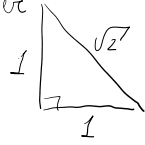


Module 2

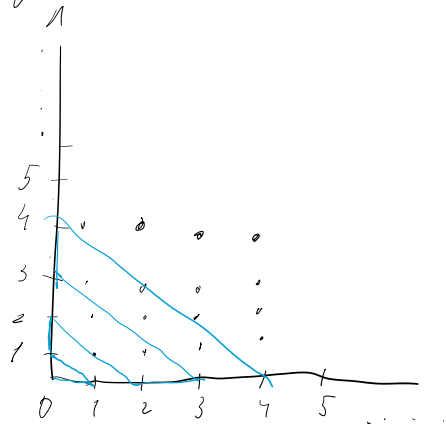
Monday, September 11, 2023 3:41 PM

- Prove that $\sqrt{2}$ is actually a number

For a \triangle with sides as 1 will have
a hypotenuse that's $\sqrt{2}$



- Algebraic real numbers: numbers that are the solutions to a polynomial
- Counting and Comparing Infinities
 - Cardinality: if there is $1-1$ mapping between 2 sets, they are of the same size - have the same cardinality
 - The sets of all rational numbers and integers are equal:



- If an infinite set of numbers can be lined up with natural numbers then this set is a countable infinity
 - Irrational and Real number sets are uncountable

In-Class Exercise 7: Show that the cardinality of $[0, 1]$ is the same as that of any $[a, b]$, for example $[0, 10]$.

For could to be the same there needs to be a $1-1$ mapping, so $f(x)$ - the mapping function should have a single solution for any input from $[0, 1]$ that's mapped into $[a, b]$.

Let $f(x) = a + (b-a) \cdot x$ - linear func that maps the set $[0, 1]$ to $[a, b]$

Let $f(x) = a + (b-a) \cdot x$ - linear func that maps the set $[0,1]$ to $[a,b]$

$\Rightarrow f(x_1) = f(x_2)$ should only be the case when $x_1 = x_2$

$$\Rightarrow a + (b-a) \cdot x_1 = a + (b-a) \cdot x_2$$

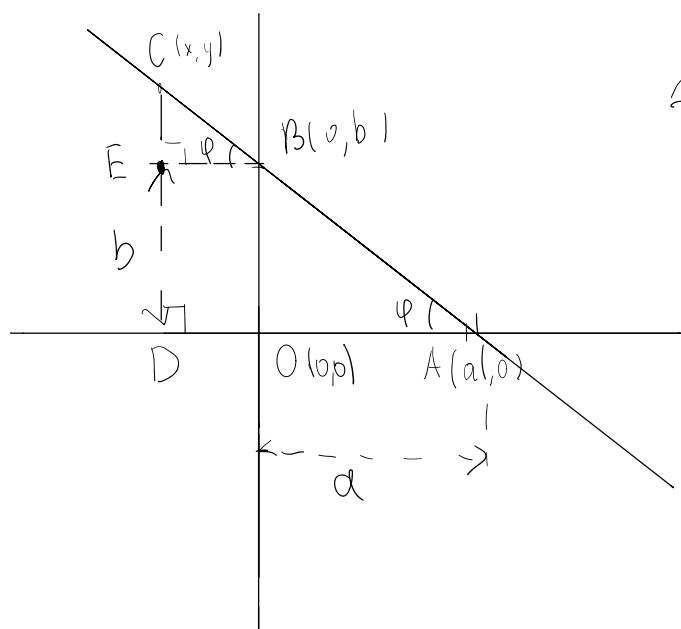
$$\Rightarrow \underline{(b-a) \cdot x_1} = \underline{(b-a) \cdot x_2}$$

$\Rightarrow x_1 = x_2$, so card of any $[0,1]$

where $A_i \in \mathbb{R}$ is = card of $[a, b]$ where $B_i \in \mathbb{R}$

In-Class Exercise 9: Why is $ax + by + c = 0$ the equation of a line? Can you find an argument using simple geometry (similar triangles)?

A line can be represented by the location of its initial (a, b) and terminal points (x, y) on a Cartesian coordinate system.



