Sets, continued

We're going to continue with a few more useful *operations* on sets, both *binary* and *unary*. First, though, it might not be a bad idea to discuss exactly what we mean by *binary* and *unary operations*, which requires that we first define *order pairs*.

Definition. Given a set X, an *ordered pair* in X is a set of the form $\{\{x\}, \{x, y\}\}$ for $x, y \in X$, denoted (x, y). It is important to note the

Lemma. Let X be a set and let $x, y, x', y' \in X$. Then $(x, y) = (x', y') \Leftrightarrow x = x'$ and $y = y'$.

Proof. Suppose that $(x, y) = (x', y')$. Then what happens?

We can now consider another

Definition. Given a set S, a *unary operation* on S is a rule u which assigns to every element $s \in S$ of the set, a unique element of the set S, which we denote by $u(s)$. (This definition should remind you of the definition of *function* you probably heard in calculus; we'll come back to this similarity later.)

A *binary operation* on the set S is a rule b assigning to every ordered pair (s, s') of elements $s, s' \in S$, a unique element of the set S, which element we denote by $b(s, s')$.

Examples. We've already discussed the binary operations of *union* ($\cup$) and *intersection* ($\cap$) of sets. These are binary operations on the collection of all sets: applied to any two sets, we obtain a new set as a result. Note that we could write $\cup(S, T)$ for the union $S \cup T$, and $\cap(S, T)$ for the intersection $S \cap T$, but the latter notations are more standard.

Definition. Here's another binary operation: the *set difference* $S \setminus T$ yields the set

$$S \setminus T = \{s \in S \mid s \notin T\}.$$

Examples.

1. Find $S \setminus T$ if $S = \mathbb{Z}$ and $T = \mathbb{N}$.

2. What is $S \setminus T$ if $S \cap T = \emptyset$? Can you prove your assertion?

Some proofs involving sets are more easily visualized using *Venn diagrams*, charts which indicate the relationships between sets pictorially. In the space below, you should draw a Venn diagram illustrating each of the binary operations we've discussed so far:

Time for our first unary operation on sets:

Definition. Suppose that $A \in \mathcal{P}(S)$. (What does this mean?) The *complement of A in S*, denoted A^c, is the set

$$A^c = \{s \in S \mid s \notin A\}.$$

Notice how this definition relates to that of set difference.

Examples. The complement of a set behaves much like negation, as the following *De Morgan Laws* demonstrate:

1. Prove that for any subsets $A, B \subseteq S$, $(A \cup B)^c = A^c \cap B^c$. (*Note*: recall that by definition of set equality, you must show that the set on the left contains the one on the right, and *vice versa*.)

2. Prove that for any subsets $A, B \subseteq S$, $(A \cap B)^c = A^c \cup B^c$.

3. Prove that for any $A \subseteq S$, $A \cup A^c = S$.

4. What is $A \cap A^c$? Prove your assertion.

Here's one more important binary operation on the collection of all sets.

Definition. The *Cartesian product* of two sets A and B, denoted $A \times B$, is the collection

$$\{(a, b) \mid a \in A \text{ and } b \in B\}$$

of ordered pairs in which the first element is taken from A and the second from B. (If you want to get technical with the definition of "ordered pair" given above, we can think about drawing both a and b from the set $A \cup B$.) Recall that here *order matters.* For instance, $(1, 2) \neq (2, 1)$ for these two elements in $\mathbb{R} \times \mathbb{R}$.

Examples.

1. Compute both $S \times T$ and $T \times S$ if $S = \{1, 2, 3\}$ and $T = \{a, b\}$.

2. What is $S \times \emptyset$ for any S? What about $\emptyset \times S$?

Finally, let's wrap up with the following definition:

Definition. For any set S with finitely many elements in it, the *cardinality* of S, denoted either $|S|$ or $\#S$, is the number of elements it contains. For example, $|\{1, 3, 1007\}| = 3$, and $|\emptyset| = 0$.

Examples. Suppose that $|S| = m$ and $|T| = n$. What's the most you can say about...

1. $|S \cup T|$?

2. $|S \cap T|$?

3. $|S \times T|$?

We will be able to make the above assertions more precise after we've thoroughly discussed the topic of *functions*.