UNIVERSITÉ DE NANTES

Projet de développement d'une application en langage COBOL

Présentation du logiciel Tournamenthe

Matthieu Gonzalez
Clément Leclercq

Notre société COBOLPERFORM est spécialisée dans l'édition de logiciel de gestion d'évènement tel que des tournois sportifs ou des festivals.

La fédération française de jeu d'échec a fait appel à nos services afin de concevoir un logiciel fiable mais simple d'utilisation appelé Tournamenthe.
Cet outil sera utilisé pour gérer une saison de tournois d'échec.

Tournamenthe permettra aux membres des associations de créer un tournoi dans leurs locaux en renseignant la date, le lieu, le nombre de participants, le nombre de jeux disponibles, la durée du tournoi...
Ils pourront renseigner les informations concernants les jeux, le nom, le nombre de joueur, la difficultés et la durée.

Lors d'un tournoi, les organisateurs devront remplir une fiche pour chaque joueur comprenant son prénom, son nom, son adresse, son âge et son niveau de joueur.
Les fichiers participants seront mis à jours au cours du tournoi avec les différentes manches, ils comprendront les noms des joueurs, le nom du jeu auquel ils participent ainsi que la manche dans laquelle ils sont rendus.

<u>Le logiciel « Tournamenthe » utilisera donc les fichiers suivants :</u>

1 fichier indexé : “tournoi.dat”

L’enregistrement sert à renseigner les informations d’un tournoi.
Il ne peut y avoir deux tournois organisés la même semaine.
Un tournoi est organisé avec un type de partie. Un joueur ne peut s’inscrire à un tournoi que de si son niveau correspond au rang du tournoi(comparaison avec nbpoint d’un joueur).
Ainsi les tournois de rang 1 sont ouvert à tous les joueurs, les tournois de rang 2 aux joueurs ayant plus de 500 points, les tournois de rang 3 aux joueurs ayant plus de 1000 points.

type Ttournoi : enregistrement
fto_nomasso : cdc // nom de l’association organisatrice
fto_id : entier
fto_semaine: entier
fto_ville : cdc
fto_nbplaces : entier // En fonction du rang du tournoi
// 32 pour ouvert à tous
fto_ntours: entier // log à base 2 du nb de places
fto_typepartie: id de type partie
fto_rangtournoi : entier: 1 ouvert à tous / 2 Restreint / 3 Elite
finEnregistrement

CP : fto_id
CSAD: fto_typepartie,fto_rangtournoi

Fonctionnalités :

1 / Ajouter_tournoi : qui permet d'ajouter un tournoi dans le fichier de type Ftournoi. Un seul tournoi pour une association donnée et un tournoi par semaine.

2/ Modifier Tournoi

3/ Supprimer tournoi

4/ Recherche fichier tournoi :

- Recherche des tournoi organisés par une association
- Recherche d'un tournoi avec un ID
- Recherche des tournois organisés pour une semaine donnée
- Recherche des tournois organisés pour une ville donnée
- Recherche des tournois organisés selon le nb de place.
- Recherche des tournois organisés selon le type de partie. .
- Recherche des tournois organisés selon le rang du tournoi.

5/ Recherche_semaine : affiche la ville et le nom de l'association qui organise le tournoi d'une semaine donnée.

1 fichier indexé : “joueur.dat”

Enregistrement qui sert à stocker les informations relatives aux joueurs. Le classement est calculé en fonction du nombre de points.Un joueur est caractérisé par un numéro unique : id, son prénom, son nom, son âge, la ville d’origine du joueur ainsi que son classement auprès de la fédération.

type Tjoueur : enregistrement
fjo_id : entier // identifie de manière unique le joueur
fjo_nom : cdc
fjo_prenom : cdc
fjo_age : int
fjo_ville : cdc
fjo_points : entier
finEnregistrement

Clé primaire : **fjo_id**
Clé secondaire avec doublon : **fjo_ville, fjo_points**

Fonctionnalités :

1 / Ajouter_joueur : permet d'ajouter un joueur dans le fichier de type Fjoueur. Attention de vérifier que le numéro de joueur n'existe pas déjà.