$\triangle CEB$ and $\triangle CDA$ are similar

$$\tan \varphi = \frac{CE}{EB} = \frac{CD}{DA} = \frac{CD - ED}{DA - OA} = \frac{y - b}{x} = m$$

$$m = \frac{y - b}{x}$$

$$mx = y - b$$

$$\boxed{y = mx + b} \text{ Proven!}$$

Now, let $y = mx + b$

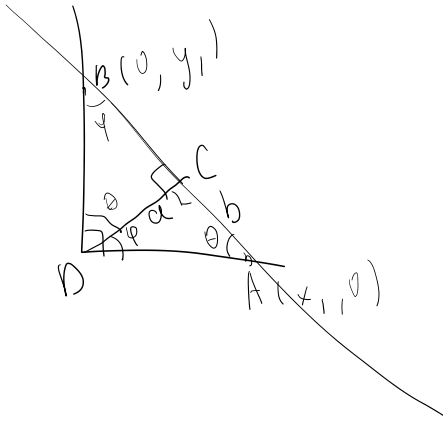
$$\Rightarrow by = bmx + b^2$$

$$\Rightarrow bmx - by + b^2 = 0$$

$$\Rightarrow bm = a \quad -b = b \quad c = b^2$$

$$\Rightarrow b m = a, -b = b, c = b^2$$

Proof 2.



$$\tan \theta = \frac{CD}{CA} = \frac{BD}{DA}$$

due to similarity of the Δ s

$$\Rightarrow \frac{a}{b} = \frac{y}{x}$$

$$\Rightarrow ax = by$$

$$\Rightarrow ax - by = 0$$

$$\Rightarrow a = a, b = b, c = 0 \quad \text{both eq. are similar}$$

Polynomials

- Revise Combinatorics?
- Bernstein Polynomial

Trig functions

- Any well-behaved function can be represented using the linear combination of \sin and \cos

$$f(t) = a_0 + \sum_{k=1}^{\infty} a_k \sin(2\pi kt/T) + \sum_{k=1}^{\infty} b_k \cos(2\pi kt/T)$$

$$e^z \triangleq 1 + z + \frac{z^2}{2!} + \frac{z^3}{3!} + \dots \quad (\text{analogous to the Taylor series for the real function } e^x)$$

$$\begin{aligned} e^{i\theta} &= 1 + i\theta + \frac{(i\theta)^2}{2!} + \frac{(i\theta)^3}{3!} + \dots \\ &= (\text{series for } \cos(\theta)) + i(\text{series for } \sin(\theta)) \\ &= \cos(\theta) + i\sin(\theta) \end{aligned}$$

- Proving Number Theory using Complex Numbers

- The cardinality of the plane is the same as the line.

In-Class Exercise 8: Why is this true?

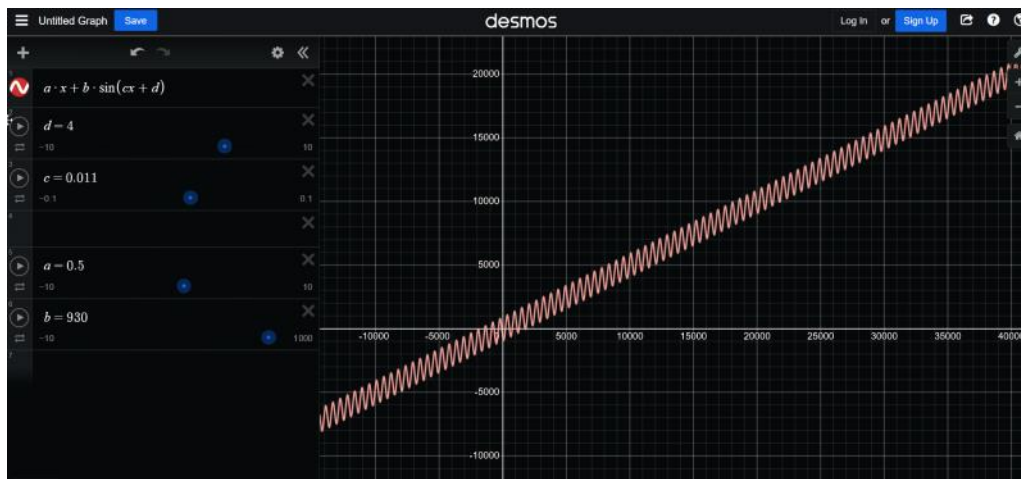
For the Cardinality of 2 sets to be the same there needs to exist a bijection. If $A = \{1, 2, 3, \dots\}$ and $B = \{1, 2, 3, \dots\}$ then $A \sim B$

