1 TP8 – ENREGISTREMENTS & TABLEAUX

Ce TP reprend à nouveau le thème de la régate abordé en TP. Relativement au TP précédent, la représentation des données reste inchangée. Le but du TP est de manipuler des tableaux d'enregistrement, mais aussi de découvrir la méthode de tri par insertion.

Régate de voiliers (2)
En complément des types de données vus au cours du TP précédent :

- t_jhm {jours, heures, minutes}
- t_voilier {nomSkipper, numVoilier, tempsCourse}

On introduit un nouveau type t_course qui est un tableau d'enregistrements du type t_voilier.

À ces déclarations de types on associe trois constantes : MAX pour le nombre de bateaux en course et MAX_CHAR pour la longueur maximum du nom du skipper.

Préparation

1. Préparer la déclaration des constantes MAX et MAX_CHAR.
2. Préparer la déclaration de 't_course' en reprenant les types déclarés dans le TP précédent.
3. Préparer un jeu de valeurs de test et compléter le tableau ci-dessous.

Numéro	**Skipper**	**Temps de course J, H, M**	**Durée minutes**

4. Donner la vue externe et écrire l'algorithme du sous-programme ajouterVoilier() qui permet de saisir les différents champs d'un voilier voilier et de l'insérer à la fin d'un tableau course. On fera appel au sous-programme saisirVoilier() vu précédemment. Attention : le temps de la course sera lu sous la forme <Jours, Heures, Minutes>. On utilisera à cet effet le sous-programme saisirJHM() et convJHMmin()
5. Donner la vue externe et écrire l'algorithme du sous-programme afficherVoilier() qui permet d'afficher les différents champs d'un voilier voilier.
   Attention : le temps de la course sera affiché sous la forme : TotalMinutes (JJ Jours, HH Heures, MM Minutes). On utilisera à cet effet le sous-programme convminJHM() et afficherJHM().
6. Donner la vue externe et écrire l'algorithme du sous-programme afficherVoiliers() qui permet d'afficher tous les voiliers participants à une course course. Ce sous-programme fera appel au sous-programme afficherVoilier().
7. Donner la vue externe et écrire l'algorithme du sous-programme penaliserVoilier() qui permet d'ajouter un nombre de minutes de pénalité minPenalite à un voilier

spécifié par son numéro numVoilier. Les valeurs minPenalite et numVoilier sont passées en paramètres. On fera appel au sous-programme ajouterPenalite() vu dans le TP précédent.

Réalisation en séance

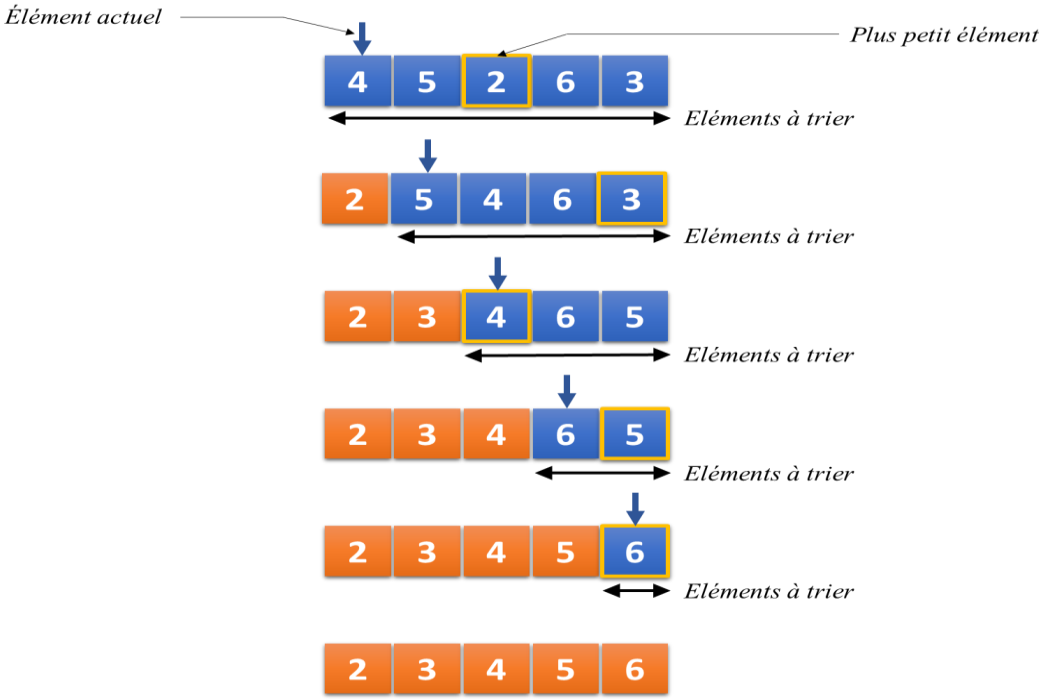
1. Ouvrir, dans **Code::Blocks**, un nouveau fichier source regate2.cpp dans le même projet.
2. Traduire l'algorithme en C++ de ajouterVoilier(), afficherVoilier(), et afficherVoiliers(). Réalisez le programme principal qui permet de les tester.
3. Traduire l'algorithme en C++ de penaliserVoilier(). Écrire le programme principal qui permet de les tester.
4. Reprendre le menu du programme principal du TP précédent et le modifier pour proposer les options suivantes et appeler les sous–programmes développés en fonction du choix de l'utilisateur:

```
Voulez-vous:
(1) Inscrire un nouveau voilier (0 déjà inscrits)
(2) Afficher la liste des voiliers
(3) Ajouter une pénalité à un voilier
(4) Quitter
Votre choix ? _
```

Classement

On propose de trier les différents voiliers en fonction de leurs temps de course par ordre croissant. Pour ce faire, on utilisera l'algorithme de tri par insertion.

L'algorithme de tri par insertion sur un tableau de N éléments se déroule en N étapes. A la $i^{ème}$ étape, on classe le $i^{ème}$ élément à sa bonne place dans le tableau, c'est-à-dire, à la $i^{ème}$ place.



Préparation

1. Donner la vue externe et écrire l'algorithme du sous-programme rangMinTempsCourse() qui retourne le rang dans le tableau course du voilier ayant le plus petit temps de course entre deux indices du tableau deb et fin spécifiés en paramètres.
2. Donner la vue externe et écrire l'algorithme du sous-programme permuterVoiliers() qui permet de permuter deux voiliers dans le tableau course, spécifiés par leurs indices respectifs voilierA et voilierB.
3. Donner la vue externe et écrire l'algorithme du sous-programme classerVoiliers() qui permet de classer les voiliers d'un tableau course selon un ordre croissant de leurs temps de course. Ce sous-programme fera appel aux deux sous-programmes rangMinTempsCourse() et permuterVoiliers().

Réalisation en séance

1. Ouvrir, dans Code::Blocks, le fichier source regate2.cpp.
2. Traduire en C++ l'algorithme de rangMinTempsCourse() et permuterVoiliers(), puis le sous-programme classerVoiliers() et les tester.
3. Modifier le sous-programme afficherVoiliers() pour qu'il affiche la liste des voiliers triés à chaque appel.