2/ Modifier joueur

3/ Supprimer joueur

4/ Recherche fichier joueur :
-Recherche d'un joueur selon son ID.
-Recherche d'un joueur selon son nom et prénom.
-Recherche de joueur selon leur âge
-Recherche de joueur selon leur ville.
-Recherche de joueur selon un nombre de point.

5 / Recherche_joueur : permet de récupérer l'identifiant d'un joueur sachant son nom et son prénom.

6/ Max_victoire : permet d'afficher le joueur ayant remporté le plus de tournoi.

1 fichier indexé : “partie.dat”
Cet enregistrement sert à stocker les informations sur le type de partie d’un tournoi. Une partie sera identifiée par un numéro de partie, le temps que dure la partie au maximum, un type en fonction du type d’ouverture de la partie.

type Tpartie : enregistrement
fpa_numP: entier
fpa_temps : entier
fpa_type : cdc // type d’ouverture
finEnregistrement

Clé primaire : **fpa_numP**
Clé secondaire avec doublon : **fpa_type, fpa_temps**

Fonctionnalités :

1/ Ajout_partie : permet d'ajouter une partie dans le fichier de type Fpartie. Nous vérifions que le type de partie n'a pas déjà été créé.

2/ Modifier partie

3/ Supprimer partie

4/ Recherche fichier partie :

- Recherche d'une partie en fonction du numéro de partie
- Recherche d'une partie selon le temps
- Recherche d'une partie selon le type d'ouverture.

5/ Recherche_ouverture : retourne le numéro des parties données (d'un certain type) dans un tournoi donné.

1 fichier indexé : “déroulement.dat”
Ce fichier détermine l’avancement et le résultat d’une partie. Il est caractérisé par le numéro de chaque joueur (idjoueur1 et idjoueur2), le numéro du tournoi, le tour de la partie dans le tournoi (huitième, quart, demi, finale, ainsi que l’id du joueur vainqueur.

Le fichier conserve le résultat de chaque partie jouée au cours d’un tournoi entre deux joueurs distincts. Il permet de déterminer le vainqueur de la partie.
Une partie gagné d’un tournoi de rang 1 rapporte 100 points.
Une partie gagné d’un tournoi de rang 2 rapporte 50 points.
Une partie gagné d’un tournoi de rang 3 rapporte 25 points.
Le vainqueur d’un tournoi gagnera en plus un nombre de points égales à celui qui est attribué lors d’une victoire sur une partie du même rang.
Ces points seront ajoutés au points du joueur gagnant dans le fichier joueur.dat

```
type Tdéroulement : enregistrement
        fp_cle
                fder_idjoueur1 : entier
                fder_idjoueur2: entier
                fder_idtournoi : entier
        fder_tour : entier // détermine le tour du tournoi
        fder_idvainqueur: entier // détermine le vainqueur de la partie
finEnregistrement
```

Clé primaire : **fder_idjoueur1, fder_idjoueur2, idtournoi**
Clé secondaire avec doublon : **fder_tour**

Fonctionnalités :

1/ Ajout_deroul : permet d'ajouter les informations de déroulement de chaque partie du tournoi et d'ajouter le nombre de points au vainqueur de la partie.

2/ Modifier déroulement

3/ Supprimer déroulement

4/ Recherche fichier déroulement :

- Recherche d'un déroulement en fonction des clé primaire.
- Recherche des déroulements selon un tour.
- Recherche d'un déroulement selon idvainqueur.

5/ Recherche_adversaire : permet d'afficher la liste des adversaires qu'à rencontré un joueur donné en paramètre lors d'un tournoi et le type d'ouverture utilisée.

Algo complexe Max_duree :

Recherche du tournoi le plus long organisé par une association rentrée en paramètre, et affichage du gagnant de ce tournoi.

Analyse:
Nous avons besoin du fichier Ftournoi pour connaitre le nombre de tours et le nom de l'association organisatrice, du fichier Fpartie pour récupérer la durée max par joueur et du fichier Fderoulement afin de récupérer le nom du vainqueur

On doit commencer par Ftournoi car la recherche est sur le nom de l'association organisatrice.

En commençant par Tournoi:

La meilleure solution consiste à faire une lecture séquentielle sur Ftournoi, quand nomAsso = fto_nomAsso on fait une recherche directe sur Fpartie pour récupérer le temps max par joueur, on le multiplie par deux puis pas le nombre de tour, et on ne garde le résultat que si il est supérieur à max. On fais ensuite une lecture sequentielle du fichier Fderoulement jusqu'à ce que l'on trouve le vainqueur du tournoi.

