# Spécifications fonctionnelles et techniques

## I. Introduction :

### Objet du document :

Ce document permet d'aborder les aspects fonctionnels et techniques de notre projet qui consiste en la réalisation d'une calculatrice à notation polonaise inversée avec des nombres entiers très grands, (notamment utilisés pour la cryptographie).

### Objectif et cible :

Cette calculatrice devra être capable de manipuler des nombres entiers dont le nombre de chiffres est supérieur aux capacités de représentation des entiers 32 ou 64 bits supportés par les types classiques et donc saisir des grands nombres à travers le clavier, indiquer si l'entrée saisie est invalide et lui permettre de recommencer cette dernière, effectuer les opérations arithmétiques classiques (l'addition, la soustraction, la multiplication, la division, le modulo, la puissance et la comparaison) puis afficher le résultat.

## II. Organisation des modules :

### Gestion de l'entrée/sortie :

- **Fonctionnement des grands nombres**

Le projet est composé de plusieurs modules afin de répondre à la demande du client, nous avons mis en place une classe définissant le concept des grands nombres, elle se nomme GN (Grand nombre) elle est composé de plusieurs parties :
- un booléen qui indique le signe du nombre
- un vecteur de caractères qui contient les différents chiffres composant le grand nombre.

- **La conversion string → vecteur**

Description : cette fonction permettra la conversion de la chaîne de caractère rentrée par l'utilisateur en un vecteur de valeurs après une conversion via la table d'ascii.

Prototype : void Conversion(string S, GN &nb)

Explication : Cette fonction sera utilisé deux fois successivement pour chaque nombre utilisé dans l'opération, il ne faut pas oublier le passage par référence au vecteur contenant les différents chiffres composant le grand nombre.

- **L'affichage**

Description : cette fonction permettra la visualisation du résultat de l'opération réalisée par l'utilisateur.

Prototype : void Affichage(GN result)

## Gestion de l'Application :

**-Fonctionnement de la calculatrice/menu :**

L'utilisateur choisit deux nombres à opérer (**a** et **b**), il choisit ensuite l'opération à appliquer :

→ touche « **+** » pour l'addition
→ touche « **-** » pour la soustraction
→ touche « ***** » pour la multiplication
→ touche « **/** » pour la division
→ touche « **%** » pour le modulo
→ touche « **^** » pour la puissance
→ touche « **c** » pour la comparaison

Si il souhaite commencer une nouvelle opération en utilisant le résultat de l'opération précédente, il utilise la touche « **r** » : le résultat devient **a** et l'utilisateur peut choisir **b** et la nouvelle opération.
Pour une nouvelle opération, il utilise la touche « **n** ».
Si il souhaite sortir de la calculatrice, il utilise la touche « **s** ».

## Gestion des opérations :

- **Addition**

Description : cette fonction permettra la gestion de l'opération d'addition des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Le résultat est stocké dans un troisième grand nombre nommé result.

Prototype : void Addition(GN nb1, GN nb2)

**- Soustraction**

Description : cette fonction permettra la gestion de l'opération de soustraction des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Le résultat est stocké dans un troisième grand nombre nommé result.

Prototype : void Soustraction (GN nb1, GN nb2)

**- Multiplication**

Description : cette fonction permettra la gestion de l'opération de multiplication des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Le résultat est stocké dans un troisième grand nombre nommé result.

Prototype : void Multiplication (GN nb1, GN nb2)

**- Division**

Description : cette fonction permettra la gestion de l'opération de division des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Le résultat est stocké dans un troisième grand nombre nommé result.

Prototype : void Division (GN nb1, GN nb2)

**- Modulo**

Description : cette fonction permettra la gestion de l'opération de modulo des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Le résultat est stocké dans un troisième grand nombre nommé result, celui ci étant le reste de la division entre nb1 et nb2.

