Séance 5&6 - Manipulation de pointeurs

A FAIRE SUR PAPIER

Exercice 1 Écrire un programme qui définit deux variables a et b de type entiers ; puis deux pointeurs $p1$ et $p2$ contenant respectivement l'adresse de a et de b. Ce programme devra affecter les valeurs $a = 5$ et $b = 8$ sans utiliser d'affectation directe sur les variable a et b (`a=5;b=8;` interdit).

Exercice 2 Dans le programme ci-dessous, évaluez après chaque instruction (à partir de la ligne 8) les valeurs des entiers a, b et c.

```
1  #include<iostream>
2  int main(){
3          int a,b,c;
4          int *p1, *p2;
5          a=b=c=3;
6          p1=&a;
7          p2=&b;
8          c=a+b;
9          *p2=b+2;
10         a=*p2**p1;
11         p1=p2;
12         *p2=*p1-a;
13         p2=&(*p1);
14         *p2=*p1+ *(&(*p2));
15         a=b+*p2;
16   return 0;
17 }
```

Exercice 3 Écrire une fonction `plusUn` qui permet d'incrémenter de un la valeur d'une variable entière x définit en dehors de la fonction (cette variable ne sera pas globale). Écrire un programme qui utilise la fonction `plusUn` pour incrémenter une variable x dont la valeur initiale est saisie par l'utilisateur.

Exercice 4 Écrire une fonction `saisieNote` qui permet à un utilisateur de saisir une note entre 0 et 20. Tant que l'utilisateur ne fait pas une saisie de note correcte (entre 0 et 20) la fonction réitère la demande d'une note. Attention, cette fonction ne doit renvoyer aucune valeur, autrement dit son type de retour est `void`. On utilisera une transmission de la note saisie par un paramètre modifiable. Écrire un programme qui demande la saisie de 4 notes et qui affiche la plus grande.

A FAIRE SUR MACHINE

Exercice 5 On cherche à construire une fonction qui permet, à partir de trois variables entières a, b et c, de renvoyer l'adresse de la variable contenant la plus grande valeur.
Est-ce que le code suivant répond à notre problème ? Expliquer votre réponse.

```
1  #include <iostream>
2  int* max(int a, int b, int c){
3          if (a>=b && a>=c) return &a;
4          if (b>=a && b>=c) return &b;
5          if (c>=a && c>=b) return &c;
6  }
7  int main(){
8    int x=2,y=4,z=3;
9    std::cout<<"le max est à l'adresse: "<<max(x,y,z)<<std::endl;
10   return 0;
11 }
```

Si votre réponse est non, modifiez la fonction et/ou le programme pour que tout fonctionne correctement.

Exercice 6 Les nombres complexes $(a + ib)$ peuvent être manipulés en utilisant deux variables : une pour la partie réelle et une autre pour la partie imaginaire. Définissez les fonctions permettant de calculer l'addition et la multiplication de deux nombres complexes. Attention, une seule fonction doit être écrite par opération.

Exercice 7 Écrire un programme permettant de calculer la distance en mémoire (en nbr d'octets) de deux variables `a` et `b`. Vous testerez votre programme pour des variables de type `int` et `double`. Que remarquez-vous ?

A FAIRE SUR PAPIER

Exercice 8

1. Écrire une fonction `Identique` qui prend deux paramètres de type `int*` et qui teste si ils référencent la même zone mémoire.
2. Écrire une fonction `ValeurIdentique` qui prend deux paramètres de type `int*` et qui teste si les valeurs pointées sont identiques.
3. Écrire un programme qui montre que les fonctions `ValeurIdentique` et `Identique` ne sont pas équivalentes.

