

Deug MIAS: UE SM23 Examen de TP Informatique

Durée 2H, sans documents.

Il est demandé de respecter le lexique des énoncés. Ne pas les recopier.

Le barème est donné a titre indicatif.

Rappels Caml:

- [] et :: dénotent respectivement les constructeurs de séquence vide d'ajout a gauche d'une séquence.
- && et || correspondent respectivement à "et puis" et "ou alors".

1 Lecture d'expressions Caml (5 pts):

Pour chacune des expressions suivantes, donner la réponse de l'interpréteur Caml (type et valeur) quand elles sont correctes. Quand les expressions sont incorrectes, expliquer précisément où se situe l'erreur. Par exemple concernant l'expression `2 = true` : elle est incorrecte car l'opérateur `=` n'est défini en Caml que si les deux opérandes sont de même type, or ici le premier est de type entier et le second de type bodéen.

1. `float_of_int(3);;`
2. `if 3 < 5 then 7.5
 else 3.0 + 7.2;;`
3. `let x = (3=4) in
 if x then [(3,4)]
 else [2];;`
4. `(2,6)::[(4,3);(6,3);(7,5)]@[9,0];;`
5. `premier([fin([2;6;7;8])::[[7]]]);;`
6. `let x=3 and y=x+2 in
 (y*x,int_of_string("234"));;`
7. `let x=((let z=3 and y=2 in y-z), 13.8) in
 x::[];;`
8. `let z=3 and x=4 and y=0 in
 y = (let x=z+2 and y=x in x-y);;`

2 Gestion d'un club de sport (8 pts):

On désire modéliser le système informatique d'un club de sports. Les informations sur les adhérents sont regroupées dans des triplets, des fiches d'inscriptions, contenant pour chaque adhérent son nom, la date limite de son inscription (sous le format jour/mois/année) et la liste des activités auxquelles il est inscrit. Il n'y a qu'une seule fiche par personne.

Par exemple:

<code>seq_exemple =</code>	Nathanaël 12/10/2002 tennis, piscine, yoga	Ariane 3/3/2003 danse, karate	Delphine 16/6/2002 step	Josua 17/1/2003 musculation, sauna
----------------------------	--	-------------------------------------	-------------------------------	--

On représente les types suivants en Caml:

```
#type Nom_Personne == string;;  
#type Date == int*int*int;;  
#type Nom_Activite == string;;
```

2.1 Questions

- Donner la définition du type Caml, `SeqAdh` représentant la séquence des informations des adhérents du club.
- Donner l'expression Caml représentant la séquence `seq_exemple`.
- Donner une fonction Caml, `Nb_Adh`, permettant de calculer le nombre d'adhérents du club.
- Donner une fonction Caml, `Seq_Kar`, en terme d'une fonction intermédiaire que l'on donnera également, qui calcule la séquence des adhérents pratiquant le Karaté.
- Donner une fonction Caml, `Adh_OK`, qui prend une date, une séquence d'adhérents et calcule la séquence des adhérents qui auront le droit d'accéder au club après la date donnée en argument. Par exemple:

<code>Adh_OK(seq_exemple, 31/12/2002) =</code>	Ariane 3/3/2003 danse, karate	Josua 17/1/2003 musculation, sauna
--	-------------------------------------	--

3 Suite de Syracuse (7 pts):

La suite de syracuse $S_n(e)$ d'un entier positif e est définie par récurrence par les équations suivantes :

$$S_0(e) = e$$
$$S_n(e) = \begin{cases} S_{n-1}(e)/2 & \text{si } S_{n-1}(e) \text{ est pair.} \\ 3S_{n-1}(e) + 1 & \text{sinon} \end{cases} \quad \text{pour } n \geq 1$$

Par exemple le début de la suite de Syracuse de 7 est le suivant :

$$\begin{aligned}S_0(7) &= 7 \\S_1(7) &= 3S_0(7) + 1 = 22 \\S_2(7) &= S_1(7)/2 = 22/2 = 11 \\S_3(7) &= 3S_2(7) + 1 = 34 \\&\dots\end{aligned}$$

On propose la réalisation Caml suivante d'une fonction qui calcule, pour un entier e donné et un rang n , la valeur de la suite de syracuse de e au rang n : $S_n(e)$.

```
let (Syr: int*int ->int)= function
  (0,e) -> e
  | (n,e) -> if Syr(n-1,e) mod 2 = 0
              then Syr(n-1,e)/2
              else 3*Syr(n-1,e)+1;;
```

3.1 Observation de la fonction Syr

Il y a une légère erreur dans la solution proposée. L'interpréteur répond.

Entree Interactive:

```
> | (n,e) -> if Syr(n-1,e) mod 2 = 0
>          ~~~
>
```

L'identificateur Syr n'est pas defini.

- Quelle est précisément cette erreur et comment la corriger ?

Une fois la correction faite on trace la fonction Syr:

```
#trace "Syr";;
#Syr(2,3);;
Syr <-- 2, 3
Syr <-- 1, 3
Syr <-- 0, 3
Syr --> 3
Syr <-- 0, 3
Syr --> 3
Syr --> 10
Syr <-- 1, 3
Syr <-- 0, 3
Syr --> 3
Syr <-- 0, 3
Syr --> 3
Syr --> 10
Syr --> 5
- : int = 5
```

- Indenter correctement la trace pour mettre en évidence la structure des appels à la fonction `Syr`.
- Combien d'appels récursifs sont engendrés par l'appel `Syr(2,3)`? Généraliser : combien faut-il d'appels récursifs pour calculer le $n^{\text{ième}}$ terme d'une suite de syracuse en utilisant `Syr` ? Justifier.
- La version proposée contient des calculs redondants. Expliquer pourquoi et comment cela se traduit sur la trace. Donner une deuxième version `Syr2` plus efficace.

3.2 Etude de la suite de Syracuse

La suite de syracuse d'un nombre e a un comportement particulier. Elle se met, quelque soit e , au bout d'un certain rang à osciller entre trois valeurs : 4, 2 et 1. Une fois qu'une de ces valeurs est atteinte on a un comportement cyclique en effet 4 étant pair le nombre suivant de syracuse sera 2, le suivant 1 et enfin comme ce nombre est impair le suivant est $3 * 1 + 1$ soit 4 etc. Considérons par exemple les suites de syracuses de 3, 13 et 21.

$$\begin{aligned} 3 &\rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow \dots \\ 13 &\rightarrow 40 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow \dots \\ 21 &\rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow \dots \end{aligned}$$

On appelle "temps de vol" d'un nombre e le rang pour lequel la suite de syracuse de ce nombre atteint pour la première fois soit 1, soit 2 soit 4. Sur nos exemples le temps de vol de 3 est 5, celui de 13 est 6 et celui de 21 est 5.

On appelle "altitude maximale" d'un nombre e la valeur maximale atteinte par la suite de syracuse de ce nombre. Sur nos exemples, l'altitude maximale de 3 et 13 est 16, et celle de 21 est 64.

- Donner une fonction Caml `Temps_Vol` qui calcule le temps de vol d'un entier.
- Donner une fonction Caml `Alt_Max` qui calcule l'altitude maximale d'un entier.

Indication : introduire une fonction intermédiaire `Maxi` prenant deux entiers comme arguments, l'un représentant l'altitude maximale rencontrée et l'autre le terme de la suite de syracuse actuel, de signature suivante.

```
let rec (Maxi: int*int->int) = function
  (mx,1) -> mx
| (mx,2) -> mx
| (mx,4) -> mx
| (mx,n) -> ...
```