

Interprétation des programmes – TP 5 :

De Hobix à Fopix

Université Paris Diderot – Master 1

(2015-2016)

Cette séance de travaux pratiques a pour objectifs de :

- vous faire comprendre la sémantique de FOPIX ;
- vous faire implémenter l'explicitation des fermetures.

Remarque : Si vous n'avez pas terminé le TP3, vous pouvez désactiver la vérification des types de HOPIX à l'aide de l'option `--typechecking false` de flap.

Exercice 1 (Sémantique de Fopix)

1. Quelles sont les différences entre les langages HOBIX et FOPIX ?
2. Que contient l'environnement d'évaluation de l'interprète de FOPIX ?
3. Complétez les cas manquants de l'interprète de FOPIX.

□

Exercice 2 (Explicitation des fermetures)

1. Écrire sur papier le code compilé FOPIX correspondant au programme HOBIX suivant :

```
val f :=  
  val z := 3 * 6;  
  val y := z * 2;  
  \ x => x * y.  
val u := f 3.
```

2. Écrire sur papier le code compilé FOPIX correspondant au programme HOBIX suivant :

```
val p := 6 * 7.  
val f :=  
  val z := 3 * 6;  
  val y := z * 2;  
  \ x => \ o => x * z + p * o.  
val g := f 3.  
val v := g 4.
```

3. Compléter le cas des variables dans la fonction `HobixToFopix.expression`.
4. Compléter le cas de l'application dans la fonction `HobixToFopix.expression`.
5. Compléter le cas des fonctions anonymes dans la fonction `HobixToFopix.expression`.
6. Écrire sur papier le code compilé FOPIX correspondant au programme HOBIX suivant :

```
rec f x := if x <= 3 then g (x - 1) else h (x - 2)  
and g x := if x <= 2 then f (x - 1) else h (x - 2)  
and h x := if x <= 1 then 1 else f (x - 1).  
val y := f 3.  
val z := g 3.  
val t := h 3.
```

7. Compléter le cas des fonctions mutuellement récursives dans la fonction `HopixToHobix.expression`.

□