INF600A : Laboratoire #7
Définition d'une API coulante Ruby pour des recettes
Solution

1 Extraits du fichier `lib/recettes/recette-dsl-chainage.rb`

```ruby
def pour( quantite, unite_qte )
  DBC.require @quantite.nil? && @unite_quantite.nil?

  @quantite = quantite
  @unite_quantite = unite_qte

  self
end

def requiert( temps, unite_temps )
  DBC.require @temps_preparation.nil? && @unite_temps.nil?

  @temps_preparation = temps
  @unite_temps = unite_temps

  self
end

def utilise( ingredient, qte, unite = Integer )
  @ingredients << Ingredient.new( ingredient, qte, unite )

  self
end

alias :et :utilise
alias :avec :utilise

def pour_debuter( etape )
  @etapes << etape

  self
end
```

2 Extraits du fichier `lib/recettes/recette-dsl-bloc.rb`

```ruby
def temps( temps, unite_temps )
  DBC.require @temps_preparation.nil? && @unite_temps.nil?

  @temps_preparation = temps
  @unite_temps = unite_temps
end

def ingredient( ingr, qte, unite = Integer )
  @ingredients << Ingredient.new( ingr, qte, unite )
end

def etape( etp )
  @etapes << etp
end
```

Deux autres DSLs

Les deux exemples qui suivent sont des exemples d'API coulantes **alternatives** — exécutables et qui ont été testées! Il est donc possible de définir des méthodes qui réalisent ces APIs coulantes... mais ces mises en œuvre requièrent l'utilisation de techniques avancées de méta-programmation. Le code pour ces deux dernières versions n'est donc pas présenté — mais cela vous donne une idée de «l'expressivité» possible — j'aime bien la dernière!

A. Une API coulante avec bloc et utilisation d'`instance_eval`

```
soupe_oignon = Recette.de( "Soupe a l'oignon" ) do
    pour 4, Portions
    requiert 45, Minutes

    utilise "oignons", 4
    et "bouillon de poulet", 3, Tasse
    et "beurre", 4, Cuillere_a_soupe

    pour_debuter "Peler les oignons"
    puis "Faire sauter les oignons dans le beurre"
    puis "Ajouter le bouillon de poulet"
    finalement "Faire mijoter 30 minutes"
  end
```

B. Une API coulante avec bloc, utilisation d'`instance_eval` et réouverture (*monkey patching*) de `Integer` et `String`

```
soupe_oignon = Recette.de( "Soupe a l'oignon" ) do
  pour 4.portions
  requiert 45.minutes

  ingredients {
    4.oignons
    3.Tasses.de "bouillon de poulet"
    4.Cuilleres_a_soupe.de "beurre"
  }

  etapes {
    - "Peler les oignons"
    - "Faire sauter les oignons dans le beurre"
    - "Ajouter le bouillon de poulet"
    - "Faire mijoter 30 minutes"
  }
end
```