IIC 2143 Ingeniería de Software
Interrogación 2 - Semestre 1 /2020
Secciones 01 y 02

En caso de responder este enunciado en papel, responda cada pregunta en una hoja separada.
Recuerden que están bajo el código de honor.

Pregunta 1:

La Universidad Católica se encuentra poniendo en marcha un nuevo sistema computacional para gestionar el inventario y uso de sus laboratorios. El sistema debe ser capaz de distinguir entre dos tipos de laboratorios: laboratorios de computación y laboratorios de actividades prácticas. Los laboratorios de computación están equipados única y exclusivamente de computadores; los laboratorios de actividades prácticas están equipados con toda clase de material según la especialidad del recinto, además de algunos pocos computadores adicionales para tareas de análisis de datos. Además, cada laboratorio define una facultad a la que pertenece, y un funcionario de la universidad que va a estar a cargo de la gestión del recinto.

Para propósitos de gestión de inventario, el sistema debe ser capaz de registrar todos los computadores y material que se encuentren en cada laboratorio. Cada computador define un número de serie, sistema operativo, memoria RAM disponible y características del procesador (número de núcleos y velocidad en GHz). Cada material define únicamente un nombre. Adicionalmente, cada computador debe definir el conjunto de programas que tiene instalado. Cada programa define un nombre y versión.

Para propósitos de gestionar el acceso a los laboratorios, es necesario aclarar que estos recintos pueden definir dos políticas de uso distintas: uso restringido o bien uso liberado. Los laboratorios con uso restringido solo pueden ser usados por personas que tengan un permiso vigente. Para estos propósitos, todo permiso otorgado siempre define una fecha de emisión, una eventual fecha de expiración y la persona que lo emitió. Los laboratorios de uso liberado pueden ser usados por cualquier individuo interesado asociado a la universidad. Independiente de la política de permisos, se espera que el sistema mantenga un registro de todas las personas que hagan uso de las dependencias: debe haber un registro de la fecha y hora de entrada de toda persona que ingrese al lugar; además de la fecha y hora de salida.

Toda persona asociada a la universidad define una facultad a la que pertenece, RUT, nombre completo y puede clasificarse como alumno, profesor o funcionario. Los profesores además pueden sub-clasificarse como profesores part-time y profesores full-time. Solamente los profesores full-time pueden emitir permisos para que otras personas hagan uso de laboratorios con acceso restringido de la facultad a la que pertenecen.

Dibuje un modelo de dominio que capture los aspectos definidos en este enunciado. Especifique asociaciones, generalizaciones e incluya las cardinalidades de las asociaciones. Explicite todo supuesto adicional que haya considerado en la elaboración de su respuesta.

Pregunta 2:

Usted es contactado por el gobierno para desarrollar un algoritmo de encriptación que permita proteger sus canales de comunicación confidencial. Para este propósito, definimos dos algoritmos de encriptación distintos:

- Rot-n: dado un string, cada carácter se sustituye por aquel que se encuentra 'n' posiciones a la derecha. Después de la letra 'z' volvemos a la letra 'a'. Los espacios se mantienen igual.
    - Ejemplo: Rot-13('hola mundo') = 'ubyn zhaqb'
- SimpleTranspose-n: dado un string, cada carácter dentro del mismo string se desplaza 'n' posiciones a la derecha. Después del último carácter del string se vuelve al comienzo.
    - Ejemplo: SimpleTranspose-3('hola mundo') = 'ndohola mu'

Una característica de los algoritmos previos es que no hay nada que impida el combinarlos para agregar más capas de seguridad.

Ejemplo: Rot-1(SimpleTranspose-1(Rot-1('hola mundo'))) = 'qjqnc owpf'

Usted debe escribir en el lenguaje de programación Ruby una función `get_random_cipher` la cual devuelva un objeto que defina dos métodos:

- `encrypt(string)`: encripta el string que recibe de input y lo devuelve como resultado
- `decrypt(string)`: desencripta el string que recibe de input y lo devuelve como resultado

El algoritmo de encriptación/desencriptación implementado por el objeto devuelto por `get_random_cipher` sigue las siguientes reglas:

1. Se elige aleatoriamente un algoritmo Rot o un algoritmo SimpleTranspose
2. Para el algoritmo previamente seleccionado, se elige aleatoriamente un valor de 'n' entre 1 y 10
3. Con un 80% de probabilidad, componemos el algoritmo anterior con un nuevo algoritmo que se determina volviendo al paso 1; con un 20% de probabilidad, terminamos.

Implemente su solución utilizando el patrón Decorator.

Pregunta 3:

Debido a la pandemia que afecta al mundo entero, ha habido un gran crecimiento de las empresas de *delivery*. RapiditoFacil es una nueva empresa de este tipo que ha encargado la construcción de un software para manejar sus pedidos.

Los encargados del negocio han explicado el proceso de la siguiente forma. Un pedido es creado por un cliente desde la aplicación donde escoge los productos y al final procede a hacer el pago con un medio de pago electrónico. El pedido pasa entonces a ser preparado de acuerdo a lo que el cliente indicó. Sin embargo, es posible que algunos productos deban ser reemplazados o definitivamente eliminados. En algunos casos, se llama al cliente para acordar estos cambios; en otros casos, se hacen cambios automáticos. Una vez que el pedido está listo (con las modificaciones necesarias) es tomado por el repartidor y comienza el viaje (bicicleta, moto, etc.) hacia su destino. En el momento de salir al destino, se le envía un mensaje de texto al celular del cliente con un código de recepción que debe ser entregado al repartidor en el momento de la recepción. Al llegar el pedido el repartidor espera que se le entregue el código y una vez que el cliente lo indica, si hay coincidencia, el repartidor entrega el pedido. Si no hay coincidencia el repartidor solicita nuevo envío del código y espera confirmación del cliente. En caso de que por tercera vez no sea posible confirmar, el repartidor no entrega el pedido.

