Exercice 1

Question 1

Écrivez la fonction `void afficher_tableau_2d(int tab[3][3], int nb_lignes, int nb_colonnes)` qui affiche sur contenu d'un tableau à deux dimensions.

Par exemple :

```
int main() {
        int tableau[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        afficher_tableau_2d(tableau, 3, 3);

        return 0;
}
```

doit afficher :

```
1 2 3
4 5 6
7 8 9
```

Question 2

réécrire pour que ça affiche

```
1 4 7
2 5 8
3 6 9
```

Exercice 2

Écrivez la fonction `init_tableau_2d(int tab[3][3], int nb_lignes, int nb_colonnes)` qui initialise à `0` toutes les cases du tableau `tab` qui est de deux dimensions.

Exercice 3

Écrivez la fonction `void additionner_matrices(int tab1[3][3], int tab2[3][3], int nb_lignes, int nb_colonnes)` où `tab1` et `tab2` sont deux tableaux 2d de la même taille. On veut mettre dans `tab1` sa somme avec `tab2`

Par exemple :

```
int main() {
        int t1[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        int t2[3][3] = {
                {1, 1, 1},
                {1, 1, 1},
                {1, 1, 1}
        };

        additionner_matrices(t1, t2, 3, 3);
        afficher_tableau_2d(t1, 3, 3);
}
```

doit afficher

```
2 3 4
5 6 7
8 9 10
```

Exercice 4

Écrivez la fonction `void multiplier_matrices(int tab1[3][3], int tab2[3][3], int nb_lignes, int nb_colonnes)` qui met dans `tab1` la multiplication entre `tab1` et `tab2`.

Par exemple :

```
int main() {
        int t1[3][3] = {
                {1, 2, 3},
```

```
                {4, 5, 6},
                {7, 8, 9}
        };

        int t2[3][3] = {
                {2, 2, 2},
                {2, 2, 2},
                {2, 2, 2}
        };

        multiplier_matrices(t1, t2, 3, 3);
        afficher_tableau_2d(t1, 3, 3);
}
```

doit afficher

```
2 4 6
8 10 12
14 16 18
```

Exercice 5

Question 1

Écrivez la fonction `int somme_diagonale(int tab[3][3], int nb_lignes, int nb_colonnes)` qui renvoie la somme de toutes les valeurs dans la diagonale (celle qui part d'en haut à gauche vers en bas à droite).

Par exemple

```
int main() {
        int tableau[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        somme_diagonale(tableau, 3, 3); // doit renvoyer 15

        return 0;
}
```

Question 2

Même question mais pour la diagonale qui part d'en haut à droite vers en bas à gauche.

Exercice 6

Écrivez la fonction `int matrices_egales(int tab1[3][3], int tab2[3][3], int nb_lignes, int nb_colonnes)` qui doit renvoyer `1` si elles sont égales, `0` sinon.

```
int main() {
        int t1[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        int t2[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        matrices_egales(t1, t2, 3, 3); // renvoie 1

        // on modifie une case du tableau
        t1[1][1] = 0;

        matrices_egales(t1, t2, 3, 3); // renvoie 0

        return 0;
}
```

Exercice 7

Écrivez la fonction `void somme_lignes(int tab[3][3], int tab_somme[3], int nb_lignes, int nb_colonnes)` qui met dans la première case de `tab_somme` la somme de toutes les valeurs de la première ligne de `tab`, dans la deuxième case de `tab_somme` la somme de toutes les valeurs de la deuxième ligne de `tab`, etc...

Par exemple

```c
int main() {
        int tableau[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        int somme[3];

        somme_lignes(tableau, somme, 3, 3);

        print_tableau(somme, 3); // affiche [ 6 15 24 ]

        return 0;
}
```