Nom :

Prénom :

Devoir surveillé

Lambda-calcul, langage CAML

Vous devez répondre dans les cadres aux questions posées. Documents de cours et notes personnelles autorisés. Téléphones portables, ordinateurs et calculatrices interdits.

Partie 1 : lambda-calcul

Exercice 1 : 2 points

Dans le terme suivant, entourez les abstractions d'un cadre pointillé et les applications d'un cadre en trait continu.

$$x\ \lambda x.\ \lambda y.\ y\ x$$

Exercice 2 : 2 points

Dans le terme suivant, entourez les variables libres et indiquez les occurrences de variables liées ensembles.

$$x\ \lambda x.\ \lambda y.\ y\ x$$

Exercice 3 : 2 points

Parmi les termes suivants, entourez ceux qui sont équivalents.

$\lambda t.(\lambda y.ty)tx$ $\lambda x.(\lambda y.xy)tx$

$\lambda z.(\lambda w.zw)zx$

Exercice 4 : 2 points

Pour chacun des termes suivants, entourez toutes les occurrences de variables qui *doivent* être renommées.

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

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

Exercice 5 : 2 points

Évaluez le terme suivant en réalisant toutes les étapes de sa réduction.

$(\lambda x.\lambda y.yx)z(\lambda t.t)$

Partie 2 : langage CAML

Exercice 6 : 4 points

Complétez la définition suivante de manière à ce que, appliquée à une liste d'entiers w et à une fonction f associant à tout entier un Booléen, la fonction **filtre** retourne la liste de tous les éléments de w auxquels f associe la valeur true.

Par exemple, si f associe la valeur true à tout entier positif ou nul, alors filtre [1,-2,0,-1,3] doit retourner la liste [1,0,3].

```
let rec filtre w f =




;;
```

Exercice 7 : 6 points

On représente de la manière suivante une formule Booléenne ne comportant que des connecteurs et, ou, non et des constantes.

```
type formule =
E of bool | Non of formule | Et of formule * formule | Ou of formule * formule;;
```

Par exemple, la formule Booléenne ((non 0) et 1) se représente :

`Et(Non(E(false)),E(true))`.

Complétez la définition suivante de manière à ce que, appliquée à une formule représentée par le type défini ci-dessus, la fonction **eval** retourne la valeur de cette formule.

Par exemple,

`eval (Et(Non(E(false)),E(true)))`

doit retourner **true**.

On rappelle qu'en CAML, les constantes Booléenes sont notées **true** et **false** et que les opérateurs et, ou, non sont notés respectivement **&&, or, not**.

```
let rec eval f =
match f with
|E(v) ->

|Non(x) ->


|Et(x,y) ->


|Ou(x,y) ->


;;
```

On appelle taille d'une formule le nombre de connecteurs logique qu'elle comporte. Par exemple, la formule

`(Et(Non(E(false)),E(true)))`

a une taille égale à 2 parce qu'elle comporte deux connecteurs, un "et" et un "non".

Complétez la définition suivante de manière à ce que, appliquée à une formule représentée par le type défini ci-dessus, la fonction **taille** retourne la taille de cette formule.

```
let rec taille f =




;;
```