IF122 : Contrôle intermédiaire

Chaque exercice est noté sur 4. Il est donc fortement conseillé de ne faire que 5 exercices et de les faire bien que de faire 6 exercices et de les faire mal.
Les fonctions récursives doivent respecter les contraintes données en TD/TP.

Exercice 1 : `Vu en TD/TP...`
Écrire une fonction qui teste si un entier est divisible par un autre.

Exercice 2 : `Les variables (1)`
Identifier toutes les déclarations de variable et indiquer leurs caractéristiques :
- locale, globale, argument ;
- passage par valeur, par référence, ne s'applique pas à ce type de variable ;
- statique, non statique, ne s'applique pas à ce type de variable.

```
static int v = 0;
int w = 1;

int f(int v) {
  v=v+1;
  return v;
}
int g() {
  v++;
  return v-1;
}
int h(int &v) {
  v = v - w;
  return v;
}
int i(int w) {
  static int v = 10;
  w = v+1;
  return v;
}
int j(int v) {
  return f(v)+1;
}
```

Exercice 3 : `Les variables (2)`
Les fonctions précédentes sont utilisées dans le programme suivant. Donner l'affichage obtenu :

```
int main() {
  int w = 5;
  cout << v;
  cout << w;
  cout << f(v);
  cout << g();
  cout << i(v);
  cout << j(5);
  cout << h(v);
  cout << i(2);
}
```

Exercice 4 : `Les pointeurs`
Commenter TOUTES les lignes du programme suivant et identifier les éventuelles erreurs :

```
#include<iostream>
int main() {
  int *pe1 = new int[5];
  int **pe2 = new (int*)[6];

  pe2[0] = pe1;

  for(int i = 0; i < 6; i++) {
    cin >> pe1[i];
  }
  for(int i = 0; i < 6; i++) {
    cout << pe2[i][0];
  }
}
```

Exercice 5 : `Liste et récursivité (1)`
Le but de cet exercice est de gérer des listes triées (par ordre croissant). Nous allons donc écrire une nouvelle fonction de construction dont l'objectif est d'insérer un élément dans une liste triée de manière à obtenir une nouvelle liste triée. Nous appellerons cette fonction `cons_sorted`.

- Quelle est la liste triée la plus simple dans laquelle on peut ajouter un nouvel élément ?
- Dans quelle situation doit-on insérer l'élément en tête de la liste ?
- Quel est l'appel récursif à effectuer si l'élément ne doit pas être inséré en tête de la liste ?
- Écrire la fonction complète.

Exercice 6 : `Liste et récursivité (2)`
Le but de cet exercice est de déterminer si un élément se trouve dans une liste triée, et si c'est le cas, de renvoyer la sous-liste à partir de laquelle se trouve l'élément. Si l'élément n'est pas dans la liste, la liste vide est renvoyée. Exemple : search_sorted(4, (2,3,4,7)) nous renvoie (4,7).

- Quelle est la liste triée la plus simple pour laquelle on sait qu'on ne pourra pas trouver l'élément recherché ?
- La liste étant triée, à partir de quel moment sait-on que l'élément recherché ne pourra pas être trouvé dans la suite de la liste ?
- Dans quel cas a-t-on trouvé l'élément recherché ?
- Écrire la fonction complète.