AP1 TP

13

Semaine 13 : Pointeurs 1/2

Vous commencerez par récupérer les fichiers sources placés dans `/net/Bibliotheque/AP1/TPSem13`.

Exercice 1 : Tableaux et pointeurs

Cet exercice a pour but de mettre en évidence la similitude entre tableau et pointeur. Le programme `exo-1.cc` permet de saisir un tableau en passant l'adresse de sa première case et l'adresse de la case après sa dernière. Complétez ce programme tel qu'indiqué dans le source du programme :

1. Écrivez une action qui affiche le contenu du tableau.
2. Écrivez une fonction qui retourne la somme des éléments du tableau.
3. Vérifiez que l'adresse du tableau n'est pas modifiable.
4. Verifiez enfin qu'un tableau n'est pas recopié lorsqu'il est passé en paramètre. Les variables pointeur `p1` et `p2` sont-elles passées par valeur, par référence, ou par adresse ?

Un tableau (e.g. de `int`) est donc, à l'utilisation, un pointeur (vers un `int`) (dont l'adresse n'est pas modifiable). Cela explique pourquoi les tableaux sont toujours passés en "entrée/sortie". A l'instanciation, un certain nombre de cases consécutives sont allouées en mémoire pour un tableau.

Exercice 2 : Passage par référence et par valeur

Cet exercice permet de vérifier le principe du passage par valeur et du passage par référence.

1. Écrivez une fonction `sqr` qui prend deux entiers en arguments et donne la valeur du carré du premier argument au second argument.
2. A l'intérieur de cette fonction, affichez les valeurs des adresses des paramètres.
3. Comparez ces valeurs aux adresses des arguments fournis par la fonction principale `main`. Que peut-on conclure ?

Exercice 3 : Passage par adresse

Le passage par adresse est un moyen de passer des paramètres en entrée-sortie. Plutôt que de passer une variable par référence, on peut passer l'*adresse* de cette variable *par valeur* : c'est le *passage par adresse*.

1. Complétez le fichier `exo-2.cc` en écrivant les deux fonctions `echange1` et `echange2` qui effectuent l'échange des contenus des deux paramètres, l'une en effectuant un passage par référence et l'autre un passage par adresse.
2. Compilez et exécutez pour vérifier l'équivalence entre les deux techniques.
3. Affichez l'adresse des variables `i` et `j` dans le `main`, puis affichez l'adresse des "variables" `a` et `b` dans la fonction `echange1`. Que constatez-vous ?

Remarque : le passage par adresse est le seul possible en C. Le C++ a défini le passage par référence pour éviter les lourdeurs de notations du passage par adresse (qui est peu lisible). En réalité, le compilateur fabrique la même chose pour les deux fonctions.

Exercice 4 : Adresses des Min-Max d'un tableau

1. Ecrire une fonction qui reçoit comme paramètres un tableau d'entiers et sa taille, et retourne un pointeur vers l'élément contenant la plus petite valeur du tableau.
   Utilisation :

```
int main()
{
const int MAX = 10;
  int t[MAX] = { 3, 7, 4, 1, 6 };
  int taille = 5;
  int *ptrmin;
  ptrmin = pointeurMin( t, taille );
  cout << "min :" << *ptrmin << endl;
  cout << "Adresse min :" << ptrmin << endl;
}
```

2. On veut maintenant avoir en sortie deux pointeurs, l'un vers l'élément contenant la plus petite valeur du tableau, l'autre vers l'élément contenant la plus grande.
   Il faut donc transformer la fonction précédente en action, et passer les deux pointeurs en paramètre.
   Utilisation :

```
int main()
{
  ...
  int *ptrmin, *ptrmax;
  pointeursMinMax( t, taille, ptrmin, ptrmax );
  cout << "min :" << *ptrmin << endl
       << "max :" << *ptrmax << endl;
  cout << "Adresse min :" << ptrmin << endl
       << "Adresse max :" << ptrmax << endl;
}
```

Exercice 5 : Algorithmes de tri par indirection

1. Reprenez un des algorithmes de tri simples implémentés lors du TP09 (sélection, bulle ou insertion).
2. Si vous ne l'avez pas déjà fait, modifiez cet algorithme pour qu'il réalise un tri par indirection, c'est-à-dire en passant par un **tableau d'entiers contenant les indices** des éléments dans le tableau original, qui lui n'est pas modifié lors du tri.
3. Faites une autre version de tri par indirection, en remplaçant le tableau d'entiers contenant les indices par un **tableau de pointeurs contenant les adresses** des éléments dans le tableau original.