Interro pas très surprise

(Comme tous les mercredis)

NOM, prénom :

*Vous avez **13** minutes.*

Cours interdit. Calculatrices, téléphones et autres outils éléectroniques interdits.

Vous rédigerez sur cet énoncé ou dans son dos.

Exercice 1

On s'intéresse à la fonction ci-dessous. Vous n'avez pas besoin de comprendre pourquoi elle fonctionne pour répondre aux questions. On ignorerez les éventuels problèmes de débordement de capacité[i].

1. (2pt) Expliquez pourquoi dans la fonction ci-dessous k n'est pas un variant (ou alors c'est très dur de prouver que c'en est un).
2. (6pt) Prouver que `n` est un variant. *Vous pouvez utiliser sans preuve des propriétés sur la partie entière (tant qu'elles sont correctes).*
3. (2pt) En déduire la terminaison de la fonction.

exprap.c

```c
/** Calcule a puissance n par exponentiation rapide itérative.

    L'idée de l'algorithme est d'utiliser la décomposition de n en base 2.
    Par exemple, pour calculer a^5, on calcule a^{1+4} = a^1 * a^4.
 */
int exponentiation_rapide(int a, int n) {
  int sortie = 1;
  int k = 0;
  int a_2k = a; // stocke a^{2^k}
  while (n > 0) {
    if (n % 2 == 1) {
      // la décomposition de n a un chiffre c_k=1 de poids k
      sortie = sortie*a_2k;
    }
    a_2k = a_2k*a_2k;
    n = n/2;
    k = k+1;
  }
  return sortie;
}
```

i. Dans le monde merveilleux de la théorie, il n'y a pas de problèmes de débordements de capacité.

Solutions des exercices

Solution de l'Exercice 1

1. `k` est strictement croissant, mais n'est pas majoré (ou alors ce n'est vraiment pas évident du tout). Donc bien qu'il soit une variable qui augmente de 1 à chaque itération, il n'est pas un variant.
2. `n` est :
   - entier : d'après sa définition ligne 11 (`unsigned int n`).
   - minoré : d'après la condition du while ligne 15 (`n > 0`), `n` est strictement minoré par 0.
   - strictement décroissant : en notant `n'` la valeur de `n` en fin d'itération, on a (ligne 21) : `n'` = `n`/2, c'est à dire `n'` = $\lfloor \frac{n}{2} \rfloor$.
     Or, d'après la condition du while ligne 15, on sait qu'au début de n'importe quelle itération on a `n` > 0.
     Comme pour tout $n \in \mathbb{N}^*$, $\lfloor \frac{n}{2} \rfloor < n$, on a bien `n'` < `n` (on admet que).
3. Puisque la boucle admet un variant, elle termine. Or, il s'agit de la seule boucle et il n'y a pas d'appels de fonction : la fonction termine donc.