

### **Exercice 1**

Ecrire un programme qui lit deux tableaux A et B et leurs dimensions N et M au clavier et qui ajoute les éléments de B à la fin de A. Utiliser le formalisme pointeur à chaque fois que cela est possible.

### **Exercice 2**

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.

### **Exercice 3**

Ecrire un programme qui range les éléments d'un tableau A du type **int** dans l'ordre inverse.

### **Exercice 4**

Soient deux tableaux t1 et t2 déclarés ainsi : float t1[10], t2[10].

Écrire un programme permettant de recopier, dans t1, tous les éléments positifs de t2, en complétant éventuellement t1 par des zéros.

### **Exercice 5**

Écrire un programme allouant dynamiquement un emplacement pour un tableau d'entiers, dont la taille est saisie au clavier, puis remplit ce tableau par des nombres entiers lus.

Créer ensuite dynamiquement un nouveau tableau qui contient les carrés des nombres contenus dans le premier.

### **Exercice 6**

Ecrire un programme qui calcule le nombre d'occurrences d'un caractère dans une chaîne donnée.

Ecrire un programme qui calcule le nombre d'occurrences d'une sous-chaîne dans une chaîne donnée.

### **Exercice 7**

Ecrire un programme qui concatène une chaîne s1 à une chaîne s2 (sans utiliser la fonction strcat, utiliser le formalisme pointeur).

Proposer plusieurs solutions : en utilisant / sans utiliser une chaîne résultat.

### **Exercice 8**

Refaire tous les exercices du TD1 en utilisant le formalisme pointeur pour le parcours des chaînes de caractères.

### Exercice 9

Ecrire un programme qui lit un caractère C et une chaîne de caractères CH au clavier. Ensuite toutes les occurrences de C dans CH seront éliminées. Le reste des caractères dans CH sera tassé à l'aide d'un pointeur et de la fonction **strcpy**.

### Exercice 10

Dessiner le résultat de l'exécution de ce programme en précisant les contenus de chaque variable :

```
char buffer[4096];
char *ptr;
int i;
int ta[5], tb[5], tc[5];
buffer[10] = 'a';
*(buffer + 9) = 'b';
i = 0;
buffer[i++] = 'c';
i += 15;
ptr = buffer + i;
ptr[0] = 'd';
*++ptr = 'e';
*ta = 1, *tb = 2;
tc[0] = ta[0] + tb[0];
```