int.1 The Syntax of the Lambda Calculus

lam:int:syn:sec One starts with a sequence of variables x, y, z, ... and some constant symbols a, b, c, ... The set of terms is defined inductively, as follows:

1. Each variable is a term.
2. Each constant is a term.
3. If M and N are terms, so is (MN).
4. If M is a term and x is a variable, then $(\lambda x. M)$ is a term.

The system without any constants at all is called the *pure* lambda calculus.

We will follow a few notational conventions:

Convention 1.

1. When parentheses are left out, application takes place from left to right. For example, if M, N, P, and Q are terms, then $MNPQ$ abbreviates $(((MN)P)Q)$.
2. Again, when parentheses are left out, lambda abstraction is to be given the widest scope possible. From example, $\lambda x. MNP$ is read $\lambda x. (MNP)$.
3. A lambda can be used to abstract multiple variables. For example, $\lambda xyz. M$ is short for $\lambda x. \lambda y. \lambda z. M$.

For example,

$$\lambda xy. xxyx\lambda z. xz$$

abbreviates

$$\lambda x. \lambda y. ((((xx)y)x)\lambda z. (xz)).$$

You should memorize these conventions. They will drive you crazy at first, but you will get used to them, and after a while they will drive you less crazy than having to deal with a morass of parentheses.

Two terms that differ only in the names of the bound variables are called α-equivalent; for example, $\lambda x. x$ and $\lambda y. y$. It will be convenient to think of these as being the "same" term; in other words, when we say that M and N are the same, we also mean "up to renamings of the bound variables." Variables that are in the scope of a λ are called "bound", while others are called "free." There are no free variables in the previous example; but in

$$(\lambda z. yz)x$$

y and x are free, and z is bound.

Photo Credits

Bibliography