

Feuille 1 : Premiers pas

Exercice 1.1 Pour chacune des expressions suivantes, indiquer si elle est correcte ; si c'est le cas donner sa valeur et son type. **Note.** La fonction `int_of_float` convertit un `float` en `int` (en supprimant sa partie décimale), et la fonction `float_of_int` convertit un `int` en `float` .

1. `12 + 30`
2. `12.0 + 30`
3. `12.0 + 30.0`
4. `int_of_float 12.5 + 30`
5. `float_of_int 12 +. 30.0`

Exercice 1.2 Donner la valeur de chacune des expressions booléennes suivantes :

1. `3 = 4 || 4 = 4`
2. `3 = 4 && 4 = 4`
3. `not (3 = 4) && 4 = 4`
4. `not (3 = 4 || 4 = 4)`

Exercice 1.3 Pour chacune des expressions suivantes, indiquer si elle est correcte ; si c'est le cas donner sa valeur et son type.

1. `12 + 30 = 10 + 32`
2. `12.0 + 30.0 = 10 + 32`
3. `12.0 +. 30.0 = float_of_int (10 + 32)`
4. `12.0 +. 30. = 42. && 3 + 4 = 4 + 4`
5. `12.0 +. 30. = 42. && 3 + 4 != 4 + 4`
6. `12.0 +. 30. = 42. || 3 + 4 = 4 + 4`
7. `12.0 +. 30. = 42. && not (3 + 4 = 4 + 4)`
8. `int_of_string "12" + int_of_string "30"`
9. `int_of_string "12" + int_of_string "30" = 42`

Exercice 1.4 Pour chacune des expressions suivantes, indiquer si elle est correcte ; si c'est le cas, donner sa valeur et son type.

1. `let x = 12.0 in x +. 30.0`
2. `let x = 12.0 in if x +. 30.0 = 42.0 then 42 else 0`
3. `let x = 12.0 in if x +. 30.0 = 42.0 then "42" else 0`

Exercice 1.5 Pour chacune des deux expressions mathématiques suivantes dépendant d'un nombre flottant f , écrire une expression OCaml permettant de la calculer en appelant une seule fois la racine carrée et le log. Pour cela, factoriser les calculs de $\ln f$, \sqrt{f} et $\ln \sqrt{f}$ en utilisant des expressions `let in`.

$$(\sqrt{f} + \ln f) * (\sqrt{f} - 2 \ln f)$$

$$(\sqrt{f} + \ln \sqrt{f}) * (\sqrt{f} - 2 \ln \sqrt{f})$$

Exercice 1.6 Tester les fonctions de conversion `float_of_int`, `int_of_float`, `int_of_char`, `char_of_int`, `string_of_int`, `string_of_bool`,

Exercice 1.7 Donner l'expression d'une fonction anonyme qui double son argument. Donner un exemple d'appel de cette fonction.

Exercice 1.8 Indiquer, parmi les phrases suivantes OCaml, lesquelles sont :

- une expression ; Dans ce cas, donner son type et sa valeur.
- une requête let (liaison variable valeur). Dans ce cas, donner le nom, le type et la valeur de la variable.
- une phrase incorrecte à cause d'une erreur de syntaxe. Dans ce cas, expliquer pourquoi la phrase est syntaxiquement incorrecte.
- une phrase incorrecte à cause d'une erreur de type. Dans ce cas, expliquer l'erreur de type.

1. `let y = let x = 12.0 in x +. 30.0`
2. `let y = let x = 12.0 in if x +. 30.0 then 42 else 0`
3. `let x = 3 in let y = 4 in x + y`
4. `let z = let x = 3 in let y = 4 in x + y`
5. `let x = 3 in let y = 4`

Exercice 1.9 Écrire une expression fonction d'une variable `i` qui retourne une chaîne de caractères : "positive" si `i` est strictement positif, "negative" si `i` est strictement négatif, "nul" sinon.

Exercice 1.10 Mêmes questions que pour l'exercice 1.8.

1. `fun x -> x`
2. `(fun x -> x) 5`
3. `fun x -> x + 1`
4. `let f x y = x - y in f (f 1 1) 1`
5. `let compose = fun f g -> fun x -> f (g x)`
6. `let compose f g = fun x -> f (g x)`
7. `let compose f g x = f (g x)`
8. `let compose = fun f g x -> f (g x)`
9. `let mystere =
let square = fun x -> x * x in`

```
let compose = fun f g -> fun x -> f (g x) in
compose square square
```

Exercice 1.11 Écrire une fonction qui prend en paramètres 3 `float` : `a`, `b` et `c`, et qui renvoie le nombre de solutions de l'équation $ax^2 + bx + c = 0$. Lorsque le nombre de solutions est infini (par exemple quand $a = b = c = 0$, la fonction renverra `-1`). Pour alléger le code, on pourra écrire une fonction `discriminant` prenant les trois paramètres `a`, `b`, `c` qui calcule le discriminant de l'équation.