Exercice 9 Corrigez le code ci-dessous pour que le programme compile et affiche le bon résultat (les valeurs doivent être échangées en utilisant la fonction `echange`)

```
1  #include <iostream>
2
3  void echange(int a, int b){
4    int t=a;
5    a=b;
6    b=t;
7  }
8
9  int main(){
10   int x,y;
11   int *p;
12   x=2;y=3;
13   std::cout<<x<<" "<<y<<std::endl;
14   p=&y;
15   echange(&x,*p);
16   std::cout<<x<<" "<<y<<std::endl;
17   return 0;
18 }
```

Exercice 10 Écrire une fonction `saisieDate` permettant la saisie une date au format jj/mm/aaaa. Vous utiliserez des variables entières pour représenter le jour, le mois et l'année. Attention, votre fonction doit vérifier que votre date est donnée dans le bon format et que les valeurs du jour et du mois sont valides ($0 < jour \leq 31$ et $0 < mois \leq 12$). **AIDE :** la saisie d'un entier par `std::cin` s'arrête dès que le caractère lu n'est plus numérique. Le caractère non numérique pourra être récupérer par une autre saisie. Écrire un programme qui appelle la fonction `saisieDate` et qui affiche la date saisie. Vous testerez votre programme sur les saisies suivante : `24/12/2015  24:12:2015  24 12 2015  32/1/1998  1/13/20`

A FAIRE SUR MACHINE

Exercice 11 Écrire une fonction de résolution d'équations du second degré. Cette fonction prendra 5 arguments : les 3 premiers seront les paramètres de l'équation et les 2 derniers seront des paramètres permettant de retourner les solutions de l'équation, si il y en a. Cette fonction retournera le nombre de solutions trouvées. Vous utiliserez la bibliothèque `cmath` pour avoir la fonction racine carrée `sqrt`. Écrire un programme permettant de vérifier votre fonction de résolution. Vous testerez votre programme sur les équations suivantes : $x^2 + 4x + 1 = 0$, $x^2 + 2x + 1 = 0$, $2.5x^2 + 10x + 10 = 0$, $x^2 + -4x + 1 = 0$

Exercice 12 (Euclide) L'algorithme d'Euclide récursif permet de calculer le pgcd de deux entiers a et b en effectuant un appel récursif sur b et $a \bmod b$ (sous condition que $a > b$). L'appel récursif s'arrête dès que le plus petit opérande est nul, le pgcd correspond alors à l'autre opérande.

$$\begin{aligned} pgcd(24, 18) &= pgcd(18, 24 \bmod 18) \\ &= pgcd(18, 6) \\ &= pgcd(6, 18 \bmod 6) \\ &= pgcd(6, 0) \\ &= 6 \end{aligned}$$

On peut voir l'ensemble des opérandes des appels récursifs comme une suite d'entiers régie par la loi $U_{n+1} = U_{n-1} \bmod U_n$. Dans notre exemple, la suite est :

$$U_0 = 24, U_1 = 18, U_2 = 6, U_3 = 0$$

Une version non récursive de l'algorithme consiste donc à calculer itérativement les couples d'entiers (U_i, U_{i+1}).

1. Écrire une fonction `OnePgcdStep` qui effectue une étape de l'algorithme d'Euclide : calcul du couple (U_i, U_{i+1}) à partir du couple (U_{i-1}, U_i) ou (U_i, U_{i-1}).
2. Utiliser la fonction précédente pour écrire la fonction complète `Pgcd`.
3. Écrire un programme complet qui teste votre fonction. Vous essaierez le programme avec les couples d'entiers suivant :

$$\mathsf{pgcd}(25, 70) = 5$$

$$\mathsf{pgcd}(123456, 654321) = 3$$

$$\mathsf{pgcd}(250, 30) = 10$$

$$\mathsf{pgcd}(2839488, 3827136) = 123456$$

$$\mathsf{pgcd}(1007, 107) = 1$$

Exercice 13 Écrire un programme qui affiche l'interprétation en codage flottant d'une zone mémoire ou est stocké un entier, et vice-versa. Par exemple, l'interprétation flottante du codage binaire de l'entier 17 est $2.382207389e{-}44$. Votre programme demandera à l'utilisateur de saisir un entier et affichera l'interprétation de son codage binaire comme un flottant ; puis il demandera de saisir un flottant et affichera l'interprétation de son codage binaire comme un entier. Vous essaierez avec les valeurs données ci-dessus, puis pour les entiers 23, -1040187392 et les flottants 32.0, $3.222986468e-44$.