



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

## AppIng1 2016 - CMP1 - 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  
0 1 2 3 4 5 6 7 8 9

**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

**Q.2** Un compilateur large

- ne peut pas détecter les fonctions inutilisées.  
 compile les langages des *mainframes*.  
 fait passer toutes les étapes au programme entier les unes après les autres.  
 fait passer toute la chaîne de traitement à chaque ligne de programme l'une après l'autre.

**Q.3** Parmi les étapes suivantes, laquelle ne fait pas partie du *front end* (partie frontale) d'un compilateur ?

- La traduction  L'analyse lexicale  
 L'analyse de vivacité  L'analyse sémantique

**Q.4** Lex/Flex sont des

- scanners.  générateurs de parsers.  
 parsers.  générateurs de scanners.

**Q.5** Yacc/Bison sont des

- générateurs de parsers.  parsers.  
 scanners.  générateurs de scanners.

**Q.6** Dans un analyseur lexical et syntaxique, que signifie le terme « valeur sémantique » ?

- Un synonyme de « token »  
 Une information de localisation dans le fichier ou le flux d'entrée  
 Une information de typage calculée lors de l'analyse sémantique  
 Une information associée à un symbole produit par le scanner ou le parser



**Q.7** Que signifie le terme '@\$\$' dans une action sémantique de Bison associée à la règle de production 'exp : INT' ?

- Il s'agit de la localisation du symbole de la partie droite de la règle.
- Il s'agit de la valeur sémantique associée au symbole de la partie gauche de la règle.
- Il s'agit de la localisation du symbole de la partie gauche de la règle.
- Il s'agit de la valeur sémantique associée au symbole de la partie droite de la règle.

**Q.8** Comment désambiguiser pour Yacc/Bison le morceau d'arithmétique suivant :

exp: exp '+' exp | exp '-' exp | NUM;

- %left '+' %left '-'
- %left '-' %left '+'
- %left '+' '-'
- %left '+' %left '-' %nonassoc NUM

**Q.9** Comment désambiguiser pour Yacc/Bison le morceau d'arithmétique suivant :

exp: exp '\*' exp | exp '+' exp | NUM;

- %left '\*' %left '+'
- %left '+' '\*'
- %left '\*' %left '+' %nonassoc NUM
- %left '+' %left '\*'

**Q.10** Le rôle d'un parser est de

- segmenter un flux de caractères en un flux de tokens.
- faire de l'analyse syntaxique.
- s'assurer que les types sont bien utilisés.
- éliminer les récursions terminales.

**Q.11** Les « start conditions » de Lex/Flex (%s et %x) permettent

- de supporter différents contextes lexicaux.
- de déterminer quand l'analyse lexicale doit commencer.
- la conversion des chaînes de chiffres en la valeur qu'elles représentent.
- le choix du parser à utiliser.

**Q.12** Yacc repose sur l'algorithme

- YACC(1).
- LR(k).
- LALR(1).
- LL(k).

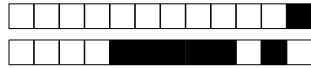
**Q.13** La syntaxe concrète est

- une représentation des programmes à partir d'objets et de classes.
- l'interface homme-machine d'un langage de programmation.
- une méthode de modélisation pragmatique.
- une grammaire de Backus en forme de Naur partagée.

**Q.14** Que signifie AST ?

- Arbre de syntaxe abstraite
- Arbre abstrait de synthèse
- Arbre de synthèse abstraite
- Arbre abstrait de syntaxe

**Q.15** ASN.1 est



- un langage de spécification fonctionnel.
- le premier outil de calcul électromécanique.
- un langage de programmation abstrait, simple, normalisé.
- une syntaxe pour décrire des paquets de données structurées.

**Q.16** Le patron de conception « Visitor » permet l'utilisation

- des multiméthodes dans un langage objet qui en est démunie.
- d'itérateurs en largeur d'abord.
- d'accesses sur des membres pourtant privés.
- d'itérateurs en profondeur d'abord.

**Q.17** Quelle vérification n'est pas effectuée lors de l'analyse sémantique ?

- La présence d'une construction de boucle autour d'une instruction `break`
- La présence de la déclaration d'une variable préalablement à son utilisation
- La conformité des noms utilisés comme identifiants
- La vérification des types des arguments lors d'un appel de fonction

**Q.18** Désucreur signifie

- retirer les commentaires, signes de ponctuation et retours à la ligne.
- reconnaître et corriger les erreurs de programmation typiques.
- traduire certaines constructions syntaxiques en une forme plus primitive.
- convertir une grammaire de SUGAR (SUGAR Unleashes Grammar Attribute Rules) à YACC (Yet Another Compiler Compiler).

**Q.19** Tiger est un langage dans lequel

- la portée est statique.
- les symboles des portées extérieures ne sont pas automatiquement importés dans les portées intérieures.
- la portée est dynamique.
- il y a une unique portée.

**Q.20** La résolution des appels « virtual » nécessite

- la connaissance du type des contenants.
- la connaissance du type des classes.
- la connaissance du type des contenus.
- la connaissance du type des opérateurs.

**Q.21** La résolution de la surcharge nécessite

- la connaissance du type des contenants.
- la connaissance du type des opérateurs.
- la connaissance du type des contenus.
- la connaissance du type des classes.

**Q.22** Lorsque l'on écrit une hiérarchie de classe et que l'on utilise le transtypage ascendant, il est recommandé, pour éviter les fuites mémoire

- d'écrire des constructeurs virtuels purs.
- d'écrire des destructeurs virtuels.
- d'écrire des destructeurs virtuels purs.
- d'écrire des constructeurs virtuels.

**Q.23** Utiliser des références à la place de pointeurs en C++ produit du code



- plus rapide.
- plus lent.
- équivalent en vitesse d'exécution.
- moins sûr.

**Q.24** Quel rôle ne jouent pas les langages intermédiaires ?

- Factorisation de certaines optimisations
- Décomposition en plusieurs étapes de la traduction
- Résolution de la surcharge
- Indépendance des parties frontales et terminales

**Q.25** Quelle réponse ne désigne pas un langage intermédiaire ?

- Bytecode Java
- Langage Tree
- Langage LLVM
- Langage MIPS

**Fin de l'épreuve.**