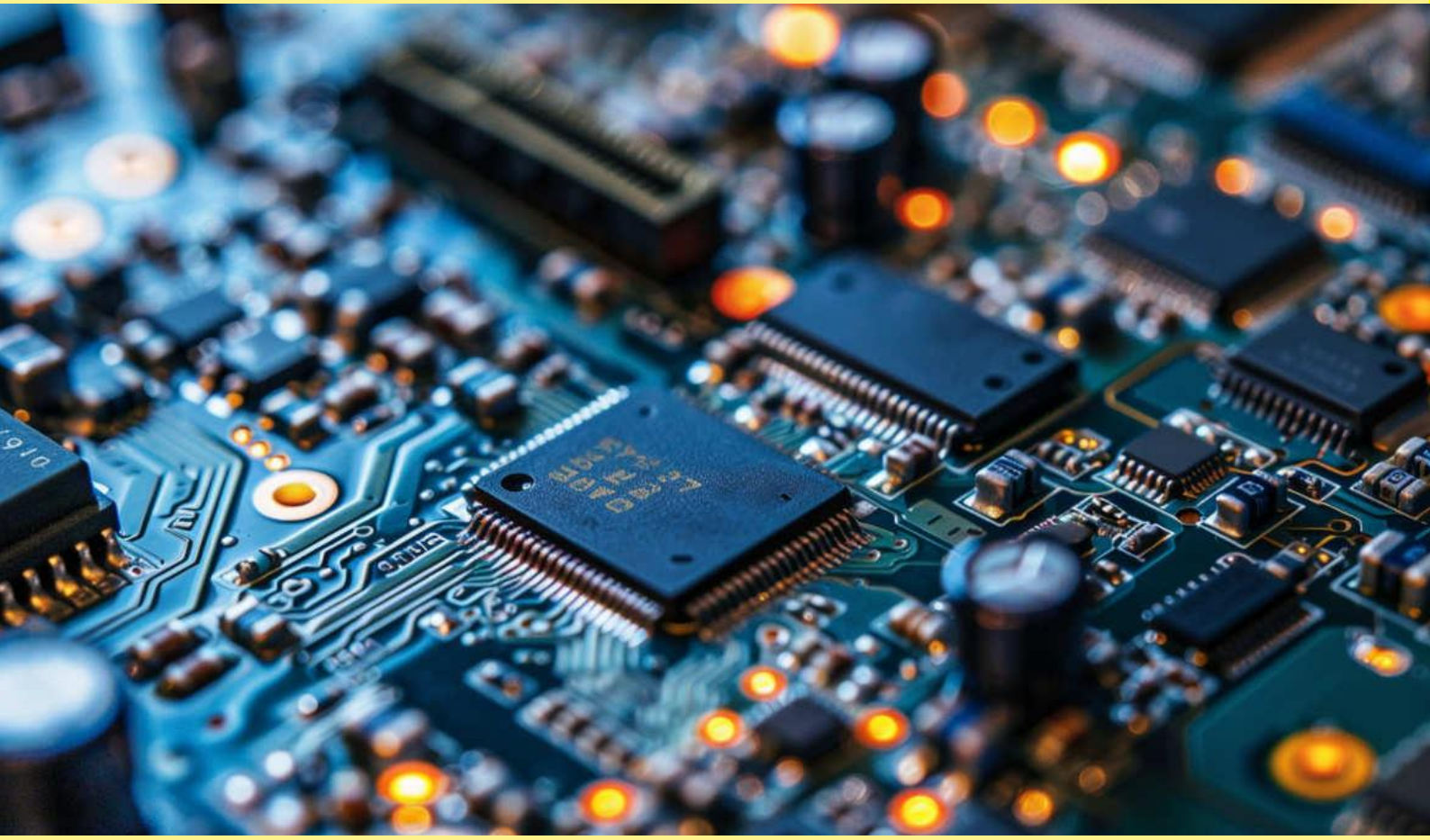
CPI A2 – PROJET POO

PROSIT 5 – CONCEVOIR SON APPROCHE



CESI
ÉCOLE D'INGÉNIEURS

CONTEXTE :

Claude Delannoy est un célèbre auteur de livres éducatifs portés sur les langages informatiques et la programmation orientée objet. Dans un nouvel ouvrage, il souhaite donner les clés au lecteur pour bien architecturer son code C++. Ses élucubrations le mènent vers la représentation et la manipulation des polynômes. Il tient en effet à illustrer son propos au travers d'un cas pratique. Pour rédiger son chapitre, il entreprend alors une expérience de pensée complexe.

INFORMATIONS IMPORTANTES :

MOTS INCONNUS :

Template (dans le code GitHub) :
https://openclassrooms.com/fr/courses/7137751-programmez-en-oriente-objet-avec-c/7533236-creez-des-templates Permet a une fonction ou classe de se lancé quelque soit le type de son argument, évite beaucoup de répétitions car au lieu de déclarer une fonction pour les char et une pour les Int qui fait la même chose on utilise Template sur une seule et même fonction.

Namespace et collisions : Un namespace, ou espace de nom (parfois aussi espace de nommage, voire référentiel lexical) est une zone de déclaration d'identificateurs permettant au compilateur de résoudre les conflits de noms.
Si, par exemple, deux développeurs définissent des fonctions avec le même nom, il y aura un conflit lors de l'utilisation de ces fonctions dans un programme. Les espaces de nommage permettent de résoudre ce problème en ajoutant un niveau supplémentaire aux identificateurs.

PROBLEMATIQUE :

Comment pouvons-nous comprendre puis réorganiser le code donné afin d'éviter les collisions de noms ?

PLAN D'ACTION :

- Reformuler et réinterpréter les problèmes issus de notre réflexion.
- Décortiquer le code dont un ersatz est disponible sur GIT à l'adresse ci-dessous.
- Identifier et documenter les éléments d'implémentation liés aux problématiques soulevées (PB 1, PB 2, PB 3, PB 4).

- Organiser la source de sorte à faire emploi des namespaces pour arranger notre code sémantiquement et éviter les collisions de noms.

REALISATION :

PB1 : les entités qu'on a besoin sont, monôme et polynôme

PB2 : Non un polynôme est un regroupement de plusieurs monômes

PB3 : Non un monôme est une part du polynôme

PB4 : Rassembler les déclarations et définitions des classes dans un seul fichier peut simplifier les petites implémentations.

Pour des projets plus complexes :
Utiliser un fichier d'en-tête (.h) et un fichier source (.cpp).
Organiser le code avec des `namespace` pour éviter les collisions de noms et structurer logiquement les classes.