

Rapport du premier projet d'IAP

Groupe 101 et 103

Noé GODIN
Louis MASSON

Table des matières

1) Présentation de l'application	page 3
2) Organisation des tests	page 4
3) Bilan de validation des tests de développement	page 5-8
4) Bilan du projet	page 9
5) Code C complet (annexe s)	page 10-13

1) Présentation de l'application

Cette application a pour objectif de mémoriser et de manipuler jusqu'à dix tournois de tennis féminin. Il fonctionne grâce à des lignes de commande en langage C et peut :

- enregistrer les joueuses participant à des tournois donnés
- afficher le classement d'un ou plusieurs tournois et le nombre de points gagnés par chaque joueuse
- rechercher les matchs précis d'une joueuse particulière dans un tournoi particulier

2) Organisation des tests

Afin de vérifier la validité de nos sprints, nous avons testé chacun d'entre eux avec des JDT et par la suite nous avons comparé nos résultats aux résultats attendus à l'aide du site diffchecker.

Ce site met en évidence les différences entre deux fichiers texte. Ainsi nous avons pu vérifier que notre application fonctionne correctement.

3) Bilan de validation des tests de développement

Ce test vérifie si on peut enregistrer et afficher un tournoi précis. Il valide le Sprint 1.

The two files are identical

Editor ▾ Compare & merge Clear ↔ Export as PDF Save Diff Share

Original Text

Changed Text

123 Pavlyuchenkova Azarenka
124 Zidansek Cirstea
125 Badosa Vondrousova
126 quarts de finale
127 Krejcikova Gauff
128 Sakkari Swiatek
129 Pavlyuchenkova Rybakina
130 Zidansek Badosa
131 demi-finales
132 Krejcikova Sakkari
133 Pavlyuchenkova Zidansek
134 finale
135 Krejcikova Pavlyuchenkova
136

123 Pavlyuchenkova Azarenka
124 Zidansek Cirstea
125 Badosa Vondrousova
126 quarts de finale
127 Krejcikova Gauff
128 Sakkari Swiatek
129 Pavlyuchenkova Rybakina
130 Zidansek Badosa
131 demi-finales
132 Krejcikova Sakkari
133 Pavlyuchenkova Zidansek
134 finale
135 Krejcikova Pavlyuchenkova
136

3) Bilan de validation des tests de développement

Ce test vérifie si on peut afficher une joueuse d'un tournoi précis et ses matchs. Il valide le Sprint 2.

The two files are identical

Editor ▾ Compare & merge Clear ↔ Export as PDF Save Diff Share

Original Text

Changed Text

```
1 Badosa Davis
2 Badosa Kovinic
3 Badosa Bogdan
4 Badosa Vondrousova
5 Zidansek Badosa
6
```

```
1 Badosa Davis
2 Badosa Kovinic
3 Badosa Bogdan
4 Badosa Vondrousova
5 Zidansek Badosa
6
```

3) Bilan de validation des tests de développement

Ce test vérifie si on peut afficher un tournoi précis et le score de chaque joueuse dans l'ordre alphabétique. Il valide le Sprint 3.

The two files are identical

Editor ▾ Compare & merge Clear ↔ Export as PDF Save Diff Share

Original Text

Changed Text

117 Uytvanck 10
118 VWilliams 10
119 Vekic 10
120 Vesnina 90
121 Voegele 10
122 Vondrousova 180
123 Watson 10
124 XWang 10
125 Zavatska 10
126 Zhang 10
127 Zheng 45
128 Zhu 10
129 Zidansek 720
130

117 Uytvanck 10
118 VWilliams 10
119 Vekic 10
120 Vesnina 90
121 Voegele 10
122 Vondrousova 180
123 Watson 10
124 XWang 10
125 Zavatska 10
126 Zhang 10
127 Zheng 45
128 Zhu 10
129 Zidansek 720
130

3) Bilan de validation des tests de développement

Ce test vérifie si on peut afficher chaque tournoi et le cumul de points de chaque joueuse. Il valide le Sprint 4.

