|  |  |
| --- | --- |
| *Pointeurs* | *[Enseignement supérieur](https://fr.wikipedia.org/wiki/Enseignement_sup%C3%A9rieur)* |

“The computer was born to solve problems that did not exist before.”

float a=2;

float b=3;

float c, \*pa, \*pb;

pa= &a;

\*pa \*= 2;

pb= &b;

c= 3\*(\*pb - \*pa);

*Exercice 1 :*

Donner la valeur des variables a, b et c à la suite de l’exécution de ces instructions :

*Exercice 2 :*

Soit la fonction main suivante :

Exécuter manuellement le programme ci-dessus et compléter le tableau suivant :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **P1** | **P2** |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 11 |  |  |  |  |  |
| 12 |  |  |  |  |  |
| 13 |  |  |  |  |  |
| 14 |  |  |  |  |  |
| 15 |  |  |  |  |  |

void main()

{

    int A = 1;

    int B = 2;

    int C = 3;

    int \*P1, \*P2;

    P1 = &A;

    P2 = &C;

    \*P1 = (\*P2)++;

    P1 = P2;

    P2 = &B;

    \*P1 -= \*P2;

    ++\*P2;

    \*P1 \*= \*P2;

    A = ++\*P2 \* \*P1;

    P1 = &A;

    \*P2 = \*P1 /= \*P2;

}

*Exercice 3 :*

int \*a;

int e = 0;

a = &e;

(\*a)++;

Expliquer le résultat de chacune des instructions suivantes :

*Exercice 4 :*

Étant donné un ensemble E1 de n1 entiers et un ensemble E2 de n2 entiers, écrire un programme principal permettant de déterminer l'union, l'intersection et la fusion de ces deux ensembles.

NB : Les trois fonctions union, intersection et fusion doivent avoir un type de retour void.

*Exercice 5 :*

Étant donné un tableau de n entiers, écrire un programme principal faisant appel à des fonctions permettant de :

1. Remplir le tableau, La saisie du nombre d'éléments doit se faire avant l’appelle de la fonction.
2. Afficher le tableau.
3. Ajouter un élément au tableau.
4. Chercher un élément dans le tableau. la position doit être renvoyée en paramètres si l'élément est trouvé, sinon la valeur renvoyée est -1.
5. Modifier la première occurrence d'un élément donné.
6. Modifier toutes les occurrences d'un élément donné.
7. Supprimer la première occurrence d'un élément donné.
8. Supprimer toutes les occurrences d'un élément donné.

NB : Toutes les fonctions doivent avoir un type de retour void.

*Exercice 6 :*

Écrire une **structure complexe** qui représente un nombre complexe. Cette structure  
contient deux **flottants re** et **im** représentant respectivement la partie réelle et la partie  
imaginaire du nombre.

1. Écrire une fonction qui prend en paramètre deux flottants, et qui retourne un pointeur  
   vers un nouveau complexe. **Prototype : complexe \* new\_complexe(float r, float i) ;**
2. Écrire une fonction qui prend en paramètre un pointeur vers un complexe, et qui affiche ce complexe dans la console. **Prototype : void print\_complexe(complexe \*p) ;**
3. Écrire une fonction qui prend en paramètre deux pointeurs vers des complexes, et qui  
   retourne un pointeur vers un nouveau complexe, somme des deux reçus en paramètre.  
   **Prototype : complexe \* addition\_complexes(complexe \*c1, complexe\*c2) ;**
4. Écrire une fonction qui prend en paramètre deux pointeurs vers des complexes, et qui  
   retourne un pointeur vers un nouveau complexe, produit des deux reçus en paramètre.  
   **Prototype : complexe \* produit\_complexes(complexe \*c1, complexe \*c2) ;**