Correction du partiel TYLA

EPITA – Ing1 promotion 2015 – **Sans documents ni machine**

Avril 2013 (1h30)

Correction: Le sujet et sa correction ont été écrits par Roland Levillain.

Bien lire les questions, chaque mot est important. Écrire court, juste, et bien. Une argumentation informelle mais convaincante est souvent suffisante.

Répondre aux questions sur les formulaires de QCM ; aucune réponse manuscrite ne sera corrigée. Ne pas oublier de renseigner les champs d'identité. Il y a exactement une et une seule réponse juste pour ces questions. Si plusieurs réponses sont valides, sélectionner la plus restrictive. Par exemple s'il est demandé si 0 est *nul*, *non nul*, *positif*, ou *négatif*, cocher *nul* qui est plus restrictif que *positif* et *négatif*, tous deux vrais. Répondre incorrectement à une question à choix multiples est plus pénalisé que de ne pas répondre.

Cette correction contient 5 pages. Les pages 1–4 contiennent l'épreuve et son corrigé. Ce document comporte en page 5 une enquête (facultative) sur le cours ; y répondre sur la feuille de QCM.

1 Histoire & langages

Q.1 À quel langage peut-on associer Grace Hopper ?

✓ COBOL ✗ FORTRAN ✗ Pascal ✗ Smalltalk

Q.2 Lequel de ces langages ne peut-on associer à John Backus ?

✗ ALGOL ✓ APL ✗ BNF ✗ FORTRAN

Q.3 Qui a conçu le langage C ?

✗ Donald Knuth ✓ Dennis Ritchie
✗ Brian Kernighan ✗ Ken Thompson

Q.4 Quel langage a introduit la structure de blocs (fonctions imbriquées et variables non locales) ?

✓ ALGOL ✗ BASIC ✗ COBOL ✗ FORTRAN

Q.5 Lequel de ces langages est typé statiquement ?

✓ C ✗ Javascript ✗ Python ✗ TEX

2 Fonctions

Q.6 Le support des fonctions récursives nécessite

✓ une pile. ✗ un *static link*.
✗ des types fonctions. ✗ des *forward declarations*.

Q.7 Où sont normalement placés les arguments formels d'une fonction ?

✗ Dans le tas ✗ Dans l'espace noyau
✓ Sur la pile ✗ Dans le segment de données

Q.8 En C, la ligne suivante

```
int foo = getc() + getc() * getc();
```

✗ est invalide.
✗ n'est pas déterministe.
✗ produit le même résultat partout.
✓ dépend de la plate-forme et/ou de l'implémentation.

Q.9 La liaison retardée en C++

✗ repose sur `template`.
✗ s'appuie sur `dynamic_cast`.
✗ va de pair avec la surcharge.
✓ est liée aux méthodes polymorphes.

Q.10 Comment peut-on réaliser un passage par nom en C++ ?

✓ Avec des macros.
✗ Avec des objets fonctions.
✗ Avec des références constantes.
✗ Il n'y a aucun moyen d'effectuer un passage par nom en C++.

Q.11 À la fin de ce programme, avec un *Mode* de passage des arguments par valeur (copie), quelles sont les valeurs des l-values ?

```
var t    : integer
    foo  : array [1..2] of integer;

procedure shoot_my(x : Mode integer);
begin
  foo[1] := 5;
  t      := 1;
  x      := x - 1;
end;
```

```
begin
  foo[1] := 0;
  foo[2] := 1;
  t      := 2;
  shoot_my(foo[t]);
end.
```

✗ `foo[1] = 0, foo[2] = 1, t = 1`
✗ `foo[1] = 4, foo[2] = 1, t = 1`
✗ `foo[1] = 5, foo[2] = 0, t = 1`
✓ `foo[1] = 5, foo[2] = 1, t = 1`

Q.12 Même question, mais avec un *Mode* de passage d'arguments par valeur-résultat, à la Algol W. On rappelle qu'en Algol W, la l-value dans laquelle est copiée la valeur d'un argument passé par résultat (`out`) est évaluée *au retour* de la fonction.