The two files are identical

Editor ▾ Compare & merge Clear ↔ Export as PDF Save Diff Share

Original Text

Changed Text

143 Paquet 10
144 Parry 10
145 Perez 10
146 Pironkova 10
147 Riske 10
148 Sanders 10
149 Schmiedlova 10
150 Stosur 10
151 Swan 10
152 Teichmann 10
153 Voegele 10
154 Volynets 10
155 YWang 10
156 Zavatska 10
157

143 Paquet 10
144 Parry 10
145 Perez 10
146 Pironkova 10
147 Riske 10
148 Sanders 10
149 Schmiedlova 10
150 Stosur 10
151 Swan 10
152 Teichmann 10
153 Voegele 10
154 Volynets 10
155 YWang 10
156 Zavatska 10
157

4) Bilan de projet

Ce projet nous a fait énormément progresser en C.

Il nous a permis d'approfondir nos connaissances des types structurés, des tableaux et des pointeurs, et d'utiliser ces types de données.

Il a également mis en évidence l'importance des fonctions dans un programme, permettant l'organisation et l'optimisation du code.

Nous avons eu des difficultés pour comprendre le sprint 5 et nous n'avons pas eu assez de temps ensemble pour le terminer. Notre programme pourrait être amélioré si l'on modifiait la fonction "afficher_classement" de façon à valider le Sprint 5.

En conclusion, cet exercice nous a montré à quel point il faut être attentif à chaque étape de la conceptualisation du code, du fonctionnement et des sorties de programme.

5) Codes source

Les codes source sont disponibles dans le dossier Annexe.

Le dernier fichier (source.c) est entièrement commenté.

Code C I.1 à I.58

```
1 //sprint 1 à 4
2 #include <stdlib.h>
3 #include <stdio.h>
4 #include <time.h>
5 #include <string.h>
6 #pragma warning (disable : 4996)
7
8 #define maxTournois 10
9 #define nbMatchTournoi 127
10 #define nbJoueusesTournoi 128
11 #define longueurMot 30
12
13 typedef struct {
14     char nom[longueurMot + 1];
15     unsigned int points;
16 }Joueuse;
17
18 typedef struct {
19     unsigned int idxGagnante;
20     unsigned int idxPerdante;
21 }Match;
22
23 typedef struct {
24     char nom[longueurMot + 1];
25     int date;
26     Match dataMatch[nbMatchTournoi];
27 }Tournoi;
28
29 typedef struct {
30     Tournoi dataTournois[maxTournois];
31     Joueuse dataJoueuse[maxTournois * nbJoueusesTournoi];
32     unsigned int nbTournois;
33     unsigned int idxJ;
34     unsigned int idxT;
35 }TournoiWTA;
36
37 //prototypage des fonctions utilisees et documentation
38
39 /*
40 * demande le nombre de tournoi avec lequel on travaille
41 * [in] tableau de type tournoiWTA
42 */
43 void definir_nombre_tournois(TournoiWTA* ins);
44
45 /*
46 * regarde si le tournoi recherché existe
47 * [in] tableau de type tournoiWTA
48 * [in] tableau de type char
49 * [in] entier
50 * [retour] entier
51 */
52 int ExistenceTournoi(const TournoiWTA* ins, const char ville[], const int date);
53
54 /*
55 * enregistre un tournoi spécifié
56 * [in] tableau de type tournoiWTA
57 */
58 void enregistrement_tournoi(TournoiWTA* ins);
59
```

Code C I.60 à 116

```
60 /*
61 * inscrit ou retrouve l'index d'une joueuse
62 * [in] tableau de type tournoiWTA
63 * [in] tableau de type char (nom de la joueuse)
64 * [retour] entier (index joueuse)
65 */
66 int referenceID(TournoiWTA* ins, char joueuse[]);
67
68 /*
69 * Ajoute des points aux joueuses dans le tournoi
70 * [in] tableau de type tournoiWTA
71 */
72 void attributionpoints(TournoiWTA* ins);
73
74 /*
75 * Affiche un tournoi spécifié
76 * [in] tableau de type tournoiWTA
77 */
78 void affichage_matches_tournoi(const TournoiWTA* ins);
79
80 /*
81 * Affiche les matchs d'une joueuse
82 * [in] tableau de type tournoiWTA
83 */
84 void afficher_matches_joueuse(const TournoiWTA* ins);
85
86 /*
87 * Affiche le score de chaque joueuse dans un tournoi donné
88 * [in] tableau de type tournoiWTA
89 */
90 void affichage_joueuses_tournoi(TournoiWTA* ins);
91
92 /*
93 * Affiche le cumul de points de chaque joueuse
94 * [in] tableau de type tournoiWTA
95 */
96 void afficher_classement(TournoiWTA* ins);
97
98 void definir_nombre_tournois(TournoiWTA* ins) {
99     int x;
100     scanf("%d", &x);
101     if ((x >= 1) && (x <= 10)) { //on verifie si il y a bien un tournoi compris entre 1 et 10 compris on ne pourra pas enregistrer moins de 1 tournoi et pas plus de 10
102         ins->nbTournois = x; //écrit le nombre de tournoi dans la structure tournoiWTA
103     }
104 }
105
106 int ExistenceTournoi(const TournoiWTA* ins, const char ville[], const int date) {
107
108     for (int i = 0; i < ins->idxT; ++i) {
109         if ((strcmp(ins->dataTournois[i].nom, ville) == 0) && (ins->dataTournois[i].date == date)) {
110             return i; //renvoi le tournoi ayant le même nom / date que celui demandé sinon renvoi -1
111         }
112     }
113     printf("tournoi inconnu");
114     return -1;
115 }
116
```

