Examen - Janvier 2009

Durée : 2h

Les notes de cours, de travaux dirigés et de travaux pratiques sont autorisées. Le sujet comporte deux pages et trois parties (première partie : 5 points, deuxième partie : 10 points, troisième partie 5 points). Vous vous attacherez à soigner la présentation générale et l'orthographe, et vous serez notamment vigilant quant à la syntaxe lors de l'écriture de fragment de code.

Le sujet vous impose de programmer en utilisant des classes et des méthodes. Vous devez respecter cette contrainte !

1 Question de cours (5 pts)

Expliquer ce qu'est une méthode. Comment et à quel endroit dans le programme une telle opération est-elle déclarée ? Comment programme-t-on le corps de cette opération ? Comment procède-t-on pour faire un appel de méthode dans le `main` par exemple ? Afin de faciliter les explications, il est fortement recommandé d'illustrer sa réponse par un exemple d'utilisation concret.

2 Séquence de bits (10 pts)

On s'intéresse aux séquences non vides de bits : il s'agit simplement d'une liste non vide de bits 0 ou 1. On notera ces séquences entre crochets avec une virgule comme séparateur entre les bits, par exemple $[1, 0, 0, 0, 0, 0, 0, 0, 1, 0]$.

```
class bits
{
  int b;
  bits *suiv;

public:
  bits(bool bo);
  int longueur();
  void ajout(bool i);
  bool is_ok();
  void flip();
  friend ostream& operator<<(ostream &o, bits b);
};
```

Question 1 Expliquer comment les séquences $[1, 0, 0, 0, 1]$, $[1, 1]$ et $[0]$ peuvent être construites à l'aide des constructeurs disponibles.Concrètement, que faut-il écrire dans le `main` pour produire ces séquences à l'aide des constructeurs et méthodes de la classe `bits` ?
Question 1 bis La classe `bits` permet-elle de représenter la séquence vide ? Justifier votre réponse.
Question 2 Proposer une implantation du constructeur `bits` qui prend un booléen `true` ou `false` en argument et produit une séquence à un seul élément 0 ou 1.

Question 3 Proposer une implantation de la méthode `ajout` qui ajoute un entier 0 ou 1 à la fin de la séquence.
Question 4 Programmer une méthode `longueur` qui calcule la longueur d'un séquence.
Question 5 Programmer une méthode `is_ok` qui vérifie si toutes les valeurs d'une séquence ne sont que des 0 et des 1.
Question 6 Programmer une opération `flip` qui prend une séquence et inverse les bits, les 1 sont remplacés par des 0 et réciproquement. Cette opération sur les bits peut-elle être implantée de manière arithmétique (c'est-à-dire sans test sur la valeur du bit) ?
Question 7 Déclarer et implanter une nouvelle méthode `nb01` de la classe `bits` qui retourne le nombre de 0 et le nombre de 1 présents dans la séquence.
Question 8 Proposer une fonction d'affichage d'une séquence de bits. Expliquer les raisons pour lesquelles une telle fonction doit être déclarée amie (`friend`) de la classe `bits`. Justifier pourquoi l'arguments de type `bits` est passé en paramètre par valeur.
Question 9 Ajouter un constructeur à la classe `bits` qui prend en argument un entier non nul et le transforme en une séquence de bits.

3 Représentation numérique (5 pts)

On peut représenter les séquences de bits par des nombres. Dans ce cas, la séquence de bits représentée est l'écriture en binaire du nombre. Par exemple, la séquence $[1, 1, 1, 1]$ pourrait être représentée par le nombre 15.
Question 10 Proposer une nouvelle représentation sous forme d'entiers des séquences de bits. Il faudra bien sûr prendre en compte le fait que les séquences $[0, 0, 0, 1, 0, 0]$ et $[0, 0, 1, 0, 0]$ par exemple ne peuvent pas toutes les deux être représentées par le nombre 4. Comment faire pour les distinguer ?
Question 11 Quelles autres précautions serait-il nécessaire de prendre ? Proposer une classe `bits_number` permettant d'implanter les opérations demandées dans la partie 1 en utilisant la représentation numérique.
Question 12 Programmer les opérations de la classe `bits` dans cette nouvelle classe `bits_number`.