Série 12: pointeur et chaînes à-la-C

Opérateurs Sizeof et cast

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

Exercice complémentaire (ExC)

ExC9 : passage d'arguments à la fonction main() et manipulations de chaînes à-la-C (niveau 1)

1) Ecrire un programme exploitant la forme de la fonction main avec les deux paramètres **argc** et **argv** et qui affiche l'ensemble des chaînes reçues au moment du lancement du programme sur la ligne de commande du terminal. Ecrire les chaines les unes à la suite des autres avec un espace comme séparateur.

2) Ce mécanisme de transmission est utilisé par les commandes linux pour indiquer toutes sortes d'options pour ces commandes. Vous allez adopter un style similaire d'option pour offrir un fonctionnement différent de la question 1 :

- si la première chaine ne commence PAS par le caractère '-' alors afficher sur une ligne seulement les chaines qui suivent le nom du programme, dans l'ordre dans lequel elles sont données.

Sinon
- si la première chaine fournie à la suite du nom de l'exécutable est « **-r** »
    - alors afficher les chaines supplémentaires dans l'ordre inverse

- si la première chaine fournie à la suite du nom de l'exécutable est « **-s** »
    - alors afficher les chaines supplémentaires dans l'ordre normal mais une seule par ligne

- si la première chaine fournie à la suite du nom de l'exécutable est « **-rs** » ou « **-sr** »
    - alors prendre en compte les deux options précédentes

- sinon afficher le message « *Incorrect option : accepted options are r for reverse and s for single* »

Comment faire pour accéder à un caractère d'une chaîne transmise à main ?

→ il suffit d'utiliser le pointeur qui donne l'accès à une chaine comme un nom de chaine-à-la-C : on y ajoute un second opérateur [ ] avec une valeur d'indice pour isoler un caractère en particulier.

Exemple1 : soit l'exécutable **prog** qui est lancé sur la ligne de commande avec : **./prog**
La fonction main() reçoit avec **argv[0]** un pointeur sur la chaîne à-la-C qui contient « **./prog** » et qui se termine avec le caractère **\0**.

L'expression argv[0][0] donne le premier caractère de cette chaîne, c'est-à-dire le caractère '**.**' . On peut accéder et tester individuellement chaque caractère de chaque chaîne transmise à main().

Exemple2 :la commande **./prog –r un deux trois**
Produit l'affichage suivant : **trois deux un**

Pour vos sections, il est utile de maîtriser un minimum les **tableaux à-la-C**. Pour l' exercice suivant le mot « **tableau** » désigne un « **tableau à-la-C** ».

ExC 10 : pointeur de pointeur illustré sur un tableau à-la-C de pointeurs sur des chaînes à-la-C (niveau 1)

Le tableau exploité ici ressemble au tableau **argv** des pointeurs vers les chaînes de caractères à-la-C de l'exercice précédent mais il présente néanmoins une différence importante. Il s'agit ici d'un tableau de pointeurs de chaînes de caractères **constantes,** comme illustré par le tableau **t** suivant**.**

Le type du tableau **t** est d'être l'adresse du premier élément du tableau, ce qui donne (**const char *) *.** C'est un pointeur de pointeur.

```
#include <iostream>
#include <cstring> // remarquer le 'c' devant string
                   // pour chaine à-la-C

using namespace std;

int main(int argc, char* argv[])
{
     // on obtient la longueur d'une chaîne à-la-C avec strlen()
     // la longueur n'inclut pas le caractère de fin de chaine
     cout << argv[0] << "  de longueur = "
          << strlen(argv[0]) << endl;


     const char * t[]= {"Rome","Oslo","Bern","Wien"};

     const char * a_ptr = t[1];
     const char * b_ptr = t[2];
     const char ** p = t;

     cout << strlen(a_ptr) << endl;
     a_ptr += 2;
     cout << strlen(a_ptr) << endl;

     cout << *b_ptr        << endl;

     cout << static_cast<char> (*b_ptr + 1) << endl;

     p += 3 ;

     cout << *p   << endl;

     cout << **p  << endl;

     return 0;
}
```

Si on fait une seule indirection sur ce nom de tableau, avec l'expression ***t,** on obtient la valeur du premier élément du tableau t ; la valeur de ce premier élément est aussi obtenue avec l'expression **t[0]**. C'est l'adresse de la première chaîne de caractères constante. Si on fait deux indirections, avec l'expression ****t**, c'est équivalent à ***t[0]**, ce qui fournit le premier caractère de la première chaîne constante.

Un but de cet exercice est de continuer à se familiariser avec la représentation avec boite et flèche comme dans le cours. En effet l'adresse des variables n'est pas connue tant que le code n'est pas compilé et de toute façon une flèche suffit pour indiquer qu'un pointeur pointe sur une variable ou un élément de tableau, ou un début de ligne de tableau.

a) Dessinez, avec des boites et des flèches, l'état initial correspondant aux déclarations jusqu'à la déclaration de **p** incluse.

b) Déterminez ce qui est affiché et dessinez l'état de toutes les variables après exécution des instructions jusqu'au return.

c) Que se passe-t-il si vous ajoutez l'instruction ****p = 'X';** , par exemple juste avant le return ? Compile-t-il ? y a t-il un warning ? Si un exécutable est produit, que se passe-t-il à l'exécution?

ExC 11 : reprendre les exemples de code vu en classe inversée cette semaine (niveau 0-1-2)

Compilez, exécutez et explorez les programmes fournis sur moodle pour vous familiariser avec les concepts vus cette semaine