For the Cardinality of 2 sets to be the same there needs to exist a function that'll perform 1-1 unique mapping from set A to set B

If we can prove the existence of such a function, then we can prove that the cardinalities of set A and B are the same. Let's take a line (called X) which will inevitably have all of its points in \mathbb{R} and a plane (XY) with all its points in the set $\mathbb{R} \times \mathbb{R}$. For example, we can create a mapping with the following function:

$$f(x) = a_0 x + a_1 \sin(a_2 x + a_3)$$

In this case, we have a continuous function that'll map every point in X to a different point in XY. Similarly, every point in XY will map to a unique point in X. Since we have established that this function f is a bijection, the cardinalities of these sets are the same



In-Class Exercise 10: Consider the two-variable function $f(x, y) = (x + y)^n$. Show that $(x + y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k}$

We now that $(x + y)^n$ is n factors of $(x+y)$. In this representation, we know that the unique terms in the equation will be $\sum_{i=0}^n \{x^i y^{n-i}\}$. Now, the problem is to find how many times each of these terms will occur.

We know that $(x+y)^n = (x+y)(x+y)\dots(x+y)$ n times

$$\Rightarrow (x+y)^n = f(x, y) = \sum_{k=0}^n x^k y^{n-k}$$

\Rightarrow Number of ways we can choose k such that k satisfies $\sum_{k=0}^n x^k y^{n-k}$; $\binom{n}{k}$

\Rightarrow Sum of all terms will be $\sum_{k=0}^n \binom{n}{k} x^k y^{n-k}$

Unique terms in $(x+y)^n$

Binom. coef provides the num of terms for each value of k from n terms

Sum of all copies

sum of all terms will be

$$\sum_{k=0}^n \binom{n}{k} x^k y^{n-k}$$

sum of all copies
of all unique terms

- Replace y with $1 - x$ and define

$$b_{n,k}(x) = \binom{n}{k} x^k (1-x)^{n-k}$$

In-Class Exercise 11: Why is $b_{n,k}(x)$ a polynomial?

Polynomial is an equation of the following form: $f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x_1 + a_0$

$$b_{n,k}(x) = \binom{n}{k} x^k (1-x)^{n-k}$$

$\Rightarrow \binom{n}{k}$ is a constant

Binomial coefficient

$\Rightarrow x^k$ is a polynomial

$\Rightarrow (1-x)^{n-k}$ is a polynomial

$\Rightarrow x^k (1-x)^{n-k}$ is a polynomial

Product of Polynomials is a polynomial.

$\Rightarrow \binom{n}{k} \cdot x^k \cdot (1-x)^{n-k}$ is a polynomial

Product of a constant and a polynomial is a polynomial

In-Class Exercise 12: Let's explore the last point.

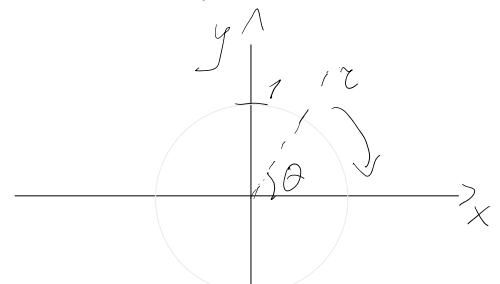
- Note: 2π is approximately 6.29. Why is it OK to define a value for $\sin(17.5)$ or $\sin(-156.32)$? How are these values defined?
- Download [PlotSin.java](#), [Function.java](#), and [SimplePlotPanel.java](#). Then compile and execute `PlotSin.java` to plot the function `sin` and `cos` in the range $[0, 20]$. How many times does the function "repeat" in this range?

First of all, $\sin(\theta)$ is a periodic function and one period is equal to 2π . This means that every ~ 6.29 radians, the values will repeat themselves. This means that any value of θ can be represented as multiples of 2π and the remainder.

