

Correction du partiel TYLA

EPITA – AppIng1 promotion 2014 – **Sans documents ni machine**

Juin 2012 (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 *négalif*, cocher *nul* qui est plus restrictif que *positif* et *négalif*, 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 Langages

Q.1 Quel langage ne repose pas sur une machine virtuelle pour assurer l'exécution du code ?

- | | | | |
|--|--|--|---|
| ☒ C# | ☒ Go | ☒ Perl | ☒ Pascal UCSD |
|--|--|--|---|

Q.2 Lequel de ces langages est typé statiquement ?

- | | | | |
|--|--|---|--|
| ☒ Javascript | ☒ MATLAB | ☒ OCaml | ☒ Ruby |
|--|--|---|--|

Q.3 À quel nom ne peut-on associer John Backus ?

- | | | | |
|---|---|---|---|
| ☒ ALGOL | ☒ BNF | ☒ COBOL | ☒ FORTRAN |
|---|---|---|---|

2 Fonctions

Q.4 Le support des fonctions récursives nécessite

- | | |
|--|---|
| ☒ une pile. | ☒ un <i>static link</i> . |
| ☒ des types fonctions. | ☒ des <i>forward declarations</i> . |

Q.5 Les variables non locales sont permises grâce à

- | | |
|---|--|
| ☒ <i>auto</i> . | ☒ la structure de blocs. |
| ☒ <i>static</i> . | ☒ la déclaration en blocs. |

Q.6 En C, la ligne suivante

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

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

Q.7 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] = 5, foo[2] = 0, t = 1
☒ foo[1] = 4, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 1, t = 1

Q.8 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] = 5, foo[2] = 0, t = 1
☒ foo[1] = 4, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 1, t = 1

Q.9 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] = 5, foo[2] = 0, t = 1
☒ foo[1] = 4, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 1, t = 1

Q.10 Même question, mais avec un *Mode* de passage d'arguments par référence.

- ☒ foo[1] = 0, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 0, t = 1
☒ foo[1] = 4, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 1, t = 1

Q.11 Même question, mais avec un *Mode* de passage d'arguments par nom.

- ☒ foo[1] = 0, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 0, t = 1
☒ foo[1] = 4, foo[2] = 1, t = 1 ☒ foo[1] = 5, foo[2] = 1, t = 1

Q.12 Comment peut-on simuler le passage par nom en C++ ?

- ☒ Avec des macros. ☒ Il n'y a aucun moyen de simuler du passage par nom en C++.
☒ Avec des concepts.
☒ En utilisant void*.

Q.13 La liaison retardée en C++

- ☒ repose sur template. ☒ a un rapport avec virtual.
☒ s'appuie sur dynamic_cast. ☒ est liée à la surcharge des opérateurs.

3 Programmation fonctionnelle

Q.14 Quel trait du langage Haskell permet d'écrire le programme suivant, calculant la liste des entiers naturels ?

```
reapply f a = a : reapply f (f a)
incr x = x + 1
integers = reapply incr 0
```

- ☒ L'évaluation stricte. ☒ La compilation séparée.
☒ L'évaluation paresseuse. ☒ La séparation interface/implémentation.

Q.15 Quel langage ne propose pas de « fonctions anonymes » ?

- ✗ C++ 2011. ✗ C# 3.0. ✓ Java 7. ✗ Python 3.0.

Q.16 Comment appelle-t-on une fonction qui capture des références à des variables libres dans l'environnement lexical ?

- ✓ Une fermeture. ✗ Une fonction involutive.
✗ Une lambda abstraction. ✗ Une fonction d'ordre supérieur.

4 Programmation orientée objet

Q.17 Dans quel langage a été introduit la notion de « classe » ?

- ✗ CLU ✗ COBOL ✓ Simula ✗ Smalltalk

Q.18 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.19 Le typage en Smalltalk est

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

5 Programmation générique

Q.20 Dans quel langage doit-on explicitement instancier les types paramétrés avant de pouvoir les utiliser ?

- ✓ Ada ✗ C++ 1998/2003 ✗ C++ 2011 ✗ Eiffel

Q.21 Quel langage ne dispose pas d'une fonctionnalité conçue pour contraindre les paramètres des types paramétrés ?

- ✗ Ada ✓ C++ ✗ Eiffel ✗ Java

Q.22 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. ✗ Le polymorphisme *ad hoc*.
✗ La substitution. ✓ La spécialisation explicite.

Q.23 Parmi les termes suivants, lequel ne peut pas être utilisé comme paramètre effectif d'un template de classe en C++ 2011 ?

- ✗ 42. ✗ 51u. ✓ auto. ✗ long long.

Q.24 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.25 Le code C++ ci-dessous :

```
#include <complex>
#include <vector>
#include <algorithm>

int
main()
{
```

```
std::vector< std::complex<float> > v;  
v.push_back (std::complex<float>(1.f, 1.f));  
v.push_back (std::complex<float>(42.f, 51.f));  
v.push_back (std::complex<float>(4.3f, -2.2f));  
std::sort (v.begin(), v.end());  
}
```

- ✗ compile correctement et s'exécute sans erreur.
- ✗ compile correctement mais provoque une erreur à l'exécution.
- ✓ provoque une erreur de compilation mentionnant l'absence d'un 'operator<'.
- ✗ provoque une erreur de compilation mentionnant explicitement l'inadéquation entre `std::complex<float>::iterator` et le concept `RandomAccessIterator`.

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.26 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.27 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.28 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.29 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.30 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.31 La charge générale du cours en sus de la présence en amphitheâtre (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.32 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.