does not alter or lose any of the original information
        i. A single missing or wrong bit of data could completely break it entirely
        ii. “Lossless” compression = capture all of the data of your original file and nothing from the original file is lost
            1. file may still be compressed
            2. all lossless formats will be able to reconstruct file to its original state
    b. For some other types of data, such as images, audio, and video, it may be perfectly acceptable to deliver an "approximate" representation of the original data
        i. We can often get away with discarding some information about each pixel in order to reduce the filesize of an image without the human ear or eye being able to detect the difference
        ii. “Lossy” compression = approximate what your original image looks like
        iii. It might reduce the amount of colors in your image or analyze the image for any unnecessary data
        iv. It typically reduces the file size, though it may reduce the quality of your image as well
        v. Lossy files typically much smaller than lossless files
    c. Each raster image file is either lossless or lossy, depending on how the format handles your image data
        i. any image can undergo a lossy compression step to reduce its size
        ii. The difference between various image formats GIF, PNG, JPEG, and others is the combination of the specific algorithms they use or not when applying the lossy and lossless steps
    d. There is no optimal configuration/universal setting
        i. It all depends on the image and what is acceptable to the display

D. GIF is an abbreviation that stands for Graphics Interchange Format
    a. lossless raster format
    b. Limited to 256 colors because it only supports up to 8 bits per pixel for each image
    c. GIF supports transparency
        i. What we call aliased transparency not alpha channel transparency (like the alpha from color in CSS where you can have a degree of transparency)
        ii. GIFs support full opacity or full transparency for each pixel only
        iii. “Jaggy edge”, transparency does not blend nicely

E. PNG is an abbreviation that stands for Portable Network Graphics
    a. lossless raster format
    b. Sometimes referred to the “next-generation GIF”
    c. Supports anti-aliased transparency (ie, alpha channel with some degree of transparency)
    d. “Smooth edge”, blends nicely
    e. PNGs are capable of displaying higher color depths
        i. 8 bit PNG - 256 colors, just like GIFs, produces slightly smaller files
        ii. 24 bit PNG - 16.8million colors plus alpha transparency

f. PNGS are quickly becoming one of the most common image formats used online

F. JPG is an abbreviation that stands for Joint Photographic Experts Group
   a. lossy raster format
   b. JPGs give you a sliding scale of compression that enables decreasing file size tremendously, but increases artifacts or pixelation the more the image is compressed

G. Optimizing images
   a. Do you need an image at all?
      i. Eliminate unnecessary image resources
      ii. Every image downloaded means an extra data load on your user
   b. Select the right image format
      i. three universally supported image formats: GIF, PNG, and JPEG
      ii. Need small file sizes for simple graphics?
         1. Use GIF
         2. compression techniques in the GIF format allow image files to shrink tremendously
      iii. need animated image?
         1. GIF is the only universal choice
      iv. Need transparency?
         1. PNG = alpha transparency
         2. GIF = aliased transparency
      v. need to preserve fine detail with highest resolution?
         1. Use PNG
         2. does not apply any lossy compression algorithms beyond size of the color palette
         3. highest quality image, but at a cost of significantly higher filesize than other formats -- use wisely!
      vi. Image is a photo, screenshot, or similar?
         1. Use JPEG
         2. JPEG uses a combination of lossy and lossless optimization to reduce file size of the image asset
         3. Try different JPEG quality levels to find the best quality vs. filesize tradeoff for your asset
      vii. Image contains imagery composed of geometric shapes?
         1. consider converting it to SVG format
      viii. Image contains text?
         1. Use web fonts instead of encoding text in images
         2. Still selectable, searchable, resizeable, easily translatable -- usability!
   c. SVG files should be minified/optimized to reduce their size
      i. https://jakearchibald.github.io/svgomg/
      ii. Illustrator SVG → optimized
   d. Compress raster images
      i. Bit depth
         1. number of memory bits used to store color data for each pixel in a raster image
         2. 1 bit = 2 colors (black and white) "bit map"
         3. 2 bits = 4 colors
         4. 4 bits = 16 colors
         5. 8 bits = 256 colors
         6. 16 bits = "thousands"
         7. 32 bits = "millions"

ii. reduce the "bit-depth" of the image from 8 bits per channel to a smaller color palette
iii. 8 bits per channel = 256 values per channel = 16,777,216 (256 ^ 3) colors in total
iv. Reducing to 256 colors = 8 bits in total for the RGB channels and immediately save two bytes per pixel
v. Complex scenes with gradual color transitions (gradients, sky, etc.) require larger color palettes to avoid visual artifacts
vi. if the image only uses a few colors, then a large palette is simply inflating bits for no reason

e. Leverage CSS effects (shadows, gradients, etc.) where possible
i. Already being loaded along with your css
ii. CSS effects/animations produce resolution-independent assets
iii. always look sharp at every resolution and zoom level
iv. Need a fraction of the bytes required by an image file

