

Le classement du Tour de France

Dans ce TP, on se propose de simuler la mise à jour du classement général du tour du France cycliste. Il est tout d'abord nécessaire de représenter un coureur cycliste de façon informatique. Un coureur sera donc constitué de son nom, de son prénom, de son numéro de dossard, de son équipe et du temps cumulé pour faire les étapes depuis le départ du tour. Le tableau ci-dessous vous représente ces informations. Le classement général sera bien sûr établi en fonction de ce temps cumulé.

Nom	Prénom	Dossard	Equipe	Temps cumulé
Hinault	Bernard	51	BON	5h30m15s
Amstrong	Lance	103	GOOD	5h45m10s
Monteil	Thierry	12	CUIT	50h20m2s

1 Gestion d'un coureur

Dans un premier temps, on se propose de créer une structure permettant de représenter un coureur et un certain nombre de fonctions qui seront utiles.

1. Créer un fichier `coureur.h` contenant la structure `coureur` permettant de représenter un coureur (le temps cumulé sera stocké en secondes sous forme d'un entier).
2. Créer un fichier `coureur.c`. Créer une fonction `Creer_Coureur` qui permet d'initialiser une structure `coureur` à partir des caractéristiques du coureur. La structure `coureur` sera allouée en mémoire dans cette fonction qui renverra l'adresse de la structure allouée.
3. Créer une fonction `Ajouter_Temps` qui permet d'ajouter au temps cumulé d'un coureur le temps de l'étape du jour.

4. Créer une fonction **Afficher_Coureur** permettant d'afficher correctement les caractéristiques d'un coureur (le temps cumulé notamment sera affiché sous forme d'heures, minutes et secondes).
5. Actualiser le fichier **coureur.h** pour qu'il contienne la déclaration des fonctions précédentes.
6. Créer éventuellement un petit programme de test pour valider l'ensemble des fonctions .

Si vous l'estimez utile, vous pouvez bien sûr créer d'autres fonctions.

<code>coureur.h</code>	<code>coureur.c</code>
-----	-----
<code>struct coureur{</code>	<code>... Creer_Coureur(...)</code>
<code>... nom;</code>	<code>{</code>
<code>... prenom;</code>	<code>....</code>
<code>... ;</code>	<code>}</code>
<code>};</code>	
<code>... Creer_Coureur(...);</code>	<code>... Ajouter_Temps(...)</code>
<code>... Ajouter_Temps(...);</code>	<code>{</code>
<code>... Afficher_Coureur(...);</code>	<code>...</code>
<code>...</code>	<code>}</code>
	<code>... Afficher_Coureur(...)</code>
	<code>{</code>
	<code>...</code>
	<code>}</code>
	<code>...</code>

2 Gestion du classement général

On se propose de représenter le classement général sous forme d'une liste. Les coureurs ayant été créés par ailleurs, seules des références de coureurs seront gérées dans cette liste.

Par ailleurs, étant donné que l'on aura à parcourir la liste itérativement pour modifier chacun des coureurs, il est conseillé d'inclure dans la structure

un pointeur ”*courant*” permettant de se déplacer dans la liste et de conserver l’emplacement où l’on se trouve. De la même façon que pour un coureur, on créera un fichier `liste.h` et `liste.c`.

1. Créer une structure permettant de représenter une liste de coureurs.
2. Créer une fonction qui permet d’initialiser une liste.
3. Créer une fonction qui permet d’ajouter un coureur en fin de liste.
4. Créer un ensemble de fonctions permettant de parcourir la liste (ces fonctions nous seront nécessaires pour modifier le temps cumulé de chaque coureur à chaque étape). Par exemple, on peut imaginer utiliser la fonction `Aller_Debut` qui permet de se positionner au début de la liste, `Avancer` qui permet d’avancer dans la liste et `Fin` qui permet de tester si on est à la fin de la liste.
5. Créer une fonction qui permet de retourner le coureur qui est pointé par le pointeur courant.
6. Créer une fonction qui permet d’effacer un coureur dans la liste. On effacera le coureur pointé par le pointeur courant.
7. Créer une fonction qui permet de trier la liste pour obtenir le classement général. Ce tri se fera de façon à mettre en tête de liste le coureur ayant réussi le meilleur temps cumulé. Celui qui est en fin de liste a réussi le plus mauvais temps cumulé. L’algorithme de tri sera le tri-bulle.
8. Créer une fonction qui permet d’afficher le contenu de la liste.

Si vous l’estimez utile, vous pouvez bien sûr créer d’autres fonctions.

3 Programme principal

Créer ensuite un programme de test.

En particulier, on initialisera les coureurs à partir d’un fichier qui contiendra :

- Le nombre d’étapes
- Le nombre d’équipes
- Pour chaque équipe, une ligne comprenant le nom de l’équipe, suivie de 5 lignes décrivant chaque coureur à l’aide du format suivant : Numéro de dossard, Nom, Prénom. **Il y a forcément 5 coureurs par équipe.**

A chaque itération (étape), on parcourra la liste de façon à mettre à jour le temps cumulé de chaque coureur (le temps de l'étape sera par exemple calculé à l'aide de fonctions aléatoires). On triera ensuite la liste de façon à obtenir le classement général à chaque étape.

Un coureur peut être éliminé pour dopage. Une sous-fonction du programme principale indiquera (en utilisant des fonctions aléatoires) si un coureur est dopé ou pas.

Les fonctions aléatoires utilisées seront par exemple les fonctions `srandom` et `random`. De plus amples informations seront fournies en TP par les enseignants.

Après avoir affiché le classement général, vous afficherez le classement par équipes suivant les règles suivantes :

- Le classement est un classement au temps cumulé des 3 meilleurs coureurs de l'équipe
- Une équipe ayant moins de 3 coureurs à l'arrivée ne peut pas participer à ce classement