Code C I.127 à 175

```

117 void enregistrement_tournoi(TournoiTA* ins) {
118     char ville[longueurMot + 1];
119     int date;
120     scanf("%s", ville);
121     scanf("%d", &date);
122     strcpy(ins->dataTournois[ins->idxT].nom, ville);
123     ins->dataTournois[ins->idxT].date = date;
124
125     char joueurGagne[longueurMot + 1];
126     char joueurPerd[longueurMot + 1];
127
128     for (int countMatches = 0; countMatches < nbMatchTournoi; ++countMatches) {
129         scanf("%s", joueurGagne); //Lorsqu'on enregistre, la premiere joueur (a gauche) est toujours gagnante
130         scanf("%s", joueurPerd); //A l'inverse, la deuxieme joueur (a droite) est toujours perdante
131
132         ins->dataTournois[ins->idxT].dataMatch[countMatches].idxGagnante = referenceID(ins, joueurGagne); //on recupere son index
133         ins->dataTournois[ins->idxT].dataMatch[countMatches].idxPerdante = referenceID(ins, joueurPerd); //on recupere son index
134     }
135     attributionpoints(ins); //on leur attributs leurs points
136     ins->idxT ++;
137 }
138
139 void attributionpoints(TournoiTA* ins) {
140     for (int i = 0; i < nbMatchTournoi; ++i) //boucle sur tous les matchs pour ajouter des points aux perdants
141     if (i >= 0 && i < 64) ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 10;
142     else if (i >= 64 && i < 96) {
143         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 45;
144     }
145     else if (i >= 96 && i < 112) {
146         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 90;
147     }
148     else if (i >= 112 && i < 128) {
149         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 180;
150     }
151     else if (i >= 128 && i < 124) {
152         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 360;
153     }
154     else if (i >= 124 && i < 126) {
155         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 720;
156     }
157     if (i == 126) {
158         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxGagnante].points += 2080; //ajout des points a la gagnante
159         ins->dataJoueur[ins->dataTournois[ins->idxT].dataMatch[i].idxPerdante].points += 1280;
160     }
161 }
162
163 int referenceID(TournoiTA* ins, char joueur[]) {
164     for (int i = 0; i < ins->idxJ; ++i) { //pour toutes les joueuses déjà inscrite
165         if (strcmp(ins->dataJoueur[i].nom, joueur) == 0) {
166             return i; //test qui renvoi i si joueur est dans le tableau
167         }
168     }
169     strcpy(ins->dataJoueur[ins->idxJ].nom, joueur); //si le nom n'est pas dans le tableau, alors on le met dedans
170     ins->dataJoueur[ins->idxJ].points = 0; //initialisation des points des nouvelles joueuses à 0
171     ins->idxJ ++;
172
173     return ins->idxJ - 1; //renvoi l'index de notre joueur
174 }
175

