**Excercice1 :**

Soit le programme C suivant :

|  |
| --- |
| 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;  } |

Copiez le tableau suivant et complétez-le pour chaque instruction du programme ci-dessus.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | P1 | P2 |
| Init. | 1 | 2 | 3 | / | / |
| P1=&A | 1 | 2 | 3 | &A | / |
| P2=&C |  |  |  |  |  |
| \*P1=(\*P2)++ |  |  |  |  |  |
| P1=P2 |  |  |  |  |  |
| P2=&B |  |  |  |  |  |
| \*P1-=\*P2 |  |  |  |  |  |
| ++\*P2 |  |  |  |  |  |
| \*P1\*=\*P2 |  |  |  |  |  |
| A=++\*P2\*\*P1 |  |  |  |  |  |
| P1=&A |  |  |  |  |  |
| \*P2=\*P1/=\*P2 |  |  |  |  |  |

**Excercice2 :**

Soit P un pointeur qui 'pointe' sur un tableau A:

**int A[] = {12, 23, 34, 45, 56, 67, 78, 89, 90};**

**int \*P;**

**P = A;**

Quelles valeurs ou adresses fournissent ces expressions:

|  |  |  |  |
| --- | --- | --- | --- |
| **a)** | **\*P+2** |  |  |
| **b)** | **\*(P+2)** |  |  |
| **c)** | **&P+1** |  |  |
|  |  |  |  |
| **d)** | **&A[4]-3** |  |  |
| **e)** | **A+3** |  |  |
| **f)** | **&A[7]-P** |  |  |
| **g)** | **P+(\*P-10)** |  |  |
| **h)** | **\*(P+\*(P+8)-A[7])** |  |  |

Exercice 4:

Ecrire un programme qui range les éléments d'un tableau A du type int dans l'ordre inverse. Le programme utilisera des pointeurs P1 et P2 et une variable numérique AIDE pour la permutation des éléments.

Exercice 5:

Ecrire un programme qui lit un entier X et un tableau A du type int au clavier et élimine toutes les occurrences de X dans A en tassant les éléments restants. Le programme utilisera les pointeurs P1 et P2 pour parcourir le tableau.

Exercice6:

Ecrire un programme qui lit 10 phrases d'une longueur maximale de 200 caractères au clavier et qui les mémorise dans un tableau de pointeurs sur char en réservant dynamiquement l'emplacement en mémoire pour les chaînes. Ensuite, l'ordre des phrases est inversé en modifiant les pointeurs et le tableau résultant est affiché. Finalement, l’espace alloué pour les chaines est libéré.

**Exercice7:**

Un employé est caractérisé par un code, nom, prénom, poste, salaire, commission, le code de département, le code de son chef.

Un employé qui n’est pas affecté à un département admet la valeur 0 pour le code de département (la même chose pour le code du chef).

1. Définir la structure d’un employé
2. Ecrire un programme qui permet de remplir un tableau de n employés. Vous devez parcourir le tableau en utilisant un pointeur
3. Etant donné le code de l’employé, écrire un programme qui permet d’afficher les détails d’un employé (code, nom, prénom, salaire, commission, nom du responsable et le code de département)
4. Ecrire un programme qui permet de copier tous les employés qui ne sont pas affecté à un département dan un tableau T\_non\_affect.

**Exercice 8**

On utilise la même structure de l’exercice n°7. Un département est caractérisé par un code, un nom de département et le code de responsable.

1. Etant donné un nom de département, écrire une fonction qui détermine le code de ce dernier (la fonction est de type void).
2. Ecrire une fonction qui détermine le nombre des employés affectés à un département dont le code est donné.
3. Ecrire une fonction qui détermine le salaire le plus élevé et le nombre des employés recevant ce salaire d’un département donné
4. Ecrire un programme qui permet de saisir le nom de département, déterminer et afficher le nombre des employés affecté à ce département, le salaire le plus élevé et le nombre des employés qui le reçoivent .

Vous devez utiliser le formalisme pointeur dans les différentes questions.