+1/1/60+

Paradigmes et interprétation - partie 1/3

Examen blanc

L2/L3 informatique – Université Côte d'Azur

Durée: 40 minutes (1/3 de 2 heures).

- Aucun document ni aucune machine ne sont autorisés. Un mémo est donné en fin de sujet.
- Les téléphones doivent être rangés.
- Vous pouvez déborder du cadre de réponses si besoin, sans écrire sur les cases réservées à la correction en haut du cadre.
- Les réponses doivent être écrites lisiblement, en respectant autant que possible l'interligne proposé. Le correcteur blanc et l'effaceur sont autorisés.
- Les questions sont indépendantes: une question peut être sautée, et une fonction `f` demandée à une question peut être utilisée pour définir une fonction `g` dans une question ultérieure même si la solution pour `f` n'a pas été trouvée.
- Le barème est une estimation basse, il pourra être revu à la hausse (note > 20 possible dans ce cas). Le nombre de points d'une question est proportionnel à l'estimation de la difficulté de la question.

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

← **Codez ci-contre votre numéro d'étudiant : cochez dans la première colonne le premier chiffre (a priori un 2), puis dans la deuxième colonne le deuxième chiffre, etc**

Date de naissance ou pseudo[a]:

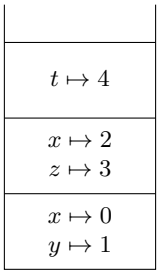
.....

[a] utilisé en cas de problème pour vous retrouver avec le numéro d'étudiant. Merci de garder l'anonymat et de ne pas donner un pseudo qui permet de vous identifier

Pile des espaces de noms (2 points)

On suppose que l'interpréteur du langage impératif vu en cours accepte une instruction `print_stack` qui affiche la pile des espaces de noms (la "pile d'appel") dans l'état où elle se trouve au moment où l'instruction est évaluée. Par exemple, le programme ci-dessous affiche la pile des espaces de noms suivante.

```
{
  let x = 0;
  let y = 1; {
    let x = 2;
    let z = 3; {
      let t = 4;
      print_stack
    }
  }
}
```



On considère maintenant le programme ci-dessous.

```
{ let x = 0; let y = 1;
    { let x = 2; let z = 3 };
    { let t = 4; let x = 5; print_stack; x+y+t}}
```

Question 1 (0,5 pts) Dessinez la pile des espaces de noms affichée par ce programme.

☐0 ☐1 *Cases réservées à la correction*

Question 2 (0,5 pts) Quelle est le résultat de l'évaluation de `x+y+t`?

☐0 ☐1 *Cases réservées à la correction*

.....

On souhaite écrire un interpréteur pour notre langage jouet en Python, et on choisit de représenter la pile des espaces de noms par une liste de dictionnaires. Par exemple, la pile dessinée plus haut est représentée par

```
namespace_stack = [{'x': 0, 'y': 1}, {'x': 2, 'z': 3}, {'t': 4}]
```

Notez que le sommet de pile se trouve en fin de liste (les listes Python sont des tableaux extensibles).

Question 3 (1 pts) Définissez en Python une fonction `eval_var(var_id, ns_stack)` qui prend en arguments un identifiant de variable et une pile d'espaces de noms et qui renvoie la valeur de cet identifiant de variable. Par exemple, `eval_var('z',namespace_stack)`. renvoie 3, et `eval_var('a',namespace_stack)` renvoie `"undefined identifier"`.

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 *Cases réservées à la correction*

.....

.....

.....

.....

.....

.....

Fuite mémoire (2 points)

Question 4 (2 pts) Donnez un exemple de programme donnant lieu à une fuite mémoire. Vous pouvez utiliser le langage de votre choix parmi ceux que vous avez étudiés à l'université, pourvu que ce soit un langage dans lequel il soit possible de faire une fuite mémoire (indication: votre choix est assez restreint). Vous pouvez aussi utiliser le langage jouet vu en cours, dont un exemple est donné dans le mémo.

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 *Cases réservées à la correction*

.....

.....

.....

.....

Le barbier de Datalog (3 points)

On considère le programme Datalog suivant.

```
1. rase(adam, bob).  % adam rase bob
2. rase(charlie, charlie).  % charlie se rase tout seul
3. differents(X, X) :- !, fail.
4. differents(X, Y).
5. barbier(X) :- rase(X,Y), differents(X, Y).
```

Question 5 (1 pts) Dessinez l' arbre de recherche de la requête `?- barbier(adam)`, en précisant à chaque fois les lignes de programme et les substitutions utilisées ainsi que les branches coupées (voir exemple dans le mémo). Précisez si la requête termine par un succès ou un échec.

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 *Cases réservées à la correction*

Question 6 (1 pts) Même question pour la requête `?- barbier(bob).`

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 *Cases réservées à la correction*

Question 7 (1 pts) Même question pour la requête `?- barbier(charlie).`

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 *Cases réservées à la correction*

Memo sur Python

```
>>> L = [1,2,3]
>>> L[0]
1
>>> L[-1]
3
>>> for i in range(1, len(L)+1):
        print(L[-i])
3
2
1
>>> D = {'a':1, 'b':2, 'c':3}
>>> D['a']
1
>>> 'a' in D
True
>>> 'z' in D
False
```

Memo sur le langage du cours 2 avec désallocation explicite

```
imp # let x = Box::new(1)
x : Box<int> = @0(1)
imp # #dump_heap
[1, ∅, ∅, ∅]
imp # free(x)
- : unit = ()
imp # #dump_heap
[∅, ∅, ∅, ∅]
imp # *x = 2
Evaluation Error : dangling pointer error: address 0 is not allocated
imp # free(x)
Evaluation Error : double free or dangling pointer error: trying to free 0 that is not allocated
```

Memo sur Prolog On numérote les lignes par comodité.

```
1. a(1,1).
2. a(2,2).
3. b(X,Y) :- a(X,X), a(Y,Y), a(X,Y), !.
4. ?- b(Z,T).
```