IF2B – Examen final
Durée : 1h30

Seule une feuille A4 manuscrite nominative recto-verso est autorisée comme document.

Tous les autres documents ou tout autre dispositif électronique sont interdits.

Barème donné à titre indicatif (±1)

Exercice 1 – Tri à peigne (7 points)

Le tri à peigne est un algorithme de tri inspiré du tri à bulles mais dont la performance est supérieure dans certains cas.

Le principe de l'algorithme est de permuter les éléments deux à deux si le premier est supérieur au deuxième. La différence avec le tri à bulles vient du fait que les deux éléments ne sont pas forcément successifs, mais sont espacés d'un intervalle I. A chaque itération sur l'ensemble du tableau, cet intervalle I est divisé par 1,3 (sans jamais pouvoir être inférieur à 1). L'intervalle initial I est égal à la longueur du tableau moins 1.

1) Ecrire une fonction « peigne » qui prend en paramètre un tableau T, la taille N de ce tableau, et un intervalle I → Pour chaque élément en position P du tableau, si l'élément en position P + I est inférieur, on permute l'élément en position P et l'élément en position P + I (on arrêtera le parcours de manière à ne pas évaluer d'éléments en dehors du tableau). Cette fonction retournera le nombre de permutations effectuées.
2) Ecrire une fonction « tri_peigne » prend en paramètre un tableau T et la taille N de ce tableau → Cette fonction appellera la fonction « peigne » jusqu'à ce que le tableau T soit trié, la valeur initiale de l'intervalle I sera N-1, puis I sera divisé par 1,3 à chaque itération (sans jamais pouvoir être inférieure à 1).

Exercice 2 – Émulateur et langage assembleur (13 points)

Le langage assembleur ou ASM est un langage de programmation très simpliste et très bas niveau (proche du langage machine). Il est défini par une liste de mots-clés qui permet d'indiquer des opérations élémentaires que le processeur d'un ordinateur doit réaliser.

Par exemple :

- **MOV AH,0** → mets la valeur 0 dans le registre AH (on peut considérer un registre comme une variable pré-définie)
- **ADD AH,10** → Ajoute la valeur 10 à la valeur préalablement enregistrée dans le registre AH
- **DEC AH** → Décrémente de 1 la valeur enregistrée dans le registre AH
- **NOP** → Ne fait absolument rien (NO OPERATION)

Un fichier assembleur contient une liste d'instructions similaires. Chaque instruction se trouve sur une ligne et suit le schéma suivant :

MOT-CLE CIBLE,ORIGINE

Où MOT-CLE correspond à un mot clé du langage et où CIBLE et ORIGINE sont deux paramètres dont la présence ou non dépend du mot-clé (par exemple : **MOV** attend les deux paramètres, **DEC** n'attend que CIBLE, **NOP** n'attends rien).

Nous nous proposons d'écrire un programme en C capable de lire un fichier contenant des instructions assembleur et de simuler leur exécution (un tel programme s'appelle un émulateur).

Structure d'une instruction (2 points)

Une instruction est définie par un mot clé d'au plus 4 caractères, une cible représentée par un mot d'au plus 3 caractères et une origine représentée par une valeur numérique

Définir en C la structure correspondante.

Chargement du fichier (6 points)

Ecrire la fonction qui :

- Prend en paramètre le nom d'un fichier, un tableau d'instructions de taille fixe et la taille N de ce tableau.
- Rempli le tableau passé en paramètre avec les instructions lues dans le fichier
- Retourne le nombre total d'instructions effectivement lues ou -1 en cas d'erreur

Votre fonction fera tous les tests nécessaires pour assurer la sécurité de l'exécution du programme. Si le nombre d'instructions dans le fichier est supérieure à N, alors seules les N premières instructions seront lues.

Émulation des instructions (5 points)

Le processeur que nous souhaitons émuler possède 3 registres, qui sont des emplacements mémoire (ou des variables) prédéfinis qui peuvent être manipulés par les instructions assembleur.

Ces registres sont nommés R1, R2, R3 et mesurent chacun 16 bits.

Ecrire une fonction qui :

- Prend en paramètre un tableau d'instructions, la taille de ce tableau d'instructions, et un tableau de 3 éléments représentant les registres.
- Exécute toutes les instructions du tableau passé en paramètre dans l'ordre selon les règles suivantes (nous nous limitons à l'émulation de 3 instructions assembleur) :
    - Si l'instruction est « NOP », on passe à l'instruction suivante sans rien faire
    - Si l'instruction est « INC CIBLE », on incrémente la valeur du registre CIBLE (CIBLE peut valoir soit R1, soit R2, soit R3)
    - Si l'instruction est « ADD CIBLE,ORIGINE », on met la valeur numérique ORIGINE dans le registre CIBLE