Chapter 08, 8.1 – The Map Interface, Review Problem(s): ***6***

Chapter 08, 8.2 – Map Implementations

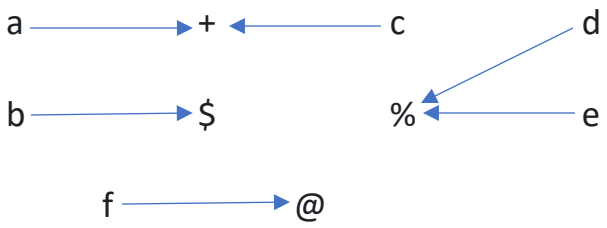
Chapter 08, 8.3 – Application: String-to-String Map

What do you remember about mathematical functions?

We'll specify that these are functions of one variable.

We can represent them as:

- A set of ordered pairs: { (a,+), (b,$), (c,+), (d,%), (e,%), (f,@) }
- An arrow diagram (note that each arrow corresponds to an ordered pair):



- A table of values:

domain	***co-domain***
a	+
b	$
c	+
d	%
e	%
f	@

In our example above, what do you notice about the elements of the domain?

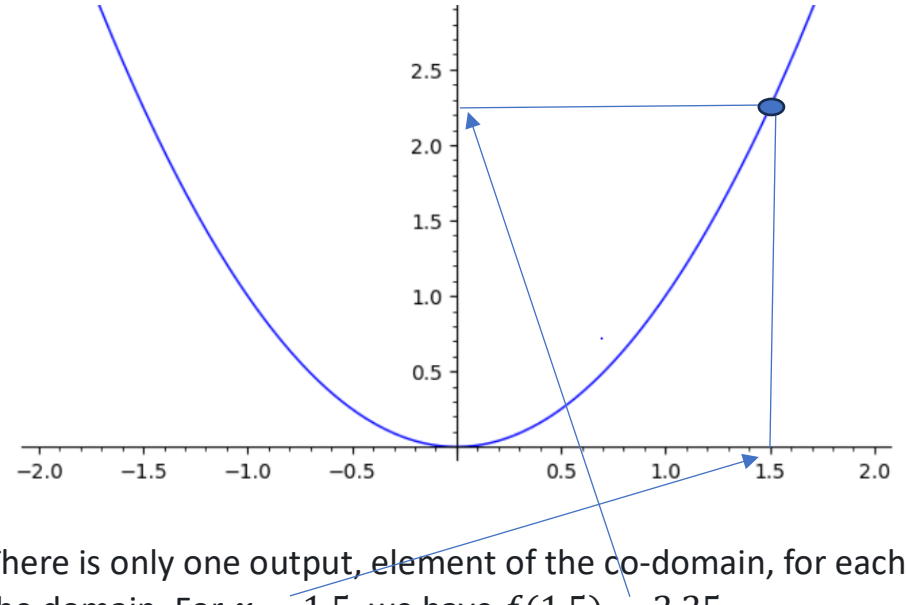
What do you notice about the elements of the co-domain?

Are there any repeated elements? Where?

If an element appears in the domain, it is only there once. But an element can be repeated in the co-domain.

Consider a "traditional" mathematical function, like $f(x) = x^2$.

Here it is plotted on the domain $-2 \leq x \leq 2$



There is only one output, element of the co-domain, for each input, or element of the domain. For $x = 1.5$, we have $f(1.5) = 2.25$.

You may remember this as "*the vertical line test*."

Back to our original example, what is one ordered pair we could add so this is no longer a function?

Several: (a,$), (b,+), and so on…

It is nice to review mathematical concepts like this from MA 116 – Discrete Structures, and one of the reasons it is a pre-requisite for this course, because we apply these concepts here. The theory of many concepts we discuss in CS 240 is from Discrete Mathematics.

Which leads us to our next ADT, Maps.

Maps are (***key***,***value***) pairs, which associate a key with exactly one value. Which sounds like our mathematical functions.

Maps have some of the same functionality as other Collection ADTs.

Chapter 08 PPT, pg 01-15.

We will explore the Map class in a lab. Create a new Java Project in Eclipse, Download the Map Lab files, and import them into your Java Project.

Look at each file, taking note of what it does.

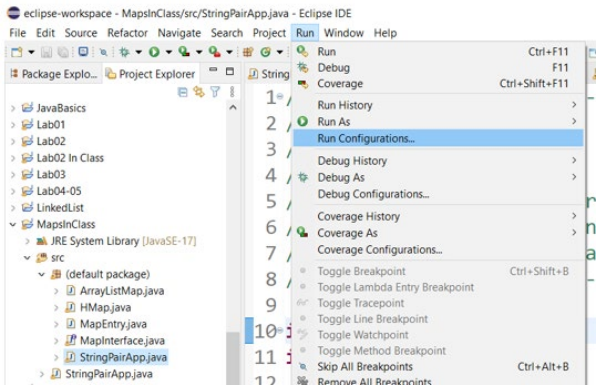
The application file (we would name it with a TD for "test driver") is `StringPairApp.java`. Take note of the `delimiters` defined in `main`.

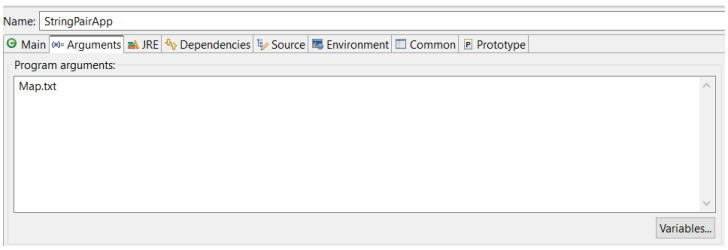
Run the application, `StringPairApp.java`. What happens? What do you need?

You need an input file, with key value pairs.

Create an ASCII text input file, and then copy it (no need to import it) into the folder for your Java Project (***not*** in `src` subfolder).

Then modify your command line arguments to read this file:





Start with defining a map for the function defined above. You will use the key-value pairs, and you can delimit inputs with the `#` or a new line. The first line in your input file needs to provide a name for the keys and values.

For example:

```
domain#co-domain
a#+
b#$
c#+
d#%
e#%
f#@
```

would define the function above. Run the `StringPairApp.java` application, inputting values which are and aren't in the domain.

Define your own Map function and run the `StringPairApp.java` application.