```

Code C I.177 à 234

```

177 void affichage_matches_tournoi(const TournoiTA* ins) {
178     char ville[longueurMot + 1];
179     int date;
180     scanf("%s", ville);
181     scanf("%d", &date);
182     int t = ExistenceTournoi(ins, ville, date);
183     if (t == -1) {
184         return;
185     }
186     printf("%s %d\n", ins->dataTournois[t].nom, ins->dataTournois[t].date); //écriture du nom du tournoi et de sa date
187     for (int j = 0; j < nbMatchTournoi; ++j) {
188         if (j == 0) printf("64emes de finale\n");
189         else if (j == 64) {
190             printf("32emes de finale\n");
191         }
192         else if (j == 96) {
193             printf("16emes de finale\n");
194         }
195         else if (j == 112) {
196             printf("8emes de finale\n");
197         }
198         else if (j == 120) {
199             printf("quarts de finale\n");
200         }
201         else if (j == 124) {
202             printf("demi-finales\n");
203         }
204         else if (j == 126) {
205             printf("finale\n");
206         }
207         printf("%s ", ins->dataJoueur[ins->dataTournois[t].dataMatch[j].idxGagnante].nom); //affichage de la gagnante
208         printf("%s\n", ins->dataJoueur[ins->dataTournois[t].dataMatch[j].idxPerdante].nom); //affichage de la perdante
209     }
210 }
211
212 void afficher_matches_joueur(const TournoiTA* ins) {
213     char ville[longueurMot + 1];
214     int date;
215     scanf("%s", ville);
216     scanf("%d", &date);
217     int t = ExistenceTournoi(ins, ville, date);
218     if (t == -1) {
219         return;
220     }
221     char joueur[longueurMot + 1];
222     scanf("%s", joueur);
223     for (int i = 0; i < ins->idxJ; ++i) {
224         if (strcmp(joueur, ins->dataJoueur[i].nom) == 0) //on regarde si le nom de la joueuse est presente dans un tournoi.
225             for (int j = 0; j < nbMatchTournoi; ++j) {
226                 if ((i == ins->dataTournois[t].dataMatch[j].idxGagnante) || (i == ins->dataTournois[t].dataMatch[j].idxPerdante)) { //on affiche ses matchs
227                     printf("%s ", ins->dataJoueur[ins->dataTournois[t].dataMatch[j].idxGagnante].nom);
228                     printf("%s\n", ins->dataJoueur[ins->dataTournois[t].dataMatch[j].idxPerdante].nom);
229                 }
230             }
231     }
232 }
233

```

Code C l.235 à l.292

```

235 void affichage_joueses_tournoi(Tournoi*TA ins) {
236     char ville[longueurMot + 1];
237     int date, i, Retenu1, Retenu2, points[nbJouesesTournoi], Joueses[nbJouesesTournoi];
238     int notreJouese = 0; //remplace par i
239     scanf("%s", ville);
240     scanf("%d", &date);
241     int t = ExistenceTournoi(ins, ville, date);
242     if (t == -100) {
243         return;
244     }
245     printf("%s %d\n", ins->dataTournoi[t].nom, ins->dataTournoi[t].date);
246     for (i = 0; i < 64; i++) { //dans les 64 premiers matchs les perdantes ont 10 points
247         Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
248         points[notreJouese] = 10;
249         notreJouese++;
250     }
251     for (; i < 96; i++) { //dans les 32 prochains matchs les perdantes ont 45 points (64 + 32 = 96)
252         Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
253         points[notreJouese] = 45;
254         notreJouese++;
255     }
256     for (; i < 112; i++) { //dans les 16 prochains matchs les perdantes ont 90 points
257         Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
258         points[notreJouese] = 90;
259         notreJouese++;
260     }
261     for (; i < 120; i++) { //dans les 8 prochains matchs les perdantes ont 100 points
262         Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
263         points[notreJouese] = 100;
264         notreJouese++;
265     }
266     for (; i < 124; i++) { //dans les 4 prochains matchs les perdantes ont 360 points
267         Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
268         points[notreJouese] = 360;
269         notreJouese++;
270     }
271     for (; i < 126; i++) { //dans les 2 prochains matchs les perdantes ont 720 points
272         Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
273         points[notreJouese] = 720;
274         notreJouese++;
275     }
276     Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxPerdante;
277     points[notreJouese] = 1200; //la perdante de la finale à 1200 points
278     notreJouese++;
279     Joueses[notreJouese] = ins->dataTournoi[t].dataMatch[i].idxGagnante;
280     points[notreJouese] = 2000; //la gagnante du tournoi à 2000 points
281     for (int i = 1; i < nbJouesesTournoi; ++i)
282         if (strcmp(ins->dataJouese[Joueses[i - 1]].nom, ins->dataJouese[Joueses[i]].nom) > 0) {
283             Retenu1 = Joueses[i - 1];
284             Retenu2 = points[i - 1];
285             Joueses[i - 1] = Joueses[i];
286             points[i - 1] = points[i];
287             Joueses[i] = Retenu1;
288             points[i] = Retenu2;
289             i = 0;
290         }
291     for (int i = 0; i < nbJouesesTournoi; ++i)
292         printf("%s %d\n", ins->dataJouese[Joueses[i]].nom, points[i]);

