Contrôle continu - IPF

vendredi 13 décembre 2013 - 30 minutes

Nom - Prénom : .....

Compléter la session suivante. Il n'y pas d'erreur syntaxique mais des erreurs de typage peuvent exister. Dans ce cas, indiquez la réponse *erreur de typage* et expliquez en quelques mots pourquoi.

Répondez directement sur la feuille.

Echauffement : session OCaml

```
# let h x = x*.x;;
val h : float -> float = <fun>

# let f (x,t) = x-.t ;;
```

```
# h (0,0)=0;;
```

```
# let w (x, y) = y && (x>0);;
```

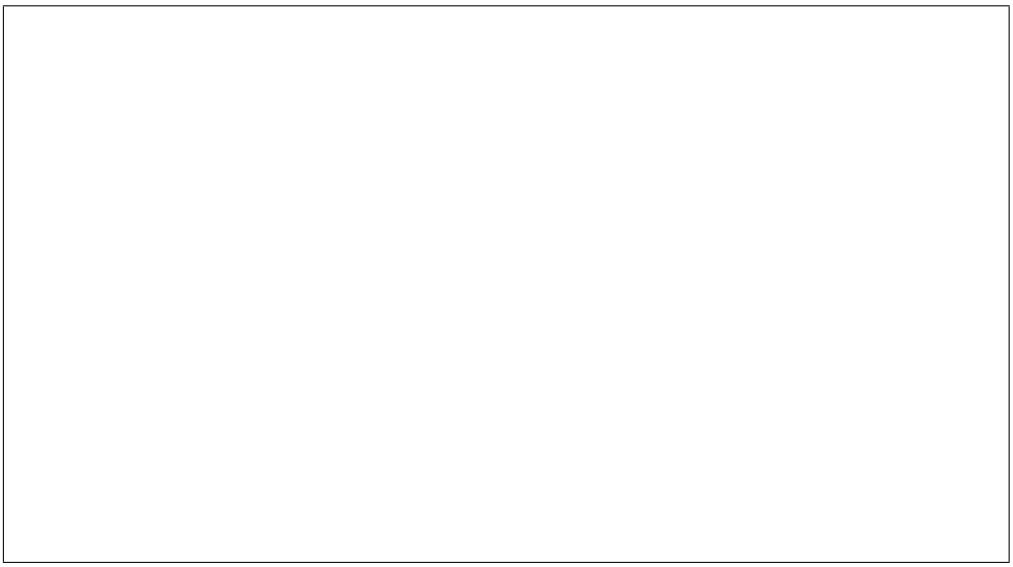
```
# [ true ; 0.0 ];;
```

```
# type lettre = Voyelle of char | Consonne of char;;

# [ Voyelle 'e' ; Consonne 'f'];;
```

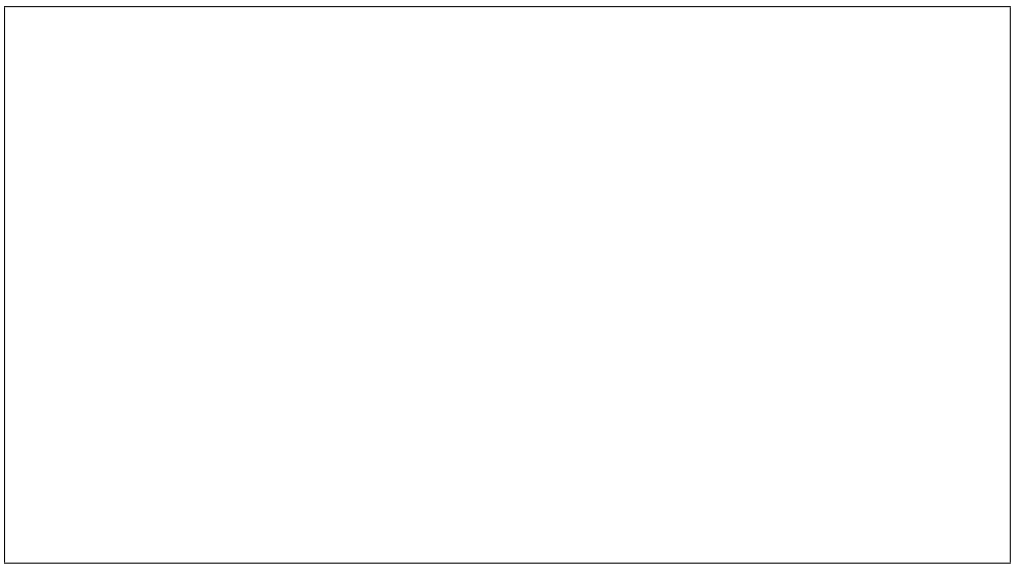
Fonction 1 : Programmez la fonction dont l'interface est la suivante

```
(* interface moyenne_pondérée
type : (int*int) list -> int,   arguments : l
précondition : liste de couples (note, coef)
postcondition : renvoie la moyenne en appliquant le coefficient "coef"
à chaque note "note"
test(s) : moyenne_ponderee [(15,2);(12,1)] = 14
rmq : on pourra utiliser une fonction intermédiaire si nécessaire.
La fonction intermédiaire pourra utiliser un accumulateur. *)
```



Fonction 2 : Programmez la fonction dont l'interface est la suivante

```
(* interface reverse
type : 'a list -> 'a list
arguments : l
précondition : aucune
postcondition : retourne la liste retournee
test(s) : reverse [1;2;3;4] = [4;3;2;1] *)
```



Fonction 3 : Programmez la fonction dont l'interface est la suivante

```
(* interface separe_voyelles_consonnes
arguments :  une liste l
précondition : l est une liste de lettres
postcondition : 2 listes (la première contenant les voyelles de l
                et la seconde les consonnes de l)
test(s) : separe_voyelles_consonnes [Voyelle 'e'; Consonne 'c'; Voyelle 'a'] =
([Voyelle 'e'; Voyelle 'a'], [Consonne 'c']) *)
```

Fonction 4 : Programmez la fonction dont l'interface est la suivante

```
(* interface puissances_2
arguments :  n (un entier)
précondition : n >=0
postcondition : la liste des puissances de 2 (de 2^0 à 2^n)
tests : puissances_2 10 = = [1; 2; 4; 8; 16; 32; 64; 128; 256; 512; 1024] *)
```