✓ `foo[1] = 0, foo[2] = 1, t = 1`
✗ `foo[1] = 4, foo[2] = 1, t = 1`
✗ `foo[1] = 5, foo[2] = 0, t = 1`
✗ `foo[1] = 5, foo[2] = 1, t = 1`

Q.13 Même question, mais avec un *Mode* de passage d'arguments par valeur-résultat, à la Ada. On rappelle qu'en Ada, la l-value dans laquelle est copiée la valeur d'un argument passé par résultat (`out`) est évaluée *à l'appel* de la fonction.

✗ `foo[1] = 0, foo[2] = 1, t = 1`
✗ `foo[1] = 4, foo[2] = 1, t = 1`
✓ `foo[1] = 5, foo[2] = 0, t = 1`
✗ `foo[1] = 5, foo[2] = 1, t = 1`

Q.14 Même question, mais avec un *Mode* de passage d'arguments par référence.

✗ `foo[1] = 0, foo[2] = 1, t = 1`
✗ `foo[1] = 4, foo[2] = 1, t = 1`
✓ `foo[1] = 5, foo[2] = 0, t = 1`
✗ `foo[1] = 5, foo[2] = 1, t = 1`

Q.15 Même question, mais avec un *Mode* de passage d'arguments par nom.

✗ `foo[1] = 0, foo[2] = 1, t = 1`
✓ `foo[1] = 4, foo[2] = 1, t = 1`
✗ `foo[1] = 5, foo[2] = 0, t = 1`
✗ `foo[1] = 5, foo[2] = 1, t = 1`

3 Programmation fonctionnelle

Q.16 Quel trait du langage Haskell permet d'exprimer des « listes infinies » ?

- ✗ Le sucre syntaxique
- ✓ L'évaluation paresseuse
- ✗ La compilation séparée
- ✗ La séparation interface/implémentation

Q.17 Comment peut-on implémenter des « fonctions anonymes » dans un langage qui en est dépourvu ?

- ✓ Avec des objets fonctions
- ✗ À l'aide d'un mot-clef comme `auto`
- ✗ Avec des fonctions à liste variable d'arguments
- ✗ En utilisant la compilation à la volée (*just-in-time compilation*)

Q.18 Comment s'appelle la transformation suivante ?

$$\lambda(x, y) \mapsto x \log y \quad \rightsquigarrow \quad \lambda x(\lambda y \mapsto x \log y)$$

- ✗ Application
- ✗ Décomposition
- ✓ Curryfication
- ✗ Linéarisation

4 Programmation orientée objet

Q.19 Dans quel langage a été introduit la notion d'« objet » ?

- ✗ ALGOL
- ✗ C++
- ✓ Simula
- ✗ Smalltalk

Q.20 Dans quel langage a été introduit la notion de « classe » ?

- ✗ ALGOL
- ✗ C++
- ✓ Simula
- ✗ Smalltalk

Q.21 Le typage en Smalltalk est

- ✓ inexistant.
- ✗ dynamique.
- ✗ statique.
- ✗ statique fort.

Q.22 En Smalltalk 76, comment instancie-t-on une classe ?

- ✗ En appelant son constructeur.
- ✗ Grâce à la primitive '`create`'.
- ✓ En envoyant un message '`new`' à la classe `Class`.
- ✗ En envoyant un message '`new`' à la classe `Object`.

Q.23 La relation « EST-UN » s'exprime via

- ✗ les concepts.
- ✗ la coercition.
- ✓ l'héritage.
- ✗ l'agrégation et la délégation.

Q.24 Quel langage fournit des multiméthodes ?

- ✗ C++
- ✗ Java
- ✓ CLOS (Common Lisp)
- ✗ Simula

5 Programmation générique

Q.25 Dans quel langage doit-on explicitement instancier les types paramétrés avant de pouvoir les utiliser ?