Let θ be the angle between the radius of an arbitrarily sized circle on a Cartesian coordinate system and the positive direction of the X axis (or the Y axis value of the coordinate)

$$\Rightarrow \sin(\theta) = \sin(\theta + k \cdot 2\pi)$$

Due to periodicity



$$\Rightarrow \sin(\theta) = \sin(\theta + k \cdot 2\pi)$$

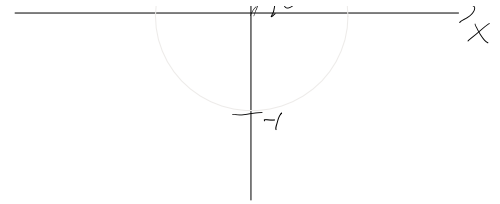
Due to periodicity

$$\Rightarrow \sin(-\theta) = -\sin(\theta)$$

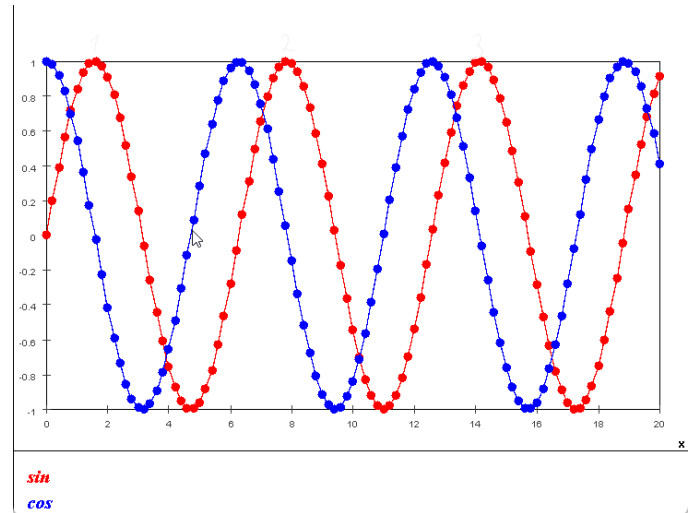
Due to the chosen direction

$$\Rightarrow -\infty < \theta < \infty$$

Any value for θ is acceptable
as it can be decomposed into the fixed
domain of $[0, 2\pi]$ using above rules



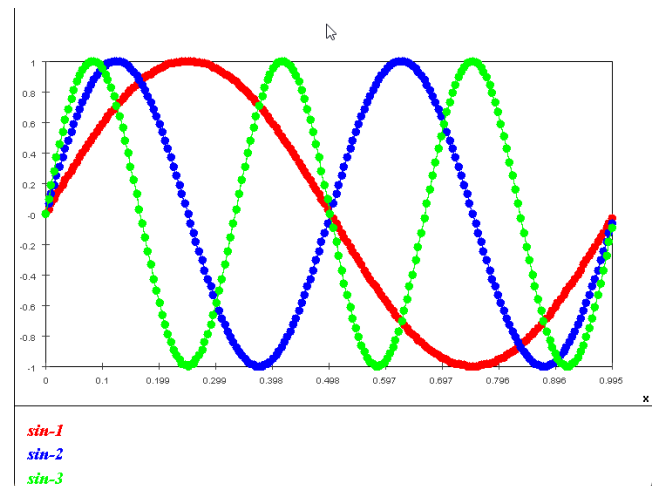
We can see that within the given range the function repeats itself 3 times



- Let's examine the function $\sin(2\pi\omega t)$.

In-Class Exercise 13: Compile and execute [PlotSin2.java](#) to plot $\sin(2\pi\omega t)$ in the range $[0, 1]$ with $\omega = 1, 2, 3$. What is the relationship between the three curves?

Despite all of the curves representing a sinusoid, their frequencies differ a lot. For example, the curve $\sin-3$ repeats itself 3 times as frequently as the $\sin-1$ curve. This is the concept of wavelength (or the period) and denotes a shorter wavelength with increased frequency



