TDB - PERIER Hugo - Année 3 2017/2018

TD6- Langage C

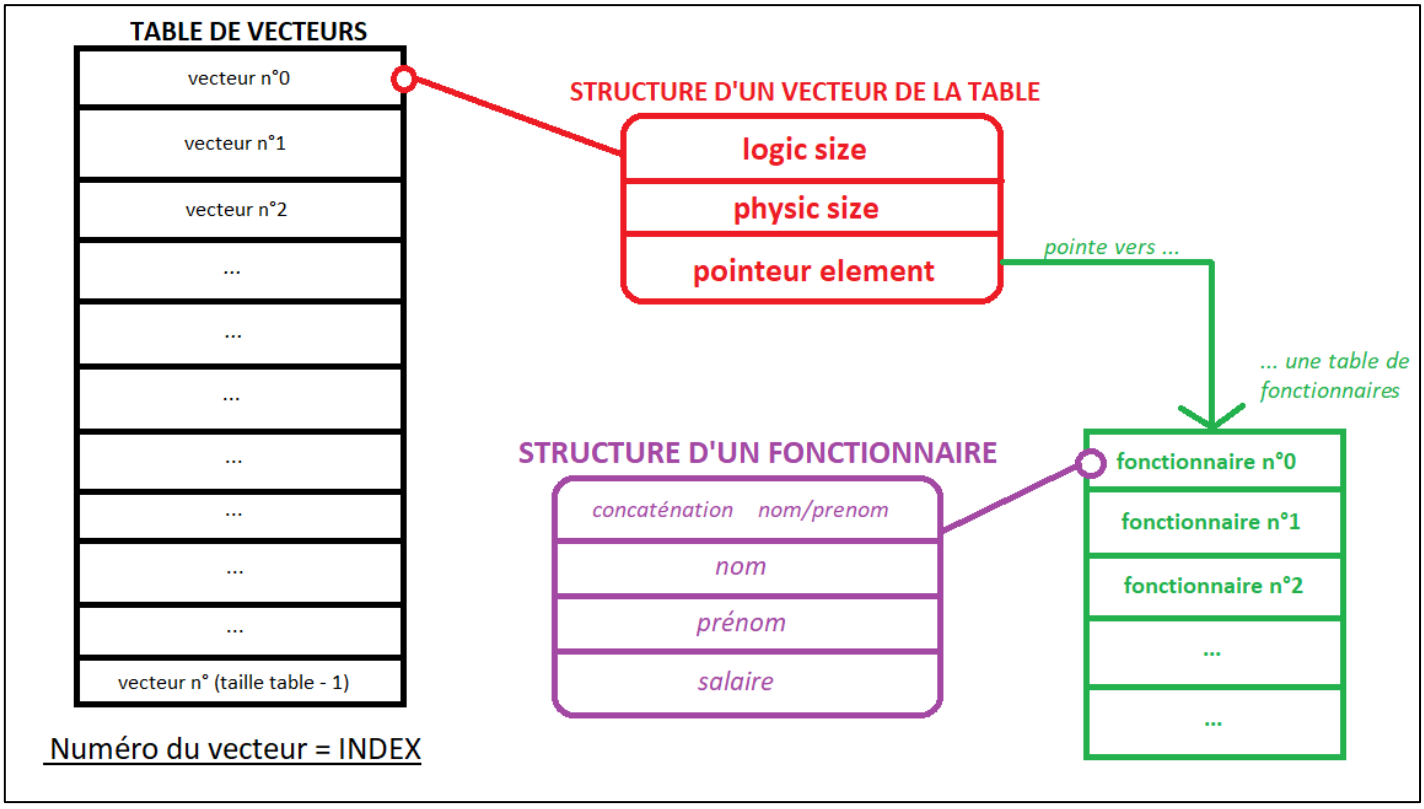
- Mini Rapport -

I-/ Explications choix de la « table »

J'ai construit un tableau d'éléments structurés de type « vecteur ». Chaque index calculé correspond à un élément de la table, donc à un vecteur. Un vecteur est composé de 3 éléments : un pointeur pointant vers un tableau de fonctionnaires (possédant tous le même index), un entier correspondant au nombre de « cases » de ce tableau occupées par des fonctionnaires : c'est le logic size, et un autre correspondant au nombre de « cases » du tableau au total (certaines cases pouvant etre vides) : c'est le physic size.

Afin de classer les fonctionnaires dans le tableau interne du vecteur, j'ai créé une structure fonctionnaire composée de 4 éléments : le nom concaténé de ce dernier, son nom, son prénom et son salaire.