```

Code C l.293 à l.351

```

293 }
294
295 void afficher_classement(Tournoi*TA ins) {
296     if (ins->nbTournois == 0) printf("pas de classement");
297     int tab[nbTournois * nbJouesesTournoi];
298     int i, h, idxGagnante, idxPerdante;
299     unsigned int onRetient, j, elemTableau, Valsuperieur;
300     for (j = 0; j < (ins->idx); ++j)
301         tab[j] = j;
302     for (j = 0; j < (ins->idx); ++j) {
303         Valsuperieur = j;
304         for (elemTableau = j; elemTableau < (ins->idx); ++elemTableau) {
305             if (ins->dataJouese[tab[elemTableau]].points < ins->dataJouese[tab[j]].points) { //si les Joueses ne sont pas trie par leur point de façon décroissante
306                 Valsuperieur = elemTableau;
307             }
308             if (ins->dataJouese[tab[Valsuperieur]].points == ins->dataJouese[tab[j]].points) {
309                 if (strcmp(ins->dataJouese[tab[elemTableau]].nom, ins->dataJouese[tab[Valsuperieur]].nom) < 1) { // si les deux Joueses ont le meme nombre de points mais ne sont pas trie en ordre alphabetique
310                     Valsuperieur = elemTableau;
311                 }
312             }
313         } //trieur
314         onRetient = tab[j];
315         tab[j] = tab[Valsuperieur];
316         tab[Valsuperieur] = onRetient;
317     }
318     for (i = 0; i < (ins->idx); ++i)
319         printf("%s %d\n", ins->dataJouese[tab[i]].nom, ins->dataJouese[tab[i]].points); //affichage des Joueses par ordre de point décroissant puis alphabetique dans tous les tournois
320 }
321
322 int main() {
323     char mot[30];
324     Tournoi*TA ins; //creation du tournoi generale du nom de ins
325     ins->idx = 0; //mise en valeur des indices des Joueses
326     ins->nbTournois = 0; //mise en valeur du nombre de tournoi
327     ins->idxT = 0; //mise en valeur de l'indice des tournois
328     while (1) {
329         scanf("%s", mot);
330         if (strcmp(mot, "exit") == 0) exit(0); //si le mot exit est tape, alors le programme s'arrete
331         else if (strcmp(mot, "definir_nombre_tournois") == 0) //si le mot definir_nombre_tournois est tape, alors la fonction de definition du nombre de tournoi s'exécute
332             definir_nombre_tournois(&ins);
333         else if (strcmp(mot, "enregistrement_tournoi") == 0) //si le mot enregistrement_tournoi est tape, alors la fonction d'enregistrement tournoi s'exécute
334             enregistrement_tournoi(&ins);
335         else if (strcmp(mot, "affichage_matches_tournoi") == 0) //si le mot affichage_matches_tournoi est tape, alors on affiche les matchs et leurs déroulement
336             affichage_matches_tournoi(&ins);
337         else if (strcmp(mot, "afficher_matches_jouese") == 0) //si le mot afficher_matches_jouese est tape, alors on affiche tous les matchs d'une Jouese lors d'un tournoi
338             afficher_matches_jouese(&ins);
339         else if (strcmp(mot, "affichage_joueses_tournoi") == 0) //si le mot affichage_joueses_tournoi est tape, alors un classement de point sera affiché par ordre alphabetique
340             affichage_joueses_tournoi(&ins);
341         else if (strcmp(mot, "afficher_classement") == 0) //si le mot afficher_classement est tape, alors on affiche les points par ordre décroissant et leurs noms par ordre alphabetique si leurs points sont egau
342             afficher_classement(&ins);
343     }
344     system("pause"); return 0;
345 }

```