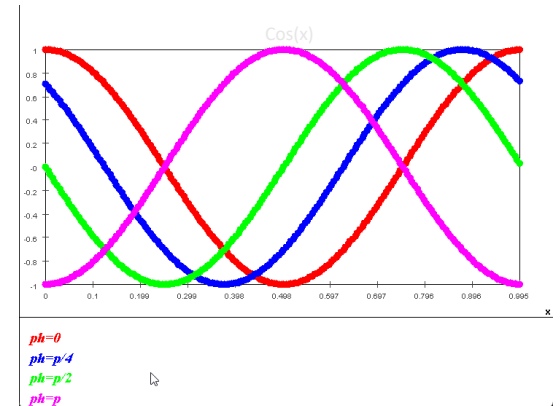
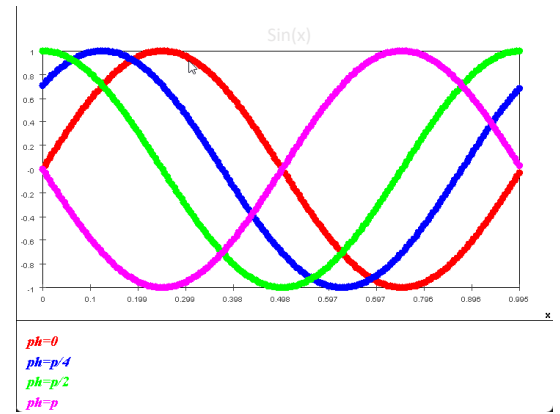
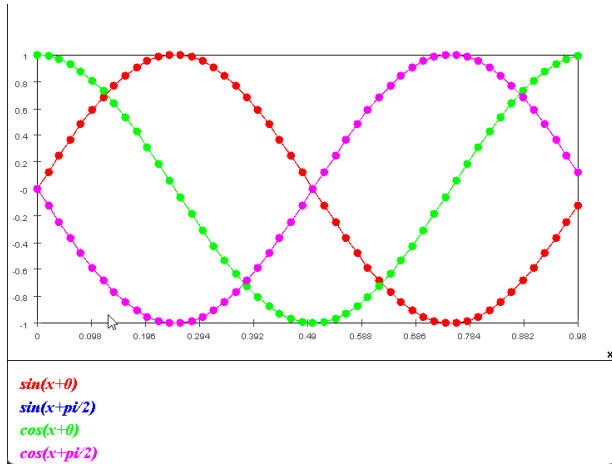
- You can shift a sin wave by adding a fixed constant to the argument: $\sin(2\pi\omega t + \phi)$

In-Class Exercise 14: Modify [PlotSin3.java](#) to plot $\sin(2\pi t + \phi)$ in the range $[0, 1]$. Plot four separate curves with $\phi = 0, \frac{\pi}{4}, \frac{\pi}{2}, \pi$. What can you say about $\sin(2\pi t + \frac{\pi}{2})$? Similarly, plot $\cos(2\pi t + \frac{\pi}{2})$ and comment on the function.

Here we can see the effects of phase shift on the $\sin(x)$ function. When it comes to the curve shifted by $\pi/2$ it is equivalent to \cos of the same x . This is more evident if we look at the example of a circle from before.

$$\cos(x) = \sin(x + \pi/2)$$

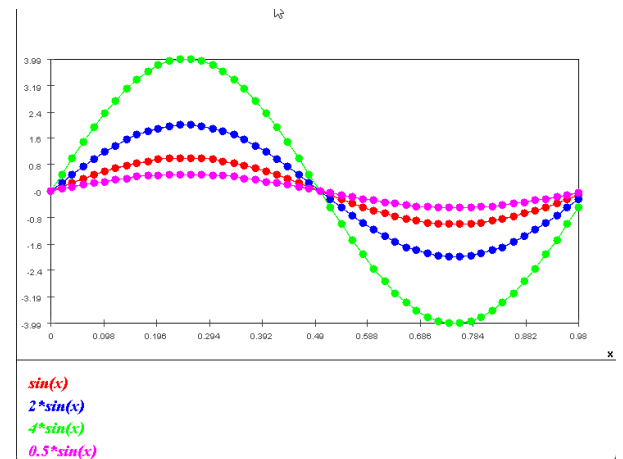
We can confirm this by checking out the cosine curves of the same phase shift values. In the below plot, we cannot see the $\sin(x+\pi/2)$ as it is overlapping with $\cos(x)$



- First, note that multiplying a sinusoid by a constant, e.g., $\alpha \sin(t)$ does not change the period.

In-Class Exercise 15: Why?

If we pay attention to the form of the function, it is evident the constant alpha will only stretch the result of the function $\sin(t)$. This will only affect the final result and will have no influence on the frequency or phase as we are not modifying the angle. In the illustration to the right, we can see that no matter what the constant (the amplitude) we choose is, the frequency will stay the same and there will be no phase shift. Basically the rate at which the function repeats itself remains unchanged



Complex numbers

- Once arithmetic has been defined, we can define *functions* on complex numbers
 ▷ e.g., for a complex number z , define $f(z) = z^2$.
- What about $\sin(z)$?

