TD – Tableau dynamique
ING1 – Programmation C
Génie Informatique

Préambule

L'ensemble du TP formera un seul programme. Les différentes fonctionnalités traitées devront être accessible via un menu. Je vous conseille de bien séparer chacune des fonctions, ainsi que celle du menu, de la saisie, d'affichage ... Bien découper votre code vous facilitera la lisibilité du code, et vous évitera un certain nombre d'erreurs. Vous travaillerez sur des tableaux d'entiers de taille définie par l'utilisateur en début de programme.

1 Tris

1.1 Tri par insertion

(1) Écrire la procédure qui permet de trier un tableau d'entier en utilisant la méthode du tri insertion. □

1.2 Tri fusion

Les étapes de la méthode du tri fusion sont les suivantes :

1. Si le tableau n'a qu'un élément, il est déjà trié.
2. Sinon, séparer le tableau en deux parties à peu près égales.
3. Trier récursivement les deux parties avec l'algorithme du tri fusion.
4. Fusionner les deux tableaux triés en un seul tableau trié.

(2) Écrire la fonction `int* copierSousTableau(int* src, int debut, int fin)` qui permet de copier dans un tableau *dest* les valeurs du tableau *src* allant de l'indice *debut* à l'indice *fin*. □

(3) Écrire la procédure `void fusion(int* tab1, int taille1, int* tab2, int taille2, int* tabRes)` qui permet de fusionner deux tableaux triés de façon croissante *tab1* et *tab2* dans un tableau résultat *tabRes* qui sera lui aussi trié de façon croissante. □

(4) Écrire la procédure `void triFusion(int* tab, int taille)` qui trie un tableau de façon croissante. Cette procédure suivra les étapes du tri fusion décrites précédemment. □

1.3 Tri par dénombrement

Le tri par dénombrement est un tri linéaire qui ne s'applique que sur des valeurs entières, avec de préférence un domaine de définition pas trop grand. Il ne nécessite aucune comparaison entre les éléments.

Son principe repose sur la construction de l'histogramme des données, puis le balayage de celui-ci de façon croissante, afin de reconstruire les données triées. Les étapes sont les suivantes :

1. Rechercher dans le tableau les valeurs minimum et maximum ;
2. Créer un tableau *histogramme* de taille $(max - min + 1)$ dont toutes les valeurs sont initialisées à 0 ;
3. Parcourir le tableau d'origine, et pour chaque valeur x du tableau, incrémenter la case $x - min$ du tableau *histogramme* ;
4. Parcourir le tableau *histogramme* et recomposer le tableau d'origine en créant autant de valeur que défini dans l'histogramme.

(5) Écrire la procédure `void minMaxTableau(int* tab, int taille, int* min, int* max)` qui recherche les valeurs minimum et maximum du tableau *tab*. □

(6) Écrire la procédure `void histogramme(int* tab, int taille, int* histo, int tailleH, int min)` qui permet de déterminer la fréquence d'apparition de chaque élément du tableau *tab*. □

(7) Écrire la procédure `void triDenombrement(int* tab, int taille)` qui trie un tableau de façon croissante. Cette procédure suivra les étapes du tri par dénombrement décrites précédemment. □