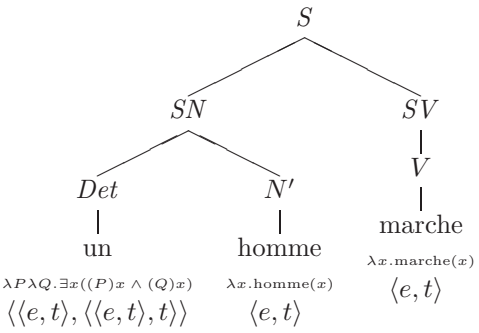
Lambda-calcul typé

L'objectif est maintenant de mettre en œuvre la méthode de calcul compositionnelle vue en cours de sémantique l'an dernier.

Rappel

L'ensemble des expressions interprétables (*meaningful expressions*) de type a, ME_a, est défini inductivement :
- Pour chaque type a, les variables et les constantes de type a sont dans ME_a.
- Pour tous types a et b, si $\alpha \in ME_{\langle a,b\rangle}$ et $\beta \in ME_a$ alors $(\alpha)\beta \in ME_b$.
- Pour tous types a et b, si u est une variable de type a et $\alpha \in ME_b$, alors $\lambda u.\alpha$ est dans $ME_{\langle a,b\rangle}$.
- Si φ et ψ sont dans ME_t, alors les expressions suivantes sont aussi dans ME_t : $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$.
- Pour tout type a, si φ est dans ME_t et u est une variable de type a, alors $\forall u\varphi$ et $\exists u\varphi$ sont dans ME_t.

Il faut noter que l'application fonctionnelle peut être vue ici comme une généralisation de l'application des prédicats à leurs arguments : la définition ci-dessus donne bien comme appartenant au langage la formule 'dormir(j)', à condition de la noter '(dormir)j'.
Plus précisément, il faut admettre que 'dormir' est une **constante** de type $\langle e,t\rangle$, et j une constante de type e. Alors, en vertu de la deuxième règle plus haut, 'dormir(j)' est bien dans ME_t. De même, en recourant à une « curryfication », on note les prédicats binaires comme une double application fonctionnelle : « *Jean aime Marie* » s'écrit $((\text{aimer})j)m$.



S	$\rightarrow$	SN	SV
0	$\leftarrow$	(1)	2
SN	$\rightarrow$	Det	N'
0	$\leftarrow$	(1)	2
SV	$\rightarrow$	V	
0	$\leftarrow$	1	

Implémentation

L'implémentation des lambda-termes s'inspire de la représentation choisie précédemment pour les formules du 1er ordre. Il faut simplement envisager des nœuds de types nouveaux : λ (qui fonctionne comme un quantificateur), et application fonctionnelle.
Une fois la représentation en mémoire des termes établie, il faut implémenter la β-réduction : il s'agit d'un parcours d'arbre avec substitution de terme (il faut prévoir la recopie de sous-arbres).

Utilisation de `yacc`

Pour faciliter la mise au point, on pourra utiliser `yacc`, pour la saisie et la construction en mémoire de formules du lambda-calcul typé.

Conventions :		
	prédicats	Identificateurs en minuscules
	variables (type e)	Lettres minuscules typées dans le lexique
	variables (autres types)	Lettres majuscules typées dans le lexique (sauf A, E, L)
	constantes	Lettres minuscules typées dans le lexique
	$\wedge, \vee, \rightarrow, \neg$	`&`, `|`, `->`, `~`
	$\forall, \exists$	`A`, `E`
	λ	`L`
	$(M)N$	`[`M`]`N
	Point	`.`

$\big(\lambda P\lambda Q.\exists x\ (P(x) \rightarrow Q(x,a))\big)\ \lambda y.\text{homme}(y)$: `[L P L Q . E x (P x -> Q x a)] Ly. homme y`