

TP : prise en main

Dans ce TP, on n'utilise pas encore de fonctions : tout le code est écrit dans la fonction `main`. Pour chaque question, on part d'un petit programme contenant déjà un tableau et on complète uniquement la partie indiquée.

Dans la première question, on donne également le code permettant d'afficher le tableau. Dans les questions suivantes, ce code ne sera plus répété : il faudra réutiliser la même boucle d'affichage.

Question 1. Maximum d'un tableau

Compléter le programme suivant afin de déterminer puis d'afficher le maximum du tableau. On supposera le tableau non vide.

```
1 #include <stdio.h>
2
3 int main(void) {
4     int tab[7] = {3, 1, 7, -2, 5, 12, 4};
5
6     /* affichage du tableau */
7     int i = 0;
8     while (i < 7) {
9         printf("%d ", tab[i]);
10        i = i + 1;
11    }
12    printf("\n");
13
14    /* A COMPLETER : calcul du maximum */
15
16
17    /* affichage du resultat */
18    printf("maximum = %d\n", maximum);
19
20    return 0;
21 }
```

Quelle est la complexité temporelle de cet algorithme en fonction de la taille du tableau ?

Question 2. Moyenne des éléments d'un tableau

Compléter le programme suivant afin de calculer puis d'afficher la moyenne des éléments du tableau.

```
1 #include <stdio.h>
2
3 int main(void) {
4     int tab[7] = {2, 5, 8, 3, 4, 9, 4};
5
6     /* afficher le tableau en reutilisant la boucle de la question 1 */
7
8     /* A COMPLETER : calcul de la moyenne */
9
10    /* affichage du resultat */
11
12    return 0;
13 }
```

Quelle est la complexité temporelle de cet algorithme en fonction de la taille du tableau ?

Question 3. Recherche d'un élément

On cherche la valeur contenue dans la variable `cible`. Compléter le programme afin de placer dans `position` l'indice de la première occurrence de cette valeur, ou `-1` si elle n'apparaît pas dans le tableau.

On souhaite arrêter la recherche dès que la cible a été trouvée.

```
1 #include <stdio.h>
2
3 int main(void) {
4     int tab[7] = {5, 8, 13, 8, 2, 9, 4};
5     int cible = 8;
6
7     /* afficher le tableau en reutilisant la boucle de la question 1 */
8     int i = 0;
9
10    /* A COMPLETER : recherche de la cible */
11
12    /* affichage du resultat */
13
14    return 0;
15 }
```

Donner la complexité temporelle dans le meilleur cas puis dans le pire cas.

Question 4. Inversion d'un tableau

Compléter le programme suivant pour inverser l'ordre des éléments du tableau *en place*, c'est-à-dire sans utiliser de second tableau. Le tableau affiché à la fin doit donc être 7 6 5 4 3 2 1.

```
1 #include <stdio.h>
2
3 int main(void) {
4     int tab[7] = {1, 2, 3, 4, 5, 6, 7};
5
6     /* afficher le tableau initial en reutilisant la boucle
7      de la question 1 */
8     int i = 0;
9
10    /* A COMPLETER : inversion du tableau */
11
12
13    /* afficher le tableau final en reutilisant la meme boucle */
14
15    return 0;
16 }
```

Quelle est la complexité temporelle de cet algorithme en fonction de la taille du tableau ?

Question 5. Tableau trié ou non

Compléter le programme afin de déterminer si le tableau est trié par ordre croissant.

La variable `est_trie` vaudra 1 si le tableau est trié et 0 sinon. On souhaite arrêter le parcours dès que deux éléments consécutifs ne sont pas dans le bon ordre.

```
1 #include <stdio.h>
2
3 int main(void) {
4     int tab[7] = {1, 2, 2, 4, 7, 9, 12};
5
6     /* afficher le tableau en reutilisant la boucle de la question 1 */
7     int i = 0;
8
9     int est_trie = 1;
10
11    /* A COMPLETER : test */
12
13 }
```

```

14  /* affichage du resultat */
15  if (est_trie == 1) {
16      printf("le tableau est trie\n");
17  }
18  else {
19      printf("le tableau n'est pas trie\n");
20  }
21
22  return 0;
23 }

```

Modifier ensuite une valeur du tableau pour vérifier que le programme détecte aussi un tableau non trié. Donner la complexité temporelle dans le meilleur cas puis dans le pire cas.

Question 6. Dutch

dans cet exercice on considère un tableau constitué uniquement de 0, 1 et 2 comme {1,2,0,2,2,1,0,1}. On veut trier le tableau de sorte qu'il représente un tableau hollandais (3 couleurs). Avec l'exemple donné, on veut obtenir {0,0,1,1,1,2,2,2}.

Cet exercice peut se faire en 2 temps :

- Production naïve d'une code réalisant l'opération sans soucis de complexité. On peut utiliser un tableau auxiliaire.
- production d'un code en un seul tour de boucle et sans tableau auxiliaire.

```

1  #include <stdio.h>
2
3  int main(void) {
4      int tab[8] = {1,2,0,2,2,1,0,1} ;
5
6      int n=8; // n : longueur du tableau.
7
8      /* afficher le tableau en réutilisant la boucle de la question 1 */
9      int i = 0;
10
11     int est_trie = 1;
12
13     /* A COMPLETER : trier le tableau */
14
15
16     /* affichage du tableau */
17 }

```