In-Class Exercise 16: Does it make sense to use a right-angled triangle to define $\sin(z)$ for a complex number z ?

Complex numbers have two components: a real part and an imaginary part. It is true that complex numbers can also be represented geometrically on the complex plane, with the real part on the one axis and the imaginary part on a second axis that is perpendicular to the first. However, this geometric representation extends beyond the simple geometric concept of length and angle present in right-angled triangles. Geometry in general deals with real lengths and angles since it was created on the basis of physical analogy to the shapes present around us in the real and observable world. However, geometry is not the only way we can use to represent this function. We can also use Euler's formula.

In-Class Exercise 17: As a little practice exercise with one dimensional arrays, write some code to determine whether a string is a *double-palindrome*. Download [ArrayExamples.java](#), which explains what a double-palindrome is, and write your code in [Palindrome.java](#).

The code on the right is my approach to solving the problem. Here the conditions are basically checked one-by-one. I have implemented the looping logic across just the 1st half of the array since we can easily obtain the indices for both the 2nd half and the entire array easily through simple arithmetic. The results are displayed at the bottom as well. I modified the `ArrayExamples.java` file so that the results are more self-explanatory

```

1 public class Palindrome {
2
3     public static boolean isDoublePalindrome (char[] digits)
4     {
5         // A double palindrome is defined as follows:
6         // (1) The string has even length.
7         // (2) The string is a palindrome.
8         // (3) Each half of the string is also a palindrome.
9
10        // check if the string has even length
11        if (digits.length % 2 != 0) {
12            return false;
13        }
14
15        int lenArr = digits.length;
16        int halfLenArr = digits.length / 2;
17
18        for (int i = 0; i < halfLenArr; i++) {
19            //We do not need to traverse the whole array, we can just traverse half of it
20            // and mirror the indices to check from the other end of the array
21            // check if the string is a palindrome as a whole
22            if (digits[i] != digits[lenArr - 1 - i]) {
23                return false;
24            }
25
26            // check if any of the halves violate the palindrome rule
27            if (digits[i] != digits[halfLenArr - 1 - i] || digits[halfLenArr + i] != digits[lenArr - 1 - i]) {
28                return false;
29            }
30        }
31
32        // if none of the above conditions are violated, then the string is a double palindrome
33        return true;
34    }
35 }

```

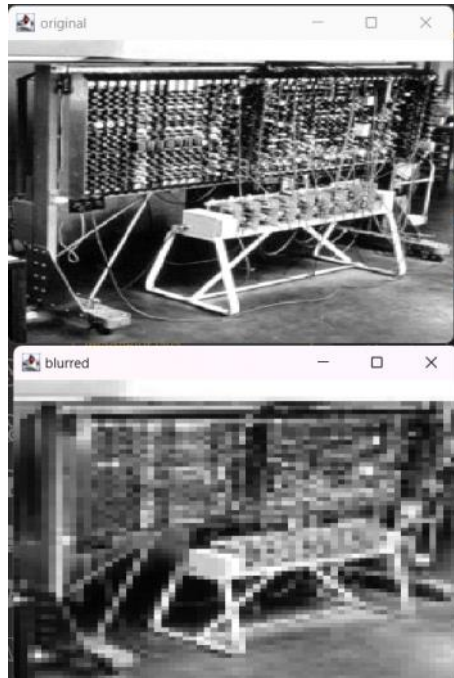
```

(base) C:\Users\togru\ADA\cla\lin\alg\mod2>java ArrayExamples
>> Passed test: 3333
[INFO] isDoublePalindrome(3333) returned true
>> Passed test: 24422442
[INFO] isDoublePalindrome(24422442) returned true
>> Passed test: 123321123321
[INFO] isDoublePalindrome(123321123321) returned true
>> Passed test: 1232112321
[INFO] isDoublePalindrome(1232112321) returned true
>> Passed test: 123242321
[INFO] isDoublePalindrome(123242321) returned false
>> Passed test: 3344
[INFO] isDoublePalindrome(3344) returned false
>> Passed test: 121212
[INFO] isDoublePalindrome(121212) returned false
>> Passed test: 12344321
[INFO] isDoublePalindrome(12344321) returned false
>> Passed test: 12211212
[INFO] isDoublePalindrome(12211212) returned false

```

In-Class Exercise 18: Write code to *blur* an image. Start by writing pseudocode to solve the problem. Then, download [ImageBlur.java](#), which contains instructions, and where you will write your code. You will also need [ImageTool.java](#). And as a test image, you can use [ace.jpg](#).

