Travaux pratiques

Gammes

1. **Caml** Écrire une fonction renvoyant la somme des éléments d'une liste de manière récursive et impérative.
2. **C** Écrire une fonction renvoyant le $n$ème nombre de fibonacci (de manière récursive naïve)
3. **Caml** Écrire une fonction qui transforme une liste en un vecteur, en inversant les données.
4. **C** Écrire dans un fichier `.h` les prototypes des fonctions nécessaires pour déterminer le $n$ème nombre premier. Écrire le fichier `.c` correspondant.
5. **Caml** Proposer un type pour étendre la division (par 0) et proposer une gestion d'erreur adéquate.
6. **C** Implémenter le tri à bulles.
7. **Caml** Créer un type pour des arbres binaires-ternaires (dont chaque noeud hormis les feuilles possède deux ou trois fils).
8. **C** Créer une structure de liste de couples.
9. **Caml** Écrire sans utiliser le mot-clef `let` ce que sont les fonctions incrémentation, addition, multiplication, inverse.
10. **C** Écrire une fonction qui 'imprime' un tableau bi-dimensionnel.
11. **Caml** Écrire une fonction qui ajoute à une liste les étiquettes des noeuds d'un arbre si elles satisfont une relation prise en paramètre (par exemple, si ces étiquettes sont paires, la fonction `fun x-> ...` est un paramètre d'appel de la fonction à écrire).
12. **C** Écrire une fonction déterminant si un arbre ne contient que des nombres pairs.
13. **Caml** Calculer les nombres binomiaux à l'aide d'un algorithme de programmation dynamique.
14. **C** ET **Caml** Implémenter les tris par insertion, sélection, fusion, Quicksort ; la création d'un arbre pour l'algorithme de compression de Huffman ; la multiplication de grands nombres par Karatsuba (chaque nombre est représentée par le tableau des chiffres qui le composent ; au contraire de la multiplication des polynômes, il faut prendre en compte la retenue) ; etc. (Les algorithmes vus en cours doivent être connus et reproduisibles)
15. **L'exercice le plus "efficace" sera la résolution d'un problème que vous vous posez à vous-mêmes et dont vous voulez absolument avoir la réponse.**