Projet Maths-Info CIR3-CNB3

Objectif

Le but de ce projet est d'aboutir à l'application de résultats mathématiques au travers d'une réalisation en langage Java (pour les CIR3) ou en langage C (pour les CNB3). Chaque type de réalisation devra s'employer à utiliser les bonnes pratiques de programmation du langage, ainsi qu'utiliser au mieux les spécificités architecturales de ce langage.

Il vous sera demandé de montrer :

- vos capacités d'auto-gestion de votre travail, au travers des documents créés (pertinence et sérieux de la réalisation) ;
- votre compréhension des concepts mathématiques, au travers des documents créés (explications données) et des résultats obtenus (critique constructive des résultats obtenus) ;
- votre maîtrise du développement logiciel, au travers des documents créés (description à plus haut niveau d'abstraction que le code tapé) et du code source écrit (qualité, lisibilité, réutilisabilité, etc.)

Modalités générales

Le projet se déroule sur la semaine du lundi 18 décembre 2017 au vendredi 22 décembre 2017. Il se termine par une soutenance de 15mn par groupe (présentation de 8 à 10mn, + questions) le vendredi à partir de 13h30.

Les groupes seront constitués de binômes, choisis de manière libre par les participants. La liste des binômes devra être figée et envoyée à vos enseignants lundi à 16h.

Le projet fera obligatoirement l'objet d'un premier rendu, comportant :

- cahier de conception (informations théoriques et pratiques) ;
- cahier de test (comment le code a-t-il été validé, ainsi que les résultats des tests) ;
- sources du projet permettant de reconstruire intégralement le programme (avec éventuellement le ou les programmes de test).

Un second rendu (voir ci-dessous) pourra être demandé en fonction de l'avancement du binôme, et conduire bien évidemment à une meilleure note.

La soutenance réalisée en langue anglaise (prestation orale + diapositives) donnera lieu à une bonification.

Enseignants référents :

frederic.fressel@isen.fr

ghislain.oudinet@yncrea.fr

lidiya.yushchenko@yncrea.fr

georges.thelot@isen.fr

Sujet

Le projet se déroulera en plusieurs parties.

Dans un premier temps, il vous sera demandé :

- d'implémenter en langage Java (le cas échéant) un objet qui sera construit à l'aide d'un nombre représentant une taille arbitraire (mais sous la forme d'une puissance de 2) passé en paramètre ;
- d'implémenter dans cet objet une méthode d'instance, ou implémenter en langage C (le cas échéant) une fonction, permettant de calculer la transformée de Fourier rapide (FFT) d'un tableau de nombres réels en virgule flottante simple précision, selon l'algorithme qui vous sera proposé ci-après ;
- de tester le bon fonctionnement de ce code ;
- d'implémenter dans cet objet Java une méthode d'instance, ou ajouter une fonction en langage C, permettant de calculer la transformée de Fourier rapide d'un tableau de nombres complexes ;
- de tester le bon fonctionnement de ce code ;
- d'implémenter dans cet objet une méthode d'instance permettant de calculer la transformée de Fourier rapide inverse (iFFT) d'un tableau de nombres complexes (pour ce faire, vous ajouterez dans votre document de conception votre recherche d'une technique, potentiellement basée sur le conjugué de la FFT du conjugué, pour aboutir à ce résultat) ;
- de tester le bon fonctionnement de ce code ;
- d'ajouter dans votre document de conception une justification de la méthodologie de test employée (il est pour cela conseillé de travailler sur un petit tableau contenant des données simples, comme une période de sinusoïde, afin de vérifier si la FFT puis la iFFT se comportent comme attendu), ainsi qu'une justification des données obtenues en sortie de la FFT et de l'iFFT en fonction du signal en entrée.

Une fois qu'un binôme pourra présenter à la fois le document de conception, les codes source et exécutable d'un programme réalisant tout ceci ainsi que le cahier de tests associé, la seconde partie du projet lui sera révélée.

Modalités de notation

La notation prendra en compte à la fois la forme et le fond, mais une attention toute particulière sera portée sur le professionnalisme de la réalisation ("satisfaction du client", qui sera incarné par vos enseignants).

Le fait d'implémenter la FFT dans un objet Java indépendant permettra de mettre en œuvre, si vous en avez le temps, des optimisations afin de réduire le temps de calcul.

L'ajout d'une interface graphique (par exemple ISENTLIB en C) à votre programme sera aussi appréciée, afin de rendre votre programme plus agréable à utiliser, mais une interface texte en ligne de commande bien pensée sera aussi valorisée.

La première partie imposée sera notée sur 15 points. La partie suivante, révélée au fur et à mesure de la progression du binôme, sera notée sur 5 points.

Des pénalités en points pourront être attribuées individuellement (ex. : retards répétés, travail notablement inférieur, etc). De même, des points de bonus pourront être attribués individuellement ou à un binôme (ex. : leadership net et efficace d'un membre du binôme, réalisation exceptionnelle, etc.)

Annexe 1 : algorithme de FFT proposé

L'algorithme de calcul de FFT écrit en Java devra être implémenté en s'appuyant sur l'algorithme ci-après (Cooley-Tuckey), fourni en langage naturel.

Pour calculer la FFT d'un signal dont la taille L est une puissance de 2 :

- créer un tableau de complexes de taille L, tableau qui sera nommé S ;
- si la taille du signal est unitaire :
  - alors copier le signal dans le tableau S ;
- sinon :
  - calculer la FFT, nommée P0, des éléments pairs du signal,
  - calculer la FFT, nommée P1, des éléments impairs du signal,
  - pour un indice k allant de 0 à L/2-1 :
    - calculer un multiplicateur complexe nommé M, de module unitaire et d'argument égal à -2.*π*k/L,
    - fixer S[k] à P0[k]+P1[k]*M,
    - fixer S[k+L/2] à P0[k]-P1[k]*M,
- renvoyer S.

Annexe 2 : nombres complexes en Java

Le langage Java ne supportant pas nativement les nombres complexes, vous aurez éventuellement à :

- soit trouver et utiliser une bibliothèque ajoutant ces capacités au langage ;
- soit implémenter par vous-même un objet NombreComplexe qui vous facilitera la vie, et qui comportera au moins les méthodes suivantes (liste non exhaustive) :
  - initialisation à partir de partie réelle et partie imaginaire,
  - récupération de la partie réelle et de la partie imaginaire,
  - addition,
  - soustraction,
  - multiplication.

La classe Math contient la constante PI, qui vous sera utile.

Annexe 3 : nombres complexes en C

Le langage C supporte depuis la norme C99 les nombres complexes, par l'intermédiaire d'un support offert par le compilateur ainsi que du fichier d'en-tête <*complex.h*>.

Un nombre complexe en langage C se déclare à l'aide du mot-clé complex, et le symbole i suivant une constante numérique impose cette constante comme imaginaire :

complex float valeur = 1+1i;

Les parties réelle et imaginaire se récupèrent respectivement grâce aux fonctions *creal()* et c*imag()*.

L'addition, la soustraction et la multiplication sont gérées par les opérateurs classiques du C.

L'inclusion de <*math.h*> permet d'utiliser la constante M_PI, qui vous sera utile.