Colle 09

Cette neuvième colle vous fera écrire une fonction simple en OCaml sur des listes, puis une autre fonction en OCaml, et enfin un exercice en C sur des calculs variés.

Ex.1 Une petite fonction sur une liste (OCaml)

1. Écrivez en OCaml une fonction `fold_left` qui prend en entrée une fonction `f` de type `'a -> 'b -> 'a`, une valeur initiale `a` de type `'a`, une liste `li` (de type `'b list`), et applique dans l'ordre `f` à chaque élément de la liste, pour produire une nouvelle liste de la même longueur. Elle se comportera comme `List.fold_left` qui est de signature `('a -> 'b -> 'a) -> 'a -> 'b list -> 'a`, qui calcule `List.fold_left f a [b1; ...; bn]` = `f (... (f (f a b1) b2) ...) bn`. Votre fonction `fold_left` devra être récursive et ne pas utiliser `List.nth` mais un parcours récursif de la liste avec du *pattern matching* ;
2. Testez la sur plusieurs listes. Que va-t-il se passer sur une liste vide ? Expliquer.
3. Quel calcul va faire `fold_left ( + ) 0` sur une liste d'entiers ? Essayer sur au moins deux listes et expliquer.
4. Et `fold_left ( *. ) 1.0` sur une liste de flottants ? Essayer sur au moins deux listes et expliquer.

Ex.2 Une simple fonction OCaml : différences finies

5. L'opérateur des différences finies Δ associe à toute suite réelle $(u_n)_{n\in\mathbb{N}}$ la suite réelle $(u_{n+1} - u_n)_{n\in\mathbb{N}}$. Écrire une fonction `delta` qui réalise cette transformation. On commencera par bien réfléchir à son type. On se souviendra qu'une suite (à valeur entières) n'est rien d'autre qu'une fonction particulière.
6. Faire un exemple bien choisi.

Ex.3 Des petits calculs en C

Écrire une fichier C `colle09.exe` important `stdio.h`, `stdbool.h`.

On rappelle qu'on compile ce fichier avec `COMPILATEUR = gcc` ou `clang` avec la ligne de commande suivante, puis on exécute le binaire produit avec la dernière ligne :

```
$ COMPILATEUR -O0 -Wall -Wextra -Wvla -Werror -fsanitize=address \
              -fsanitize=undefined -o colle9.exe colle9.c
$ ./colle9.exe
```

Si une ligne affiche un résultat qui vous semble bizarre, commenter le.

1. Votre fichier contiendra une fonction `int main()` qui contiendra les tests (sous forme d'affichage avec `printf`) des questions suivantes ;

2. Nombre de chiffres d'un entier : écrire une fonction `int nombre_chiffres(int n)` qui donne le nombre de chiffres d'un entier naturel écrit en base 10. On considère que 0 a un chiffre ;
3. Ou exclusif : on rappelle que si `b1` et `b2` sont des `bool`, le est le "et" booléen binaire, le `||` est le "ou" booléen binaire, et enfin `! b` est le "non" booléen unaire. Afficher les tables de vérité de ces deux fonctions. Implémenter une fonction `bool ou_exclusif(bool b1, bool b2)` qui calcule le "ou exclusif" (qui vaut `true` si et seulement si exactement une seule des deux valeurs est à `true`). Afficher sa table de vérité. Pour les affichage, on pourra introduire une fonction `char* str_of_bool(bool b)` qui renvoie `"true"` ou `"false"` ;
4. (X 2010) Écrire une fonction `int log2(int n)` prenant en argument un entier $n \in \mathbb{N}^*$, et qui calcule le plus grand entier k tel que $2^k \leqslant n$.