Prototype : void Modulo (GN nb1, GN nb2)

**- Puissance**

Description : cette fonction permettra la gestion de l'opération de puissance des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Le résultat est stocké dans un troisième grand nombre nommé result, celui ci étant le résultat de nb1 puissance de nb2.

Prototype : void Puissance (GN nb1, GN nb2)

**- Comparaison**

Description : cette fonction permettra la gestion de la comparaison des deux grands nombres entrés par l'utilisateur, nb1 et nb2. Cette fonction permettra de déterminer si nb1 est supérieur,inférieur ou égale à nb2 à travers l'affichage de cette réponse.

Prototype : void Comparaison (GN nb1, GN nb2)

## III. Normes de Codage

- **Nomination des fichiers**

→ fichier principal : main.cpp
→ fichier librairie : librairie.h
→ fichiers des fonctions d'opération : un fichier par fonction d'opération ayant pour nom Opération.h et Opération.cpp

- **Commentaires**

→ explications avant chaque fonction entre « /* » et « */ ».
→ explication complémentaires (pour des variables) sur une même ligne derrière « // ».
→ le passage aux commentaires dioxygen sera réalisé par le coordinateur de projet.

**- Indentation**

→ une indentation se fera pour chaque bloc fils (compris entre les balises « { » et « } » ).
→ deux blocs de même niveau débutent sur la même colonne (exemple: deux blocs de condition « if »).

- **Taille des lignes**

→ une ligne ne peut contenir plusieurs instructions si celles-ci est courte.
→ si la longueur d'une ligne dépasse 100 caractères il est nécessaire d'effectuer un retour à la ligne.

- **Nomination du code**

→ les variables d'itérations seront de préférence « i », « j »et « k ».
→ les noms des fonctions reflètent leur utilité, les espaces sont remplacés par des « _ ».
→ un nom de variable ne doit pas se retrouver plusieurs fois dans le code même lorsqu'il n'est pas entre le même groupe d'accolades (sauf pour les variables d'itération).
→ les variables au nom peu explicite doivent être commenté afin de décrire leur

rôle.

- **Déclarations**

→ plusieurs variables peuvent être déclarées sur une même ligne si elle sont de même type.
→ l'initialisation doit se faire, dès que possible, de préférence lors des déclarations.
→ les déclarations doivent se faire dès le début du code, après l'accolade ouvrante (au début
de la fonction).

## IV. Outils

**- Éditeur de Texte**

→ L'éditeur de texte choisie est GeditGedit est l'éditeur de texte officiel de l'environnement graphique GNOME . Ce logiciel, sous licence GPL, a été retenue pour son interface simple et sa facilité d'utilisation.
→ Il dispose de modes de surbrillance et coloration configurables pour de nombreux langages de programmation, d'un outil de recherche et de remplacements, de la numérotation des lignes, de l'indentation automatique ainsi que d'autres fonctionnalité utiles pour la réalisation de ce projet.

- **Compilateur**

→ Le compilateur choisie est GCC, le compilateur du projet GNU,capable de compiler divers langages de programmation, dont celui utilisé pour la réalisation du projet le C++.

- **Debugger**

→ Le débugger choisie est GDB.Son intégration au compilateur GCC et donc leur utilisation conjointe permet de faciliter les opérations de débuggage.

- **Documentation**

→ La documentation et l'explication du code des différents modules serra réalisées grâce au logiciel DOXYGEN afin de mettre en place la mise en forme d'une documentation à l'intention des différents développeurs.

- **Profiler**

→ Le profiler choisie est Gprof.Gprof est un logiciel GNU accessible en ligne de commande.
→ Son mode d'utilisation, de concert avec gcc, permet une utilisation simple et rapide.

- **Forge**

→ La forge choisie pour la réalisation de ce projet est GoogleCode.

**- Équipe**

Application - FLAUS Mathias
Saisie et Affichage - GUILHOT Robin/MACCARIO Adrien
Opération - FLAUS Mathias/GUILHOT Robin/RODRIGUEZ Marie
Intégration et coordination - MACCARIO Adrien