Langages fonctionnels (INFO406)
Examen – Session 1

L2 informatique, CMI Informatique, Maths Info
Lundi 3 mars 2024 – 15 h 15 à 17 h 15 (+ 1/3 temps)

Aucun dispositif électronique autorisé
(calculette, ordinateur, téléphone portable, montre connectée, ...)

Remarques :

- *Le sujet comporte 4 pages.*
- *Les exercices sont indépendants : vous les traiterez dans l'ordre que vous souhaitez.*
- *Le barème est donné à titre indicatif.*

Consignes :

- *Précisez systématiquement le **profil** des fonctions que vous avez à écrire dans ce qui suit.*
- *Les fonctions récursives que vous écrivez doivent engendrer des **processus récursifs.***
- *Vous introduirez toute fonction jugée utile à la définition des fonctions qui vous sont demandées. Dans le cadre d'une **analyse fonctionnelle descendante**, vous donnerez les profils des fonctions introduites, vous indiquerez ce qu'elles font et les utiliserez comme si elles existaient. Vous les définirez ensuite.*
- *Des fonctions de base pour le traitement des chaînes de caractères vous sont fournies (cf. ci-dessous). **Utilisez-les.***
- *N'hésitez pas à passer une question si vous ne parvenez pas à écrire la fonction qui y est demandée. Vous pourrez malgré cela utiliser cette fonction dans la suite de votre devoir.*

Fonctions de base pour le traitement des chaînes de caractères.

```
open String;;

let est_vide = function ch -> (ch = "");;
(** string -> bool
teste si son paramètre est une chaîne vide *)

let prem_lettre = function ch -> sub ch 0 1;;
(** string -> string
renvoie la première lettre d'une chaîne de caractères *)

let reste = function ch -> sub ch 1 ((length ch) - 1);;
(** string -> string
renvoie la chaîne de caractères passée en paramètre privée de sa première lettre *)
```

Rappel : La fonction ***int_of_string*** traduit une chaîne de caractères représentant un entier en le nombre correspondant ; la fonction ***string_of_int*** fait le contraire.

Question 1 – Jeu de lettres (7 pts)

1.1 – Soit la fonction *suivant,* de profil string → string, qui prend en paramètre un caractère et renvoie celui qui le suit dans l'ordre ascii. Exemple :

```
# suivant "a";;
 - : string = "b"
# suivant "b";;
 - : string = "c"
```

Utilisez suivant pour écrire la fonction ***lettres_se_suivent*** qui prend en paramètre un mot et teste si ce mot contient deux lettres successives qui se suivent selon l'ordre ascii. Le résultat est un booléen.

Exemple :

```
# lettres_se_suivent "amnistie";;
- : bool = True  (* il y a en effet un « m » suivi d'un « n » dans le mot *)
```

1.2 – Ecrire la fonction ***lettres_idem*** qui prend en paramètre un mot et teste si ce mot contient deux lettres successives identiques. Le résultat est un booléen.

Exemple :

```
# lettres_idem "fonctionnel";;
- : bool = True  (* il y a en effet un « n » suivi d'un autre « n » dans le mot *)

# lettres_idem "banane";;
- : bool = False
```

1.3 – Généralisez les fonctions précédentes par abstraction en écrivant la fonction ***test_prop*** qui permettra de tester une propriété donnée sur 2 lettres successives d'une chaîne de caractères donnée (ex : 2 lettres identiques, 2 lettres qui se suivent dans l'ordre alphabétique, 2 voyelles, …).

1.4 – Réécrire les fonctions ***lettres_se_suivent*** et ***lettres_idem*** en utilisant *test_prop*. Vous procéderez par application partielle.

1.5 – Donnez le profil de *test_prop* en justifiant votre réponse : vous pourrez utiliser pour cela les équations aux types.

Question 2 – Jeu de raquettes (13 pts)

Nous allons dans cet exercice travailler sur des matches de tennis opposant 2 joueurs. Pour gagner un match, il faut remporter plus de sets que son adversaire. Le nombre de sets à gagner est fixé à l'avance par les organisateurs du tournoi. Un set se décompose en plusieurs jeux. Un set est gagné par le premier des 2 joueurs qui atteint :

- 6 jeux gagnants avec 2 jeux d'écart sur son adversaire,
- 7 jeux gagnants s'il n'y a pas 2 jeux d'écart.

Exemples de score final pour un set donné :

- 6 – 3 : le joueur 1 a remporté le set par 6 jeux à 3
- 6 – 7 : le joueur 2 a remporté le set par 7 jeux à 6
- 6 – 4 : le joueur 1 a remporté le set par 6 jeux à 4

Exemples de score final pour un match donné, en 3 sets gagnants :

- 6 - 1, 6 - 7, 6 – 4, 6 – 3 : le joueur 1 a remporté le match. Il a gagné 3 sets (les sets n°1, 3 et 4) et son adversaire un seul (le 2ème)
- 4 - 6, 6 - 7, 6 – 4, 6 – 3, 5 – 7 : le joueur 2 a remporté le match. Il a gagné 3 sets (les 1er, 2ème et 5ème) et son adversaire deux (le 3ème et le 4ème).