Le coût de la procédure :

Lecture séquentielle de Ftournoi = T
Lecture direct de Fpartie = T
Lecture séquentielle de Fderoulement = D

Total = 2T + D

Algo complexe Max_victoire :

Recherche du joueur(nom, prénom etc..) qui a remporté le plus de tournois:

Analyse:
Nous avons besoin du fichier Ftournoi pour connaître le nombre de tour et l'id du tournoi, du fichier Fjoueur pour connaitre les informations du joueur et de Fdéroulement pour connaître le vainqueur d'un tournoi.

On ne peut commencer par Déroulement car on doit connaître le nombre de tours du tournoi pour déterminer le vainqueur de manière raisonnable.

En commençant par Tournoi:

On peut récupérer le vainqueur d'un tournoi mais il est difficile de sauvegarder le nombre de victoires pour chaque joueur.

La meilleure solution consiste à commencer par joueur :

1) Lecture séquentielle de joueur: on sauvegarde l'id du joueur en cours de lecture
2) Lecture séquentielle de tournoi: Pour chaque joueur on lit tout les tournois et on regarde le vainqueur de ce tournoi en faisant une lecture séquentielle de déroulement, si le vainqueur du tournoi est égale au joueur courant on incrémente un compteur.

Le coût de la procédure :

Lecture séquentielle de joueur = J

Lecture séquentielle de tournoi pour chaque joueur = J x T

Lecture séquentielle de déroulement pour chaque tournoi pour chaque joueur = (J x T) x D

Total = J x (J×T)× ((J×T)×D)

Algo complexe Recherche_adversaire :

Affiche les adversaires(nom, prénom) qu'a rencontré un joueur lors d'un tournoi dont l'id est donné en paramètre et affiche le type d'ouverture pour chaque partie.

Analyse:

Nous avons besoin du fichier Fderoulement pour connaître l'id du joueur 1 et l'id du joueur 2 et du fichier Fjoueur pour connaitre les informations des joueurs (prénom, nom..). Nous avons également besoin du fichier Fpartie pour récupérer le type d'ouverture utilisée.

En commençant par le fichier Fjoueur :

1) Lecture directe de joueur, grâce au nom et prénom on récupère l'id du joueur.
2) Lecture séquentielle de Fderoulement, grâce à l'id du joueur et à l'id du tournoi donné en paramètre on trouve l'id du joueur2 et le numéro de la partie.
3) Lecture directe au informations du joueur2 grâce à son id.
4) Lecture directe au type de la partie grâce à numP.

Le coût de la procédure :

Lecture directe de joueur = J
Lecture séquentielle de déroulement = D
Lecture directe aux infos du joueur2 = D x J
Lecture directe au type de partie = D x P

Coût de la procédure :

J + D + (D x J) + (D x P)

Organisation du développement du projet :

Jalons :

- Rédaction du cahier des charges final : **Deadline 2 mars**
- Mise en place de l'environnement de travail (github, structure du programme) : **Deadline 2 mars**
- Développement des procédures de “base” : **Deadline 11 mars**
- Mise en commun, test et débogage du logiciel : **13 mars** (TP)
- Développement des procédures “complexes” : **Deadline 21 mars**
- Test, débogage, robustesse : **Deadline 23 mars**
- Soutenance : **Mardi 27 mars.**

Répartition du travail :

Clément :

- Procédure Ajout_deroul / Modifier déroulement / Supprimer déroulement / Recherche déroulement.
- Recherche_adversaire autour du fichier Fdéroulement
- Rédaction cahier des charges.

Matthieu :

- Procédure Ajouter_joueur / Modifier joueur / Supprimer Joueur / Recherche
- Max_victoire autour du fichier Fjoueur
- Rédaction cahier des charges.

Hippolyte :

- Procédure Ajouter_tournoi / Modifier tournoi / Supprimer tournoi / Recherche tournoi
- Recherche_semaine autour du fichier Ftournoi,
- Max_Durée

Mamadou :

- Procédure Ajout_partie / Modifier partie / Supprimer partie / Recherche partie
- Recherche_ouverture autour du fichier Fpartie.