H. CSS Background images
a. Image files
i. applies an image or gradient to the background of an element
ii. Eg. `background-image: url(foo.jpg);`
iii. url() -- file path to any image to use for background
1. Just like other images, you can use an image path relative to the folder from which css is loaded, or an absolute image path starting with http
iv. We can use multiple background images:
v. Eg. `background-image: url(image-front.jpg), url(image-back.jpg);`
1. Comma separated values, the first image listed is the one that appears to the front and then layering down the stack from there
vi. Background images will tile horizontally and vertically by default
1. CSS gives us background-repeat attribute to control
a. No-repeat
b. Repeat-x
c. Repeat-y

b. Gradients
i. Linear gradients -- straight line
1. Eg. `background-image: linear-gradient(red, green);`
2. Gradients are drawn top to bottom by default
3. "To" keyword to change direction
a. Eg. `background-image: linear-gradient(to bottom right, red, green);`
4. Degrees can be added to specify rotation in a clockwise direction
a. Eg. background-image: linear-gradient(135deg, red, green);
b. 0deg = vertical gradient running bottom to top
c. 90deg = horizontal gradient running left to right
d. 180deg = vertical gradient running top to bottom
e. 270deg = horizontal gradient running right to left
f. negative angles run in the counterclockwise direction
5. Color stop: change from one color to another at specific spot
a. Eg. `background-image: linear-gradient(red, green 25%);`
b. two color stops that are the same = color instantly changes to another solid color

i. Eg. `background-image: linear-gradient(red, red 25%, green 25%);`

ii. Radial gradients -- circle out from point
1. Eg. `background-image: radial-gradient(red, green);`
2. Default shape is ellipse if box is not square
   a. Force to circle: Eg. `background-image: radial-gradient(circle, red, green);`
3. "At" keyword to change location of center
   a. Eg. `background-image: radial-gradient(at top right, red, green);`
4. Supports linear concepts like color stops, multiple colors

I. CSS Effects
   a. Box-shadow: Used in casting shadows (drop shadows)
      i. Eg: `box-shadow: 5px 5px 10px 10px red;`
      ii. 5 parameters
         1. The horizontal offset (required)
            a. positive = the shadow will be on the right of the box
            b. negative = the shadow on the left of the box
         2. The vertical offset (required)
            a. negative = the shadow will be above the box
            b. positive = the shadow will be below the box
         3. The blur radius (required)
            a. At 0, the shadow will be sharp with no blur
            b. the higher the number, the more blurred it will be, and the further out the shadow will extend
         4. The spread radius (optional)
            a. positive spread increases the size of the shadow
            b. Negative spread decreases the size of the shadow
            c. Default is 0 (the shadow is same size as its blur)
         5. Color (optional)
            a. If color is omitted, shadow will use the foreground text color
            b. Can take any named value, hex, rgb, rgba
      iii. Shadow can be inset into the box as well by using "inset" keyword
         1. Eg. `box-shadow: inset 5px 5px 10px 10px red;`
      iv. You can apply multiple shadows around the box by using comma separated box shadows

J. CSS Animations
   a. CSS gives us the ability to animate transitions from one CSS style configuration to another
   b. Animations consist of two components
      i. A style describing the CSS animation
      ii. A set of keyframes that indicate the start and end states of the animation's style + possible intermediate points
   c. Advantages to CSS animations
      i. easy to use for simple animations (does not require knowing JS, Canvas, Flash, etc.)
      ii. the animations run well because the browser can optimize performance and efficiency to keep the performance as smooth as possible
   d. Animation example
   e. Animation sub properties

i. Animation-name: the name of the @keyframes describing the animation's keyframes
ii. Animation-duration: length of time that an animation should take to complete one cycle
iii. Animation-timing-function: how the animation transitions through keyframes, by establishing acceleration curves (linear, ease-in, ease out, etc.)
    1. https://developer.mozilla.org/en-US/docs/Web/CSS/animation-timing-function
iv. Animation-delay: delay between the time the element is loaded and the beginning of the animation sequence
v. Animation-iteration-count: number of times the animation should repeat
    1. Use "infinite" to repeat forever
vi. Animation-direction: should the animation alternate direction on each run through the sequence or reset to the start point and repeat itself
vii. Animation-fill-mode: what values are applied by the animation before and after it is executing
    1. Does it retain the last css styles to persist the change? Does it reset itself?
    2. https://developer.mozilla.org/en-US/docs/Web/CSS/animation-fill-mode
viii. Animation-play-state: pause and resume the animation sequence