- ✓ Ada
- ✗ C++ 1998
- ✗ C++ 2011
- ✗ Java

Q.26 Quel langage ne permet pas d'imposer des contraintes sur les paramètres d'un type ?

- ✗ Ada
- ✓ C++
- ✗ Eiffel
- ✗ Java

Q.27 En C++, comment appelle-t-on la possibilité de fournir une version alternative d'un template pour une combinaison de paramètres effectifs particulière ?

- ✗ La surcharge
- ✗ L'inférence de type
- ✓ La spécialisation explicite
- ✗ Le polymorphisme d'inclusion

Q.28 En C++, un concept est

- ✗ une classe abstraite.
- ✗ une instance de classe paramétrée.
- ✗ un jeu de paramètres effectifs pour un template.
- ✓ un ensemble de règles sur un (ou plusieurs) paramètre(s).

Q.29 Parmi les lignes C++ suivantes, laquelle est invalide ?

- ✓ `std::pair p1(42, 51);`
- ✗ `std::pair<float, int> p2(42, 3.14f);`
- ✗ `std::pair<int, float> p3 = std::make_pair(42, 3.14f);`
- ✗ `std::pair<int, float> p4 = std::make_pair<int, float>(2.72f, 51);`

Q.30 La métaprogrammation statique

- ✗ est un oxymore.
- ✗ permet l'introspection de types dynamiques.
- ✗ permet de compiler des programmes à l'exécution.
- ✓ permet d'exécuter des programmes à la compilation.

6 À propos de ce cours

Pour terminer cette épreuve, nous vous invitons à répondre à un petit questionnaire. Les renseignements ci-dessous ne seront bien entendu pas utilisés pour noter votre copie. Ils ne sont pas anonymes, car nous souhaitons pouvoir confronter réponses et notes. En échange, quelques points seront attribués pour avoir répondu. Merci d'avance.

Sauf indication contraire, vous pouvez cocher plusieurs réponses par question. Répondez sur la feuille de QCM. N'y passez pas plus de dix minutes.

Cette épreuve

Q.31 Sans compter le temps mis pour remplir ce questionnaire, combien de temps ce partiel vous a-t-il demandé (si vous avez terminé dans les temps), ou combien vous aurait-il demandé (si vous aviez eu un peu plus de temps pour terminer) ?

a. Moins de 30 minutes.
b. Entre 30 et 60 minutes.
c. Entre 60 et 90 minutes.
d. Entre 90 et 120 minutes.
e. Plus de 120 minutes.

Q.32 Ce partiel vous a paru

a. Trop difficile.
b. Assez difficile.
c. D'une difficulté normale.
d. Assez facile.
e. Trop facile.

Le cours

Q.33 Quelle a été votre implication dans les cours TYLA ?

a. Rien.
b. Bachotage récent.
c. Relu les notes entre chaque cours.
d. Fait les annales.
e. Lu d'autres sources.

Q.34 Ce cours

a. Est incompréhensible et j'ai rapidement abandonné.
b. Est difficile à suivre mais j'essaie.
c. Est facile à suivre une fois qu'on a compris le truc.
d. Est trop élémentaire.

Q.35 Ce cours

a. Ne m'a donné aucune satisfaction.
b. N'a aucun intérêt dans ma formation.
c. Est une agréable curiosité.
d. Est nécessaire mais pas intéressant.
e. Je le recommande.

Q.36 La charge générale du cours en sus de la présence en amphi (relecture de notes, compréhension, recherches supplémentaires, etc.) est

a. Telle que je n'ai pas pu suivre du tout.
b. Lourde (plusieurs heures de travail par semaine).
c. Supportable (environ une heure par semaine).
d. Légère (quelques minutes par semaine).

Les formateurs

Q.37 L'enseignant

a. N'est pas pédagogue.
b. Parle à des étudiants qui sont au dessus de mon niveau.
c. Me parle.
d. Se répète vraiment trop.
e. Se contente de trop simple et devrait pousser le niveau vers le haut.