Correction 1 : *Système de facturation*

Modélisation avec Z, Cas 1 Q : Quels sont les types de données (domaines) nécessaires ?
R : OrderId, Product

Q : Quel est l'état des ordres ?
R : C'est dit "en attente" et "facturé". OrderState = pending, invoiced.

Q : Quel est l'état que necessite le système ?
R : C'est l'état des produits, i.e., le stock.

```
Stock
stock : bag Product
```

Q : Un ordre est limité à un seul type de produit et sa quantité ou plusieurs (un ensemble) est préférable.
R : C'est pas clair dans l'énoncé, supposons que non.

```
  Order == { order : bag Product | order != 0 }

  OrderInvoices
    orders : OrderId -|-> Order
    orderStatus : OrderId -|-> OrderState
----
    dom orders = dom orderStatus
```

Q : Un identificateur d'ordre doit être unique sur la durée de vie du système ?
R : Pas dit, il faudrait faire un choix.

```
State
  Stock
  OrderInvoiced
  newids : P(OrderId)
----
  dom orders /\ newids = emptyset
```

Q : Quel est l'état initial du système ?
R : Pas clair, on choisit un qui satisfait les invariants.

```
InitialState
  State'
---
  stock' = emptyset
  orders' = emptyset
  newIds = OrderId
```

Q : Il y a-t-il des contraintes applicables pour toutes les opérations du système ?
R : C'est pas exemple l'utilisation des nouveaux identificateurs d'ordres.

```
Delta State
  State
  State'
----
  newids' = newids \ dom orders'
```

Q : Quelles sont les opérations sur l'état ?
R : Dans le cas 1, une seule opération, la facturation d'un ordre.

```
InvoiceOrder
  Delta State
  id? : OrderId
----
  orders(id?) in stock
  orderStatus(id?) = pending
  stock = stock - orders(?id)
  orders' = orders
  orderStatus' = orderStatus + { id? --> invoiced }
```

Q : Que fait-on en cas d'erreur ?
R : C'est pas dit.
Q : Il y a t-il besoin des rapports d'erreur ?
R : Rien est dit.

```
Report == OK | order_not_pending | not_enough_stock | no_more_ids

Success
  rep! : Report
----
  rep! = OK
```

Q : Et si l'ordre n'est pas en attente ?
R : Erreur.

```
InvoiceError
  Xi State
  id? : OrderId
  rep! : Report
----
  orderStatus(id?) != pending
  rep! = order_not_pending
```

Q : Et si pas assez de produit en stock ?
R : Erreur.

```
StockError
  Xi State
  id? : OrderId
  rep! : Report
-----
  orders(id?) not in stock
  rep! = not_enough_stock
```

Q : Qelle est la priorité entre les erreurs ?
R : A choisir en mettant la précondition adéquate.

Le système final pour le cas 1 est :

```
InvoiceOrderOp == (InvoiceOrder /\ Success) \/ InvoiceError \/ StockError
```