In the blur method, the input image is traversed in blocks of a specified size, blurSize, with non-square dimensions being carefully handled by checking the boundaries of the image. The average pixel intensity for each block is calculated, and this average value is assigned to all pixels within the block, effectively blurring the image. Finally, the newly generated blurred image, accommodating any non-square dimensions, is returned.



```

14 public class ImageBlur {
15     Run | Debug
16     public static void main(String[] argv) {
17         ImageTool imTool = new ImageTool();
18         int[][] pixels = imTool.imageFileToGreyPixels(fileName:"ace.jpg");
19         imTool.showImage(pixels, title:"original");
20         // Each block of k x k pixels has the same color.
21         int k = 4;
22         int[][] blurredPixels = blur(pixels, k);
23         imTool.showImage(blurredPixels, title:"blurred");
24     }
25
26     static int[][] blur(int[][] pixels, int blurSize) {
27         // Get the dimensions of the image
28         int height = pixels.length;
29         int width = pixels[0].length;
30
31         // Initialize a copy of the original array with the same dimensions
32         int[][] blurredPixels = new int[height][width];
33
34         // Iterate through the image in blocks of size blurSize x blurSize
35         for (int i = 0; i < height; i += blurSize) {
36             for (int j = 0; j < width; j += blurSize) {
37                 int sum = 0;
38                 int count = 0;
39
40                 // Calculate the average intensity in the current block
41                 for (int ki = 0; ki < blurSize && i + ki < height; ki++) {
42                     for (int kj = 0; kj < blurSize && j + kj < width; kj++) {
43                         sum += pixels[i + ki][j + kj];
44                         count++;
45                     }
46                 }
47                 int avg = sum / count;
48
49                 // Set the intensity of each pixel in the current block to the average
50                 for (int ki = 0; ki < blurSize && i + ki < height; ki++) {
51                     for (int kj = 0; kj < blurSize && j + kj < width; kj++) {
52                         blurredPixels[i + ki][j + kj] = avg;
53                     }
54                 }
55             }
56         }
57
58         // Return the blurred image
59         return blurredPixels;
60     }
61 }

```

In-Class Exercise 19: Let's examine the idea of computing the "distance" between two arrays.

- A distance measure should take two arrays and return a non-negative number.
- Propose a "distance" calculation for 1D arrays.
- Propose an extension of that idea to 2D arrays.

Do the arrays have to be of the same length?

To do this, we can assign a geometric meaning to arrays. This way, we can calculate the distance between the two arrays as the sum of the distances between each corresponding point. This can be done by aligning the elements of the arrays on a 1D (line) axis. To prevent the negative numbers from distorting the actual concept of distance we can use either the Manhattan or Euclidian distance measure.

$$A = [a_1, a_2, a_3, \dots]_n \quad B = [b_1, b_2, b_3, \dots]_m$$

$$\text{Euclidian: } D_E(A, B) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

$$\text{Manhattan: } D_M(A, B) = \sum_{i=1}^n |a_i - b_i|$$

We can extend this to 2D simply by treating each column as a 1D vector and summing the distances between those vectors. $A, B \in \mathbb{R}^{m \times n}$

$$D_E(A, B) = \sum_{i=1}^m \sqrt{\sum_{j=1}^n (a_{ij} - b_{ij})^2}$$

Ideally, to compute the Euclidean distance, the arrays should be of the same length/dimensions, as the formula involves pairwise differences between corresponding elements of the arrays. However, in practice, there might be cases where arrays of different lengths need to be compared. In such cases, we can use padding. Shorter array can be padded with zeros to match the length/dimension of the longer array.