Les organisateurs du tournoi ont choisi de représenter le marque d'un match par une chaîne de caractères composée de « 1 » et de « 2 », chacun représentant la victoire d'un des 2 joueurs à un jeu. Les sets sont pas ailleurs séparés par des espaces. Ainsi par exemple, la chaîne suivante représente un match qui s'est déroulé en 3 sets :

"121212111 111111 21112111"

Dans le premier set, la marque est **"121212111" :** le joueur 1 a gagné les jeux 1, 3, 5, 7, 8 et 9 ; le joueur 2 a gagné les jeux 2, 4 et 6. Dans le second set, la marque est **"111111" :** tous les jeux ont été remportés par le joueur 1. Dans le troisième set , la marque est **"21112111" :** le joueur 2 a gagné les jeux 1 et 5, le joueur 1 tous les autres (2, 3, 4 et 6, 7, 8).
Le score correspondant à un tel match est 6 – 3, 6 – 0, 6 – 2. C'est le joueur 1 qui a gagné en 3 sets.

Cet exemple correspond à la marque d'un match qui est terminé, mais ce n'est pas toujours le cas. Ainsi par exemple, la chaîne **"121212111 111"** correspondra à un match en cours, dans lequel le joueur 1 a gagné le premier set 6-3 et dans lequel la marque pour le deuxième set est de 3-0. Ce deuxième set n'est pas terminé, le match non plus a fortiori.

2.1 – Ecrire la fonction ***nb_occur*** permettant de compter le nombre d'occurrences d'une lettre dans une chaîne de caractère (c'est le nombre de fois où la lettre apparaît dans la chaîne).

Exemple :

```
# nb_occur "a" "banane" ;;
- : int = 2
```

2.2 – Utilisez *nb_occur* pour écrire la fonction ***score_set*** qui prend la marque d'un set en paramètre (une chaîne de caractères composée de « 1 » et de « 2 ») et renvoie le score du set. Le score sera une chaîne de caractères composée de 2 nombres correspondant respectivement au nombre de sets gagnés par le joueur 1 suivi du nombre de sets gagnés par le joueur 2. Exemples :

```
# score_set "21112111" ;;          # score_set "1212121212122" ;;      # score_set "111112" ;;
- : string = "62"                  - : string = "67"                  - : string = "51"
```

2.3 – Nous allons, dans les questions qui suivent, travailler sur la marque d'un match, qu'il soit en cours ou terminé, représentée par une chaîne de caractères tel que cela a été décrit auparavant : des successions de « 1 » et de « 2 », séparées par des espaces.

Ex : "121212111 111111 22221212 21112111"

(a) Ecrire la fonction **prem_set** qui prend en paramètre la marque d'un match et renvoie la chaîne correspondant au premier set enregistré.
Exemple : # prem_set "121212111 111111 22221212 21112111" ;;
"121212111"

(b) Ecrire la fonction **reste_set** qui prend en paramètre la marque d'un match et renvoie la chaîne correspondant à la marque privée de son premier set.
Exemple : # reste_set "121212111 111111 22221212 21112111" ;;
"111111 22221212 21112111"

2.4 – Utilisez les fonctions définies précédemment pour écrire la fonction ***score_match*** qui prend en paramètre la marque d'un match donné et renvoie le score du match : c'est une chaîne de caractères dans laquelle les scores des différents sets sont mis bout à bout, séparés par des espaces.

Exemple :
```
# score_match "121212111 111111 22221212 21112111" ;;
- : string = "63 60 26 62"
# score_match "121212111 111111 2212121" ;;
- : string = "63 60 34"
```

2.5 – Ecrire la fonction ***gagne_match*** qui prend en paramètres :

- la marque d'un match (par exemple "111111 22221212 21112111 "),
- le nombre de sets requis pour gagner le tournoi (un entier),
- le nom du joueur 1 (par exemple "Paul"),
- le nom du joueur 2 (par exemple "Luc"),

et renvoie le nom du joueur qui a gagné le match si le match est terminé. Si ce n'est pas le cas, la fonction renverra l'indication "Match en cours – XXX", XXX étant le score.

Vous pourrez utiliser les fonctions écrites précédemment si besoin. Vous pourrez par ailleurs introduire toute autre fonction que vous jugerez utile. Vous donnerez les profils des fonctions introduites et indiquerez ce qu'elles font avant de les utiliser pour écrire *gagne_match*. Vous travaillerez ensuite à la définition des fonctions introduites en Caml.

Exemples :

```
# gagne_match "121212111 111111 22221212 1221212212 1212121212122"  3 "Théo" "Luc" ;;
        - : string = "Luc"
# gagne_match "121212111 111111 22221212 1221212"  3 "Théo" "Luc" ;;
        - : string = "Match en cours – 63 60 26 34"
# gagne_match "121"  3 "Théo" "Luc" ;;
        - : string = "Match en cours – 21"
# gagne_match "121212111 111111"  2 "Théo" "Luc" ;;
        - : string = "Théo"
```

2.6 – Ecrire la fonction ***set_le_plus_dispute*** qui prend en paramètre la marque d'un match et renvoie un entier correspondant au n° du set le plus disputé dans le match. C'est le set dans lequel il y a eu le plus de jeux. En cas d'égalité, votre fonction renverra le premier des numéros de sets les plus disputés. Par exemple, dans le match dont la marque est : "121212111 111111 22221212 1221212212 1212121212122", c'est le set n°5 qui a été le plus disputé : il y a eu 13 jeux au total.

A noter : le score correspondant est "63 60 26 46 67".

Ici aussi pour pourrez utiliser les fonctions écrites précédemment et en introduire de nouvelles, si besoin, dont vous donnerez les profils avant de les définir.