Université de Nice-Sophia Antipolis — mardi 26 octobre 2021
ELEC3 — Durée : 2 mois

Projet C : Grands Entiers Naturels

Avant de commencer : la qualité des commentaires, avec notamment la présence des antécédents, des conséquents, des invariants de boucle, les rôles de chacune des fonctions, ainsi que les noms donnés aux variables, l'emploi à bon escient des majuscules et la bonne indentation rentreront pour une part importante dans l'appréciation du travail. Ce projet doit permettre de montrer votre autonomie et votre compréhension tant dans la conception du programme que dans sa réalisation. Enfin, si les codes de plusieurs projets se trouvent être identiques, ou être copiés depuis le web, tous les projets concernés seront immédiatement sanctionnés par un zéro.

1 Les *BigNat*

L'objectif de ce projet est de définir et de manipuler des grands entiers naturels.

En C, le plus grand entier que l'on peut manipuler est de type `unsigned long long int`, représenté, en général, sur 8 octets sur les PC actuels.

Pour manipuler des entiers naturels (*positifs ou nuls*) de taille quelconque, en particulier plus grands que les entiers max des types etiers prédéfinis, vous allez définir le type `BigNat`. Les chiffres des entiers naturels `BigNat`, par défaut en base 10, seront conservés dans un tableau de caractères.

2 Fonctions sur les *BigNat*

Vous programmerez un ensemble de fonctions qui permettront la manipulation des *BigNat*. Une liste non exhaustive (d'autres opérations pourront être définies si nécessaires) est donnée ci-dessous :

- Initialisation ;
- Lecture et écriture ;
- Addition ;
- Soustraction ;
- Multiplication ;
- Division ;
- Modulo ;
- Comparaison ($<$, $\leqslant$, $=$, $\neq$, $>$, $\geqslant$)

Les fonctions d'initialisation permettront l'initialisation :

- à 0 d'un *BigNat* ;
- d'un *BigNat* à partir d'une chaîne de caractères ;
- d'un *BigNat* à partir d'un entier naturel ;
- d'un *BigNat* à partir d'un autre *BigNat*.

La fonction *lire* lit sur l'entrée standard un *BigNat*. Les différentes syntaxes qui définissent un entier naturel en C devront être acceptées.

La fonction *écrire* écrit sur la sortie standard un *BigNat*. On pourra indiquer un format (décimal, octal ou hexadécimal et cadrage à gauche ou à droite).

Les fonctions *additionner*, *soustraire*, *multiplier*, *diviser*, et *modulo* implémentent respectivement l'addition, la soustraction, la multiplication, la division et le reste de la division de deux *BigNat*.

Les fonctions de comparaisons renveront une valeur booléenne.

3 Structure de données

Toutes les fonctions précédentes manipulent des *BigNat* et vous serez amenés à définir un type C *BigNat* pour les représenter. Vous pourrez définir le type *BigNat* par la déclaration :

```
typedef ????? BigNat;
```

Une variable `x` de type *BigNat* sera simplement déclarée comme suit :

```
BigNat x;
```

4 Utilisation des *BigNat*

Vous écrirez ensuite la fonction *factorielle* que vous utiliserez pour calculer 100!

5 Remise du projet

Votre projet est à rendre au plus tard :

Vendredi 7 janvier 2022 à 22h – aucun délai ne sera accordé –

sous forme d'une archive `BigNat.tar.gz` que vous m'enverrez pas courriel.

5.1 Cette archive devra contenir :

- le fichier source (`.c`) correctement documenté (chaque fonction doit avoir un commentaire, les invariants de boucle doivent être marqués), indenté, et codé (les noms de variables explicites, éviter les trop longues fonctions) ;
- un fichier `Documentation` au format pdf et décrivant le fonctionnement général du programme, les algorithmes, ainsi que les choix de programmation ;
- la compilation avec les options `-Wall -pedantic` ne doit pas donner de *warning*.