Exercice 1

On veut définir un type spécial *PathLength*, comprenant les entiers et un nombre spécial représentant l'infini. Voici comment on peut le définir en Haskell :

```
data PathLength a = Infinity | Normal a deriving Show
```

a) On veut que ce type soit une instance de la classe *Eq*, ce qui exige de définir l'opérateur ==. Pour ce faire, complétez la définition suivante :

```
instance (Eq a) => Eq (PathLength a) where
   ...
```

b) Complétez le code ci-dessous en fournissant les définitions des fonctions *plus* et *specialMin*. La première effectue une simple addition, de telle sorte que lorqu'on additionne quelque chose à l'infini on obtient l'infini. La seconde retourne la valeur minimale d'une liste.

```
plus::(Num a)=>(PathLength a)->(PathLength a)->(PathLength a)
...définition à compléter

extractValue::(PathLength a) -> a
extractValue (Normal x) = x

specialMin::(Ord a)=>[PathLength a]->(PathLength a)
specialMin [] = error "Liste vide"
...définition à compléter
```

c) Expliquez la présence de la contrainte *(Eq a)* dans la déclaration d'instance de la question (a).

d) Expliquez la présence de la contrainte *(Num a)* dans la déclaration de type de la fonction *plus*.

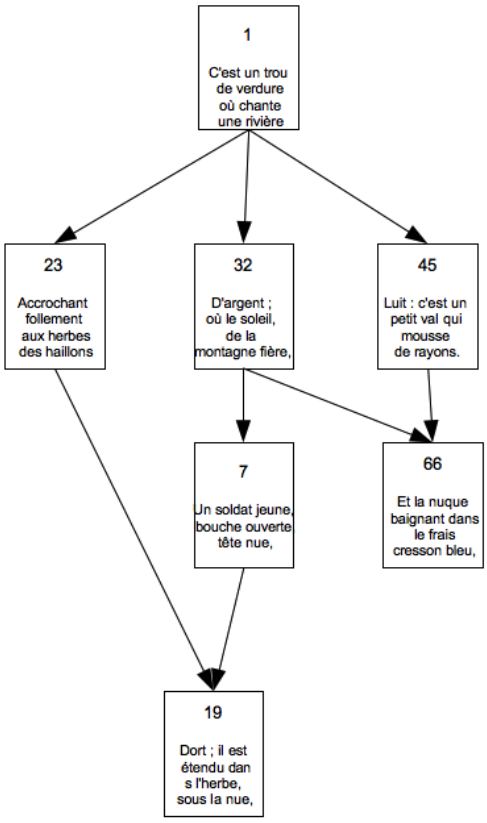

FIGURE 1 – Ensemble de pages inter-reliées

Exercice 2

Soit un ensemble de pages reliées, tel qu'illustré à la figure 1. Pour chaque page, on a un entier qui sert d'identificateur unique, un texte et une liste de pages vers lesquelles elle pointe.

a) Définissez en Haskell le type `Page`, qui représente une telle page.

b) Définissez les fonctions *findPage*, *toPage* et *shortestPath*, dont les types sont indiqués ci-dessous. La première permet de chercher dans une liste de pages celle qui correspond à l'identificateur passé en paramètre. La seconde prend une liste

d'identificateurs et les remplace par leur page correspondante. Un identificateur pour lequel il n'existe aucune page sera tout simplement ignoré. Finalement, la dernière retourne le nombre minimal de pages compris dans le chemin allant d'une page initiale fournie à au moins une des pages correspondant à la liste d'identificateurs fournie comme premier paramètre. Ainsi, si `pages` contient les pages illustrées à la figure 1, l'exécution de `shortestPath [19] pages 1` devrait retourner `Normal 3`, alors que `shortestPath [7,19] pages 45` retournera `Infinity`.

```
findPage::Integer->[Page]->(Maybe Page)
```

```
toPage::[Integer]->[Page]->[Page]
```

```
shortestPath :: [Integer]->[Page]->Integer->(PathLength Integer)
```