TP_s7.1: pointeur (solution)

Lien avec le MOOC Initiation à la Programmation (en C++)

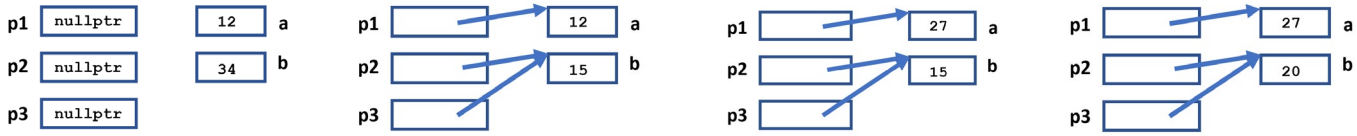
Exercices semaine7.1 du MOOC : pointeur

Exercice complémentaire (ExC)

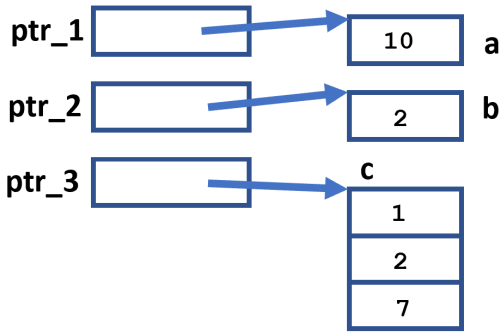
ExC 6: évaluation de code sans le compiler (niveau 1)

p1 pointe sur **a**, **p2** et **p3** pointent sur **b**. Les affectations avec **indirection** sur **p2** ou sur **p3 à gauche de l'opérateur '='** modifient **b**.

De même l'affectation avec une **indirection** sur **p1 à gauche de l'opérateur '='** modifie a. Les valeurs finales sont: **a: 27** et **b: 20** . Voici la séquence des mises à jour des variables (de la gauche vers la droite) :



ExC 7 : Type et valeur d'expressions avec des pointeurs et un tableau à-la-C (niveau 1)



Dessin de l'état initial

a) La valeur de l'expression `*ptr_1 + *ptr_2` est 12
La valeur de l'expression `c[1] == *ptr_2` est true
La valeur de l'expression `*c + b` est 3
La valeur de l'expression `*(c+b)` est 7
Celle de l'expression :

```
ptr_1 == ptr_2 ? 2* *ptr_1:3* *ptr_2;
```

la condition étant fausse et on exécute la dernière sous-expression après `:` ce qui donne la valeur 6

b) l'exécution de l'instruction `*ptr_1 = *ptr_2 + a ;`
modifie seulement la variable pointée par **ptr1** car l'expression ***ptr1** est à gauche de l'opérateur d'affectation. La variable **a** prend la valeur 12 tandis que la variable **b** ne change pas et reste à 2.

c) Le type de l'expression **&ptr_1** est « l'adresse » de « l'adresse d'un int » c'est-à-dire : `(int*)*` que l'on écrit en général avec `int**` (remarquer l'associativité **Droite-Gauche** de l'opérateur ***).

Le type de l'expression `*ptr_1` est « type de l'objet pointé par ptr1 », c'est-à-dire `int`.

d) L'instruction **c++**; n'est pas valide, car **c** est un pointeur constant ; il ne peut pas être à gauche d'un symbole d'affectation. On dit encore que **c** n'est pas une *lvalue* (comme *left-value* parce qu'elle est à gauche de l'opérateur d'affectation). Le compilateur produira un message d'erreur.

Par contre `ptr_3++;` est valide car c'est une variable ; `ptr3` pointe alors sur `c[1]`