Programmation Fonctionnelle :
des concepts aux applications web
(NFP119)

Le type `'a option`

Implanter les fonctions suivantes.

1. `getOpt : 'a option * 'a -> 'a`
   `getOpt (opt, a)` renvoie `v` si `opt` est `Some(v)` et renvoie `a` sinon.

2. `isSome : 'a option -> bool`
   `isSome opt` renvoie `true` si `opt` est `Some(v)` et renvoie `false` sinon.

3. `valOf : 'a option -> 'a`
   `valOf opt` renvoie `v` si `opt` est `Some(v)` et lève l'exception prédéfinie `No_value` sinon.

4. `filter : ('a -> bool) -> 'a -> 'a option`
   `filter f a` renvoie `Some(a)` si `f(a)` est `true` et `None` sinon.

5. `join : 'a option option -> 'a option`
   La fonction `join` envoie `None` sur `None` et `Some(v)` sur `v`.

6. `map : ('a -> 'b) -> 'a option -> 'b option`
   `map f` envoie `None` sur `None` et `Some(v)` sur `Some(f v)`.

7. `mapPartial : ('a -> 'b option) -> 'a option -> 'b option`
   L'expression `mapPartial f` est équivalente à `join o (map f)`. Donner aussi une seconde implantation (plus directe) de `mapPartial`.

8. `compose : ('a -> 'b) * (c -> 'a option) -> ('c -> 'b option)`
   L'expression `compose (f, g)` est équivalente à `(map f) o g`. Donner aussi une seconde implantation (plus directe) de `compose`.

9. `composePartial : ('a -> 'b option) * (c -> 'a option) -> ('c -> 'b option)`
   L'expression `composePartial (f, g)` est équivalente à `(mapPartial f) o g`. Donner aussi une seconde implantation (plus directe) de `composePartial`.