Voici un schéma explicatif de l'aspect structurel de la « table » :



II -/ La fonctionnalité index

- La fonction concaténation alloue dans un premier temps un espace mémoire de taille 7 caractères (6 pour le nom concaténé et 1 pour le '\0' de clôture). Si l'allocation a fonctionnée, on parcourt le nom et le prénom du fonctionnaire afin de créer le nom concaténé qui si possible doit contenir 4 lettres du nom et 2 du prénom. Dans le cas contraire a l'aide de commandes « if » la fonction s'adapte aux tailles des noms et prénoms reçus en paramètre. Enfin, le dernier terme du nom concaténé est défini par : '\0'.
- La fonction puissance est nécessaire pour calculer l'index, elle est définie par récurrence.
- Pour le calcul de l'index, le type « long long Int » s'est avéré etre nécessaire du fait de la taille des chiffres manipulés.
- Enfin la fonctionnalite_index prend en compte les printfs et appelle les autres fonctions.

III-/ Ajouter

- Dans un premier temps il faut vérifier si le fonctionnaire à ajouter est déjà présent dans la table. La recherche du fonctionnaire dans la table se fait par le biais de son nom/prénom concaténé. Ce choix semble aussi efficace qu'une comparaison sur le nom et sur le prénom et est plus rapide. Il est alors nécessaire de créer une autre fonction :
- Il s'agit de « comparer_2_chaines_char » qui compare la taille des chaines ainsi que les caractères un à un ; elle renvoie ture si les deux chaines sont identiques.
- Une fonction « créer fonctionnaire » créé tout simplement l'élément structuré à introduire dans la table ;
- L'ajout de ce dernier (par défaut en fin du tableau interne du vecteur correspondant à l'index du fonctionnaire) se fait grâce à la fonction ad du TD4 de langage C ; elle renvoie true si l'ajout a bien été fait.
- Afin de classer par ordre alphabétique, il y a la fonction « classer_fonctionnaire_alphab ». Si le tableau interne contient plus de 1 fonctionnaire, ce dernier est parcouru du début à la fin. Si le nom analysé est identique à celui à ranger, alors on le place directement à gauche de celui-ci (par défaut). Sinon on compare le maximum de caractères possibles entre les noms (défini par une autre fonction : « nombre_character_max ») et on obtient le rang ou placer le fonctionnaire.
- Ce dernier est rangé avec la fonction « mettre_fonct_au_rang » qui décale tous les fonctionnaires de la table après le rang ou on place le fonctionnaire à ranger.

IV-/ Charger

- Dans un premier temps on ouvre le fichier chicago.txt, avec un pointeur défi sur lui. Le premier fscanf permet d'obtenir le nombre total de fonctionnaires du document.
- Puis on crée une chaine_chargee assez grande pour accueillir à chaque itération de la boucle chaque lignes chargées grâce à la fonction fgets.
- La fonction « lire_ligne_chargee » sépare le nom du prénom et du salaire espacés dans la chaine par simple parcours de cette dernière et repérage des espaces.

- Il suffit ensuite de créer le fonctionnaire associe, de vérifier s'il est déjà présent dans la table et de l'ajoute ou non avec les fonctions créées précédemment.

VI-/ Afficher salaire

- Il suffit de parcourir le tableau interne correspondant au vecteur de l'index correspondant au nom et prénom entrés par l'utilisateur. Cette recherche dans la table se fait par le bien de la comparaison des noms concaténé. Si le salaire est trouvé il est associé à l'entiersalaire_a_afficher, sinon il reste à sa valeur initiale soit « -1 ».

VII-/ Afficher entre

- Cette fonction utilise « afficher table » qui parcours tous les vecteurs de la table ; et dans chaque vecteurs tous les fonctionnaires présents dans le tableau interne. Il suffit simplement d'indiquer les index de début et de fin pour le parcours des vecteurs.

VIII-/ Nombre de conflits

- Il y a un conflit uniquement s'il y a plus d'un fonctionnaire par vecteur. Il suffit donc de compter le nombre de fonctionnaires au-dessus du 1er pour chaque vecteur de la table.

IX-/ Taille moyenne des conflits

- Dès que la logic size d'un vecteur est supérieure à 1 : on affiche le nombre de fonctionnaire dans l'index correspondant.

X-/ Supprimer

- On recherche le fonctionnaire à supprimer dans le vecteur correspondant à son index. Puis on crée un nouveau tableau interne plus petit, on copie tous les éléments dedans et on libère l'espace mémoire correspondant à l'ancien tableau.

XI-/ Supprimer entre

- On parcourt la table de vecteur et pour chaque vecteur entre les index entres par l'utilisateur, on libère l'espace mémoire du pointeur fonctionnaire, on le positionne ensuite a NULL et on initialise les paramètres du vecteur.

XII-/ Quitter

- On supprimer tous les tableaux internes avec la fonction précédente. Puis on exécute exit(0).