École Centrale de Nantes

Algorithmique et Programmation

TD6

Compétences :
— Écriture de code `C++`
Notions :
— Pointeurs
— Références
— Fonctions
— Structures de données

6.1 Passage de paramètres dans les fonctions

Voir les paragraphes correspondant dans le cours de programmation.

À partir des exemples du cours, l'objectif est d'étudier les différentes manières d'échanger des informations entre une fonction appelante et une fonction appelée. Vous donnerez des noms de variables différents entre la fonction appelante et la fonction appelée. Vous testerez toutes les fonctions.

NB : Concernant les tableaux, les exemples ci-dessous et dans le code sont données avec des tableaux statiques. Il est conseillé de tester aussi les tableaux dynamiques (par exemple `vector<t_article>`). Attention aux modalités de passage de paramètres de `vector` qui sont identiques aux paramètres standards.

— Créer votre projet et écrire une fonction principale pour tester les appels de fonction.
— Écrire des structures de données pour stocker des valeurs : tableaux, enregistrement, tableaux d'enregistrements, etc. (par exemple le type `Article` du cours).
— Créer une fonction sans paramètres qui renvoie un résultat, par exemple : `Article lectureArticle()` pour lire un article au clavier,
— Créer une fonction avec paramètres qui renvoie un résultat, par exemple : `Article lectureArticle(int reference, float prix, string nom)` qui produit un `Article` à partir d'éléments passés en paramètres,
— Créer une fonction avec un tableau en paramètre qui renvoie un résultat, par exemple : `float sommePrixArticles(VectArticles articles)` qui calcule le prix total des articles,
— Créer une fonction avec un tableau en paramètre qui renvoie plusieurs résultats, avec des références, par exemple : `void prixMinMax(int nbArticles, VectArticles articles, float & pMin, float & pMax)` qui calcule les prix min et max d'un tableau d'articles,
— Créer la même fonction avec des pointeurs à la place des références,
— Créer une fonction qui modifie un paramètre d'entrée (hors tableau), par exemple : `void miseAJourPrix(Article & article, float nouveauPrix)` qui change le prix d'un article,
— Créer une fonction qui modifie un tableau d'entrée, par exemple : `void miseAJourTableau(VectArticles articles, Article nouvelArticle, int indice)` qui ajoute un nouvel article dans un tableau,
— Créer une fonction qui lit les valeurs et les stocke dans un tableau, par exemple : `void lectureTableau(VectArticles articles)` qui lit les articles dans un fichier,
— *bonus* Créer une fonction avec un tableau en paramètre qui renvoie plusieurs résultats, avec des pointeurs.

6.2 Structure de données : Listes

Reprenez le TD sur les listes (TD4).

— Créer la structure de donnée `elementListe` pour stocker un élément d'une liste d'entier,
— Écrivez une fonction pour ajouter un élément au début d'une liste,
— Écrivez une fonction pour faire l'affichage *itératif* d'une liste,
— Écrivez une fonction pour faire l'affichage *récursif* d'une liste,
— *bonus* Reprenez les autres questions du TD4.

6.3 Structure de données : Arbres

Reprenez le TD sur les arbres (TD5).

— Créer la structure de donnée `nœud` pour stocker un nœud d'un arbre d'entier,
— Écrivez une fonction pour faire l'affichage *préfixe* d'un arbre,
— Écrivez une fonction pour faire l'affichage *infixe* d'un arbre,
— Écrivez une fonction pour faire l'affichage *postfixe* d'un arbre,
— Écrivez une fonction pour calculer du nombre de nœuds d'un arbre,
— Écrivez une fonction pour calculer la hauteur d'un arbre,
— *bonus* Reprenez les autres questions du TD5.