Unit C6 Supplement, 4/4/05

Language and the Set Representation of a Problem

After reading some (but not all) of your Module C Comprehensive Exercises, I noticed that some of you confused the notion of "language" and "problem." Probably, I did not emphasize this point sufficiently. So, here is another supplement to clarify the connection. A language (set of strings) can always be seen as a problem (naturally, a set representation). For example, 0^n1^n = $\{0^n1^n \mid n$ is a natural number$\}$ is a language and a problem at the same time. However, not all problems in set notation can be a language. For example, a problem $ACCEPT_{TM} = \{(M, w) \mid$ TM M accepts string $w\}$ is not a language because a member of this problem is a "pair," not a string (even though the pair contains a string). Some problems would be represented as a set of more complex mathematical structures ("tuples" of various components). To analyze problems with respect to our extended Chomsky hierarchy, we must represent problems not just as a set, but as a language, because that is what grammars and automata are supposed to specify/process.

In some cases, it would be more natural to represent a problem using a mathematical structure as in $ACCEPT_{TM}$. In many cases, we can still modify the problem as a language. For example, by considering the string representation of the formal definition of a TM, we might represent $ACCEPT_{TM}$ as $\{M\#w \mid$ TM corresponding to the formal definition M accepts string $w\}$, which can be processed by a TM. Another example would be binary addition. We can represent it as a problem (but not language), $\{(x, y, z) \mid x + y = z$, where x, y, z are non-negative binary numbers (values)$\}$, or a language/problem, $\{x\#y\#z \mid val(x) + val(y) = val(z)$, where x, y, z are non-negative binary numbers (strings) whose values are evaluated with the function $val\}$, which can be processed by a TM (or possibly some other automata).

// End