Exercice 4 :

1. a) Créons une variable de type Double et en lui attribuant une valeur entière.

```
scala> val note:Double=2
note: Double = 2.0

scala>
```

Nous constatons que oui ça marche, car un Double contient à la base des entiers donc a une grande précision que les Int

b) l'inverse

```
scala> val note:Int=2.0
<console>:11: error: type mismatch;
 found   : Double(2.0)
 required: Int
       val note:Int=2.0
                    ^

scala>
```

Nous constatons que l'inverse ne marche, c'est-à-dire une variable de type Int ne peut pas contenir une valeur Double.

2. Créons une variable (par exemple, x) et en lui attribuant une valeur (par exemple, 10). Puis créons une autre variable (par exemple, y) et l'affecter à une autre variable (par exemple, x=y)

```
brice@wambdevps:~$ scala
Welcome to Scala 2.11.12 (OpenJDK 64-Bit Server VM, Java 11.0.16).
Type in expressions for evaluation. Or try :help.

scala> val x=10
x: Int = 10

scala> val y=20
y: Int = 20

scala> val x=y
x: Int = 20

scala> val x=30
x: Int = 30

scala> :save xxx

scala> :load xxx
Loading xxx...
x: Int = 10
y: Int = 20
x: Int = 20
x: Int = 30

scala>
```

Conclusion

Nous constatons que la valeur de Y ne change pas

Concept de Passage par valeur et Passage par référence :

- Passage par valeur copie la valeur dans une variable de l'entendue d'une fonction c'est-à-dire que quand on utilise le passage par valeur, la copie de la valeur est transmise à la fonction, et les modifications apportées à l'intérieur de la fonction ne se répercutent pas sur les autres fonctions.

- Passage par référence copie la référence dans une variable de l'étendu d'une fonction c'est-à-dire que quand on utilise le passage par référence, l'adresse de l'argument est transmis à la fonction, et les modifications effectuées à l'intérieur de la fonction sont également répercutées en dehors de la fonction.

3. créons plusieurs variables sur une seule ligne.