TYLA — Typologie des Langages
EPITA_ING1_2013_S2_TYLA

EPITA 2013 – **Sans document ni machine**

Juin 2011 (1h00)

Répondre sur les formulaires de QCM ; aucune réponse manuscrite ne sera corrigée. Renseigner les champs d'identité. Bien lire les questions, chaque mot est important. Il y a une seule réponse juste pour ces questions. Lorsque plusieurs 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 est plus pénalisé que de ne pas répondre.

1 Programmation orientée objet

Q.1 Le type dynamique d'un objet
- a. est un sous-type de son type statique.
- b. est un sur-type de son type statique.
- c. est connu à la compilation.
- d. est utilisé pour distinguer des fonctions/méthodes surchargées.

Q.2 Dans quel langage les appels de méthodes ne sont pas vérifiés statiquement ?

a. C++ b. C# c. Java d. Simula e. Smalltalk

Q.3 Qu'appelle-t-on une métaclasse en Smalltalk ?
- a. Une classe abstraite.
- b. Une classe qui hérite d'elle-même.
- c. Une classe ayant des méta-méthodes.
- d. Une classe dont les instances sont des classes.

Q.4 Les multiméthodes permettent
- a. aux méthodes de retourner plusieurs résultats.
- b. le polymorphisme dynamique sur plusieurs arguments de fonctions.
- c. à une classe d'avoir des méthodes portant le même nom.
- d. d'avoir des méthodes polymorphes (virtuelles) dans une hiérarchie de classe utilisant l'héritage multiple.

2 Programmation générique

Q.5 Les templates de classes du C++ sont

a. des collections de templates de fonctions libres.
b. des générateurs de classes.
c. des classes dont toutes les méthodes sont virtuelles.
d. des classes dont toutes les méthodes sont virtuelles pures.

Q.6 Parmi les termes suivants, lequel ne peut pas être utilisé comme paramètre effectif d'une classe paramétrée ?

a. `const`.
b. Une constante entière.
c. `unsigned`.
d. Un type classe défini par l'utilisateur.

Q.7 Parmi les lignes C++ suivantes, laquelle est invalide ?

a. `std::pair p1 (42, 51);`
b. `std::pair<float, int> p2 (42, 3.14f);`
c. `std::pair<int, float> p3 = std::make_pair (42, 3.14f);`
d. `std::pair<int, float> p4 = std::make_pair<int, float> (2.72f, 51);`

3 Programmation fonctionnelle

Q.8 On dit d'un langage qu'il est fonctionnel si. . .

a. il n'effectue aucun effet de bord.
b. il permet de manipuler des fonctions comme n'importe quel autre entité/objet.
c. il supporte le concept de fonction récursive.
d. il est Turing complet.
e. il dispose d'un compilateur implémenté et en état de marche.

Q.9 Un langage fonctionnel est dit pur lorsque

a. il proscrit tout effet de bord.
b. il ne contient aucune construction orientée objet.
c. ses fonctions ont au plus un argument.
d. ses expressions sont évaluées paresseusement.
e. la récursion est proscrite.

Q.10 On appelle fermeture

a. une fonction qui n'est pas récursive.
b. une fonction qui capture des références à des variables libres dans l'environnement lexical.
c. une fonction qui a été mise en ligne (*inlined*).
d. une fonction passée en argument à une autre fonction.

4 C++

Q.11 En C++, on appelle objet-fonction

a. un objet construit à l'intérieur d'une fonction.
b. un objet disposant d'un `operator()`.
c. une méthode.
d. un fichier de code compilé (‘`foo.o`’) ne contenant qu'une seule fonction (`foo()`).

Q.12 La liaison dynamique en C++

a. a un rapport avec `virtual`.
b. est liée à la surcharge des opérateurs.
c. repose sur `template`.
d. s'appuie sur `dynamic_cast`.

Q.13 Surcharge vs méthodes virtuelles : quelle est la bonne réponse ?

a. La surcharge et les méthodes virtuelles sont des mécanismes dynamiques.
b. La surcharge et les méthodes virtuelles sont des mécanismes statiques.
c. La surcharge est un mécanisme statique, les méthodes virtuelles un mécanisme dynamique.
d. La surcharge est un mécanisme dynamique, les méthodes virtuelles un mécanisme statique.

Q.14 Lequel de ces éléments n'entre pas en compte lors de la résolution d'une méthode surchargée en C++ ?

a. les arguments de l'appel.
b. l'arité de la fonction.
c. le nom de la fonction.
d. le qualificatif `const` de la méthode.
e. le type de retour.

5 Langages de programmation

Q.15 Qui est l'auteur du langage C ?

a. Brian Kernighan
b. Dennis Ritchie
c. Bjarne Stroustrup
d. K. N. King
e. Ken Thompson

Q.16 Quelle société est à l'origine des systèmes de fenêtrage, de la souris, de l'imprimante laser ?

a. Apple
b. Apollo
c. IBM
d. Microsoft
e. Xerox

Q.17 Lequel de ces langages n'a pas été influencé par Simula ?

a. Algol
b. C++
c. Objective C
d. Smalltalk
e. Eiffel

Q.18 Qui est l'inventeur de la souris ?

a. Walt Disney
b. Douglas Engelbart
c. Donald Knuth
d. Gordon Moore
e. Konrad Zuse

Q.19 Que signifie « BNF » ?

a. Backus-Naur Form
b. BASIC Numbering Formalism
c. Bison Normal Format
d. Bound Non-Finite (automaton)

6 Fonctions

Q.20 Le support des fonctions récursives nécessite

a. un tas (*heap*).
b. une pile (*stack*).
c. la liaison des fonctions dynamiques.
d. que le langage dispose de pré-déclarations (*forward declarations*).

Q.21 À la fin de ce programme, avec un *Mode* de passage des arguments par 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] := 6;
  t      := 2;
  x      := x + 3;
end;
```

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

a. `foo[1] = 6, foo[2] = 2, t = 2`
b. `foo[1] = 6, foo[2] = 4, t = 2`
c. `foo[1] = 9, foo[2] = 2, t = 2`
d. `foo[1] = 6, foo[2] = 5, t = 2`

Q.22 Même question, mais avec un *Mode* de passage d'arguments par référence.

a. `foo[1] = 6, foo[2] = 2, t = 2`
b. `foo[1] = 6, foo[2] = 4, t = 2`
c. `foo[1] = 9, foo[2] = 2, t = 2`
d. `foo[1] = 6, foo[2] = 5, t = 2`