

Vérifiez votre énoncé: les 4 entêtes doivent être +1/1/xx+... +1/4/xx+.

AppIng1 2016 - TYLA - 1h30 - S2 2014 *Sans document ni machine*

Noircir les cases plutôt que cocher. Renseigner les champs d'identité. Les questions marquées du symbole ♣ peuvent avoir plusieurs réponses justes. Toutes les autres questions n'ont qu'une seule réponse juste; si plusieurs réponses sont valides, sélectionner la plus restrictive (par exemple s'il est demandé si 0 est *nul*, *non nul*, *positif*, ou *négatif*, sélectionner *nul*). Il n'est pas possible de corriger une erreur. Les réponses justes créditent; les incorrectes pénalisent; et les blanches et réponses multiples valent 0.

Nom et prénom :

.....

.....

.....

.....

Cochez votre identifiant (de haut en bas):

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

1 Contrôle

Q.1 Avez-vous bien vérifié les en-têtes des 4 pages de ce sujet, comme indiqué en haut de cette première page ?

- Oui Non

2 Généralités

Q.2 Combien de nombres entiers différents peut-on coder avec 10 bits ?

- 1024 80 512 10

Q.3 À qui doit-on l'invention originelle d'Unix ?

- Richard Stallman Ken Thompson Brian Kernighan Bjarne Stroustrup

Q.4 Qui est le « N » de BNF ?

- Peter Norvig Peter Naur Lee Nackman John von Neumann

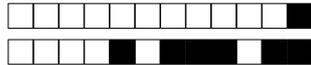
Q.5 Quel langage introduit le concept de fonctions imbriquées (avec portée statique) ?

- COBOL PL/I FORTRAN ALGOL 60

Q.6 Qu'est-ce qu'un *bytecode* ?

- Une variable signée ou non signée sur 8 bits Un trou dans une carte perforée
 Un encodage de caractères mono-octet, comme ASCII Un code compilé exécutable dans une machine virtuelle

Q.7 Parmi les expressions C++ ci-dessous, laquelle effectue une allocation sur le tas ?



- int p(51);
- int* p = new int[51];
- int p[51];
- int* p = (int*) alloca(51);

3 Fonctions

Q.8 Le support des fonctions récursives nécessite

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

Q.9 À la fin du programme ci-dessous, avec un *Mode* de passage des arguments par valeur (copie), quelles sont les valeurs de `foo[0]`, `foo[1]` et `t` ?

```
var t      : integer
    foo   : array [0..1] of integer;

procedure shoot_my(x : Mode integer);
begin
  foo[0] := 43;
  t      := 0;
  x      := x + 8;
end;
```

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

- `foo[0] = 43, foo[1] = 8, t = 0`
- `foo[0] = 8, foo[1] = 0, t = 0`
- `foo[0] = 51, foo[1] = 0, t = 0`
- `foo[0] = 43, foo[1] = 0, t = 0`

Q.10 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[0] = 8, foo[1] = 0, t = 0`
- `foo[0] = 51, foo[1] = 0, t = 0`
- `foo[0] = 43, foo[1] = 0, t = 0`
- `foo[0] = 43, foo[1] = 8, t = 0`

Q.11 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.

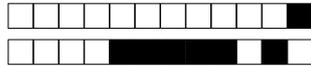
- `foo[0] = 43, foo[1] = 8, t = 0`
- `foo[0] = 51, foo[1] = 0, t = 0`
- `foo[0] = 43, foo[1] = 0, t = 0`
- `foo[0] = 8, foo[1] = 0, t = 0`

Q.12 Même question, mais avec un *Mode* de passage d'arguments par référence.

- `foo[0] = 43, foo[1] = 0, t = 0`
- `foo[0] = 51, foo[1] = 0, t = 0`
- `foo[0] = 43, foo[1] = 8, t = 0`
- `foo[0] = 8, foo[1] = 0, t = 0`

4 Programmation orientée objet

Q.13 En Smalltalk 76, comment instancie-t-on une classe ?



- En appelant son constructeur.
- Grâce à la primitive 'to'.
- En envoyant un message 'new' à Object.
- En envoyant un message 'new' à Class.

Q.14 Qu'appelle-t-on une métaclasse en Smalltalk 76 et 80 ?

- La classe dont dérivent toutes les classes, directement ou indirectement, explicitement ou implicitement.
- Une classe paramétrée.
- Une classe dont les instances sont des classes.
- Une classe qui n'a pas d'instance.

Q.15 Dans quel langage a été introduit la notion de *classe* ?

- COBOL
- Smalltalk
- ALGOL
- Simula

Q.16 Dans quel langage a été introduit la notion d'*objet* ?

- Simula
- COBOL
- Smalltalk
- ALGOL

Q.17 Le typage en Smalltalk est

- statique fort.
- dynamique.
- statique.
- inexistant.

Q.18 Templates vs méthodes virtuelles en C++ : quelle est la bonne réponse ?

- les instanciations de templates et les liaisons de méthodes virtuelles sont faites à la compilation.
- les instanciations de templates et les liaisons de méthodes virtuelles sont faites à l'exécution.
- les instanciations de templates sont faites à l'exécution tandis que les liaisons de méthodes virtuelles sont faites à la compilation.
- les instanciations de templates sont faites à la compilation tandis que les liaisons de méthodes virtuelles sont faites à l'exécution.

Q.19 Les multiméthodes permettent

- aux méthodes de retourner plusieurs résultats.
- le polymorphisme dynamique sur plusieurs arguments de fonctions.
- d'avoir des méthodes polymorphes (virtuelles) dans une hiérarchie de classe utilisant l'héritage multiple.
- à une classe d'avoir des méthodes portant le même nom (mais des arguments différents).

5 Programmation fonctionnelle

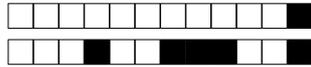
Q.20 On dit d'un langage qu'il est fonctionnel si...

- il n'effectue aucun effet de bord.
- il permet de manipuler des fonctions comme n'importe quel autre entité/objet.
- il est Turing complet.
- il supporte le concept de fonction récursive.

Q.21 Un langage fonctionnel est dit pur lorsque

- il proscrit tout effet de bord.
- ses fonctions ont au plus un argument.
- ses expressions sont évaluées paresseusement.
- il ne possède pas de construction orientée objet.

Q.22 En C++, on appelle objet-fonction



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

Q.23 Comment appelle-t-on une fonction qui capture des références à des variables libres dans l'environnement lexical ?

- Une lambda abstraction
- Une fonction d'ordre supérieur
- Une fermeture
- Une fonction récursive terminale

6 Programmation générique

Q.24 Quel langage ne dispose pas d'une fonctionnalité conçue pour contraindre les paramètres des types paramétrés ?

- C++ 2011
- Eiffel
- Ada
- Java

Q.25 Quelle différence y a-t-il entre macros et templates en C++ ?

- Les macros sont résolues à la compilation, les templates à l'exécution.
- Les macros ne supportent pas la récursion.
- Les macros sont résolues à l'exécution, les templates à la compilation.
- Les macros figurent dans le premier standard C++ (publié en 1998), mais pas les templates.

Fin de l'épreuve.