In-Class Exercise 20: How is a 2D array stored in memory? Consider this code snippet:

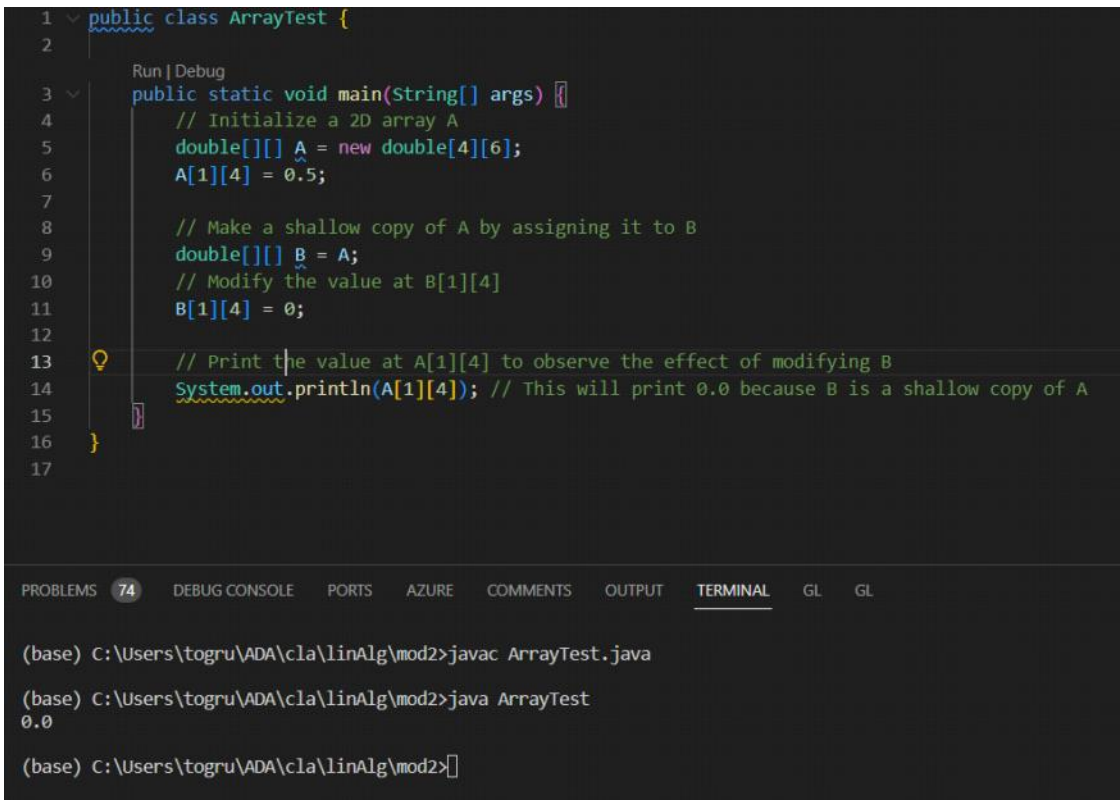
```
double[][] A = new double [4][6];
A[1][4] = 0.5;
double[][] B = A;
B[1][4] = 0;
System.out.println (A[1][4]);    // What gets printed out?
```

What does it mean to make a "deep copy" of an array?

In computer memory, a 2D array is stored as an array of arrays, often referred to as a "jagged array" because it allows each row to have a different number of columns, though in languages like Java, 2D arrays are typically rectangular.

When you declare a 2D array `double[][] A = new double[4][6];`, it's somewhat like creating 4 separate arrays (each of length 6) and then another array holding references to these 4 arrays. And B is not a deep copy of A. When the statement `double[][] B = A;` is executed, B becomes a reference to the same object in memory that A is pointing to which means it is a shallow copy

Making a "**deep copy**" of an array means creating a new array object and then copying each element from the original array to the new one



```
1 public class ArrayTest {
2
3     Run | Debug
4     public static void main(String[] args) {
5         // Initialize a 2D array A
6         double[][] A = new double[4][6];
7         A[1][4] = 0.5;
8
9         // Make a shallow copy of A by assigning it to B
10        double[][] B = A;
11        // Modify the value at B[1][4]
12        B[1][4] = 0;
13
14        // Print the value at A[1][4] to observe the effect of modifying B
15        System.out.println(A[1][4]); // This will print 0.0 because B is a shallow copy of A
16    }
17 }
```

PROBLEMS 74 DEBUG CONSOLE PORTS AZURE COMMENTS OUTPUT TERMINAL GL GL

```
(base) C:\Users\togru\ADA\cla\linAlg\mod2>javac ArrayTest.java
(base) C:\Users\togru\ADA\cla\linAlg\mod2>java ArrayTest
0.0
(base) C:\Users\togru\ADA\cla\linAlg\mod2>
```

