**Last name** ______________________

**First name** ______________________

## LARSON—MATH 310–CLASSROOM WORKSHEET 05
## Vectors!

### Chapter 2 of Klein's *Coding the Matrix* text

**Definition**. For a field $\mathbb{F}$ and a positive integer $n$, a vector with $n$ entries, each belonging to $\mathbb{F}$, is called an *$n$-vector* over $\mathbb{F}$. The set of $n$-vectors over $\mathbb{F}$ is $\mathbb{F}^n$.

**Definition**: A vector with four entries, each of which is a real number, is called a *4-vector* over $\mathbb{R}$.

1. What are examples of vectors?

2. How does our author view vectors as *functions*?

> **Definition 2.2.2:** For a finite set $D$ and a field $\mathbb{F}$, a *$D$-vector over* $\mathbb{F}$ is a function from $D$ to $\mathbb{F}$.
>
> This is a computer scientist's definition; it lends itself to representation in a data structure. It differs in two important ways from a mathematician's definition.
>
> - I require the domain $D$ to be finite. This has important mathematical consequences: we will state theorems that would not be true if $D$ were allowed to be infinite. There are important mathematical questions that are best modeled using functions with infinite domains, and you will encounter them if you continue in mathematics.
> - The traditional, abstract approach to linear algebra does not directly define vectors at all. Just as a field is defined as a set of values with some operations (+, -, *, /) that satisfy certain algebraic laws, a vector space is defined as a set with some operations that satisfy certain algebraic laws; then vectors are the things in that set. This approach is more general but it is more abstract, hence harder for some people to grasp. If you continue in mathematics, you will become very familiar with the abstract approach.
>
> Returning to the more concrete approach we take in this book, according to the notation from Section 0.3.3, we use $\mathbb{F}^D$ to denote the set of functions with domain $D$ and co-domain $\mathbb{F}$, i.e. the set of all $D$-vectors over $\mathbb{F}$.

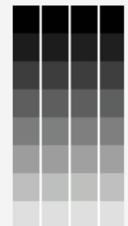
3. How does our author's Python implementation connect the vector definition with Python dictionaries?

4. Here is an example of a vector which uses pairs for its domain.

**Image** A black-and-white $1024 \times 768$ image can be viewed as a function from the set of pairs $\{(i,j) : 0 \leq i < 1024, 0 \leq j < 768\}$ to the real numbers, and hence as a vector. The pixel-coordinate pair $(i,j)$ maps to a number, called the *intensity* of pixel $(i,j)$. We will study several applications of representing images by vectors, e.g. subsampling, blurring, searching for a specified subimage, and face detection.

**Example 2.3.1:** As an example of a black and white image, consider an 4x8 gradient, represented as a vector in dictionary form (and as an image), where 0 is black and 255 is white:

```
{(0,0): 0,    (0,1): 0,    (0,2): 0,    (0,3): 0,
 (1,0): 32,   (1,1): 32,   (1,2): 32,   (1,3): 32,
 (2,0): 64,   (2,1): 64,   (2,2): 64,   (2,3): 64,
 (3,0): 96,   (3,1): 96,   (3,2): 96,   (3,3): 96,
 (4,0): 128,  (4,1): 128,  (4,2): 128,  (4,3): 128,
 (5,0): 160,  (5,1): 160,  (5,2): 160,  (5,3): 160,
 (6,0): 192,  (6,1): 192,  (6,2): 192,  (6,3): 192,
 (7,0): 224,  (7,1): 224,  (7,2): 224,  (7,3): 224}
```



5. What is the definition for adding $n$-vectors?

6. Why is addition of $n$-vectors commutative and associative?

7. How can we view $n$-vectors geometrically?

8. What is the definition for multiplication of an $n$-vector by a scalar?

9. What are the *distribution laws* for scalar-vector multiplication and vector addition?