**EXERCICE 1 (8 PTS)**

Écrire un algorithme qui :

1. Demande à l'utilisateur de saisir un tableau d'entiers **T** de taille N ne contenant que des 0 et des 1. **(3 pts)**
2. Demande ensuite à l'utilisateur de saisir un indice compris entre 0 et N-1 et qui affiche :
   - 0, si T[i] = 0
   - Le nombre des uns consécutifs dans T à partir de T[i] inclus, si T[i] = 1

   Par exemple, pour T = [0, 1, 1, 0, 0, 1, 1, 1, 0, 1], si l'utilisateur tape 0, le programme affiche 0 et s'il tape 5, le programme affiche 3. **(5 pts)**

**EXERCICE 2 (14 PTS)**

On souhaite créer une application qui gère les livres dans une bibliothèque.

1. Écrire un programme qui demande à l'utilisateur de saisir une liste nommée **Titres** qui contient 200 titres de livres. **(1 pt)**

Ensuite, à l'aide de parcours successifs de la liste, effectuer les actions suivantes

2. Afficher la liste des titres en majuscule en indiquant l'indice de chaque titre. **(1 pt)**
3. Trouver deux manières pour afficher les 7 derniers employés. **(1 pt)**
4. Demander un titre, puis afficher « existe » si celui-ci appartient à **Titres** ou « n'existe pas » sinon. **(1 pt)**
5. Compter le nombre des titres dupliqués dans la liste **Titres**. **(2 pts)**
6. Créer une autre liste **TitreLongueur** qui contient la longueur de chaque titre dans la liste **Titres**. **(2pts)**
7. Donner le plus grand élément dans **TitreLongueur** sans utiliser la fonction max(). **(2pts)**
8. Créer un dictionnaire **dicTitres** qui contient la correspondance de chaque élément apparaissant dans la liste **Titres** avec le nombre de ses occurrences dans **Titres**. **(2pts)**
9. Trier le dictionnaire **dicTitres**. **(2pts)**
   a. par ordre croissant en fonction du titre.
   b. par ordre décroissant en fonction de nombre d'occurrence.