1 Syntax of lambda calculus

Assume a countable set of variable names, denoted by $a, b, c, ..., x, y, z, a_0, a_1, ...$

Definition. A **λ-term** is defined by the following context-free grammar:

$$\begin{array}{lcl} < term > & := & < name > \\ & | & (\lambda < name > . < term >) \\ & | & (< term >< term >) \end{array}$$

Conventions.

1. **Function application** is left-associative, so $(((A_1 A_2)A_3)...A_k)$ can be abbreviated as $A_1 A_2 A_3 ... A_k$

2. Nested **abstractions** $(\lambda x_1.(\lambda x_2.(...\lambda x_k.A)...))$ can be abbreviated as $\lambda x_1 x_2 ... x_k.A$

Example.
$\lambda xy.FAB$ means $((\lambda x.(\lambda y.F))A)B$

2 Free variables

1. $< name >$ is free in $< name >$

2. $< name >$ is free in $\lambda < name' > . < term >$ if $< name > \neq < name' >$ and $< name >$ is free in $< term >$

3. $< name >$ is free in $< term' >< term'' >$ if $< name >$ is free in $< term' >$ or $< name >$ is free in $< term'' >$

3 Bound variables