Pratique de la programmation et projet

TP 11 : Pointeurs de pointeurs, pointeurs de fonctions et joyeusetés afférentes

Frédéric Vivien

Pointeurs de pointeurs

Écrivez une fonction d'allocation de pointeurs sur entiers, qui prenne en entrée (entre autres) la taille à allouer, et qui renvoie une valeur de diagnostique (0 si l'allocation s'est bien passée, et -1 sinon).

Dans la fonction `main` de votre programme, déclarez un pointeur sur entier (initialisé à NULL), et allouez-le au moyen de la fonction précédemment écrite. Vérifiez le résultat.

Pointeurs de fonction

1. Écrivez (ou récupérez d'un TP précédent) une fonction de tri croissant d'une liste d'entiers.
2. À partir de la fonction précédente, écrivez une fonction de tri d'une liste d'entiers, la fonction de comparaison étant passée en argument.
3. Écrivez deux fonctions de comparaison de deux entiers, chacune de ces fonctions prendra deux entiers en argument, la première (respectivement deuxième) sera vraie si son premier argument est plus grand (resp. plus petit) que le deuxième. Au moyen de ces fonctions et de la fonction précédente, réalisez le tri croissant et le tri décroissant d'une liste d'entiers.
4. Nous voulons implémenter le schéma de calcul récursif suivant :

   fonction(opération, neutre, liste) =
   si liste = vide
   alors renvoyer neutre
   sinon renvoyer opération(tête(liste), fonction(opération, neutre, reste(liste)))

   (a) Écrivez une fonction réalisant ce schéma de calcul en supposant que la liste est une liste d'entiers, et que `opération` est une fonction qui prend deux entiers en entrée et renvoie un entier.

   (b) Au moyen de la fonction précédente, calculez la somme des éléments d'une liste d'entiers.

   (c) Toujours au moyen de la même fonction, calculez le maximum d'une liste d'entiers positifs.

   (d) Finalement, calculez le longueur d'une liste.
5. Nous réalisons des listes chaînées au moyen du type suivant :

```
typedef struct student {
char * nom;        int age;      float taille;       void * truc;
struct student * suivant;
} Eleve;
```

   (a) Écrivez une fonction de tri d'une liste d'objets de type `Eleve`. La fonction de comparaison étant passée en argument.

   (b) Réalisez un tri croissant sur le champ `age` d'une liste d'objets de type `Eleve`.

   (c) Réalisez un tri croissant sur le champ `taille` d'une liste d'objets de type `Eleve`.

   (d) Réalisez un tri croissant sur le champ `truc` d'une liste d'objets de type `Eleve`, en supposant que ce champ sert à stocker un entier.