El cliente revisa su pedido y si todo está OK le tocará evaluar la experiencia con su pedido unas horas después (se le envía un email con el link a una pequeña encuesta). No todos los clientes evalúan su pedido.

Puede ser que luego de revisar el pedido el cliente note que hay algo que falta y que fue facturado. En ese caso el cliente se comunica con la empresa e indica las diferencias para lo cual la empresa genera un crédito por la diferencia que se acredita en la cuenta del cliente para ser usado en la siguiente compra. Después de enviarle al cliente la información de este ajuste, se le envía la encuesta para evaluar la experiencia con el pedido.

Su tarea es construir un diagrama de estados que represente los distintos estados y transiciones por las que pasa un pedido. Elija nombres auto explicativos para los estados pero de todas maneras describa por separado el significado de cada uno de ellos. Incluya todas las acciones y guardias necesarios en las transiciones y si no queda claro en el diagrama también agregue una nota explicativa. Puede hacer supuestos razonables que no contradigan nada de lo que se describe en el enunciado.

Pregunta 4:

Se está diseñando un juego con un escenario en la "Tierra Media" en que combatirán reinos de elfos, orcos y humanos. Cada reino tiene un rey, un castillo y un ejército. Para simplificar, supongamos que el rey, al igual que el castillo, está caracterizado solo por su nombre, pero un ejército incluye, además de una descripción, un número de efectivos.

Se desea que el software sea extensible de modo que mas adelante puedan participar otros reinos (hobbits, enanos, etc) y además cada uno de estos reinos además de rey, castillo y ejército podría incluir otros elementos (lema, territorio, etc) por lo que se ha recomendado hacer uso del patrón de diseño conocido como "abstract factory" para lograr este objetivo.

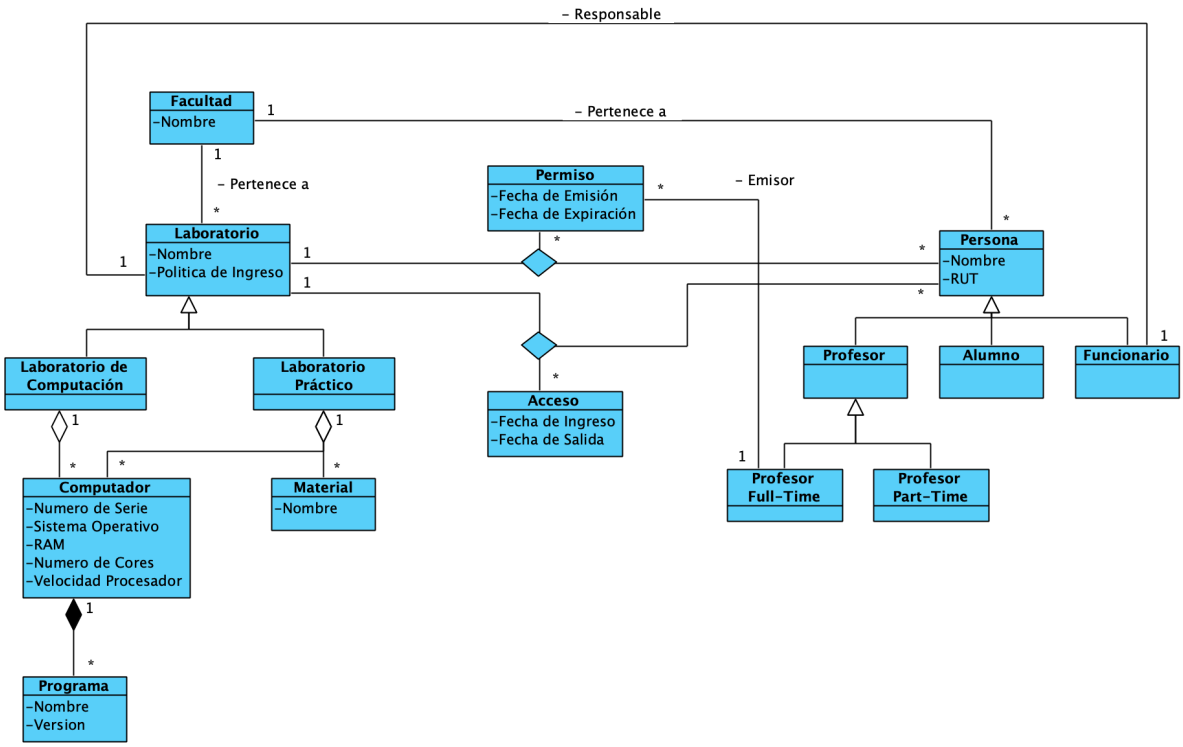
a) Haga un diagrama de clases que muestre como se usaría este patrón en el contexto del escenario que se describe más arriba.

b) Escriba la implementación en Ruby de la solución esquematizada en el diagrama de clases que dibujó en a). Debe ser completa en el sentido de incluir los 3 reinos cada uno con su rey, castillo y ejército.

c) Escriba una clase que llamaremos Realm (reino) que permite generar reinos (instancias de esta clase) de lo que se le pida (orcos, elfos, humanos). Esta clase incluye también un método describe que procede a describir el reino asociado a ese objeto. En el ejemplo siguiente se muestra la acción de describe para un objeto Realm correspondiente a un reino de orcos, uno de elfos y uno de humanos respectivamente.

```
I am Melkor, king of the Orcs
This is Isengard, home of the Orcs
This is the powerful Goblins army composed of 20000 Orcs
------------------------------------------------------------
I am Thranduil, king of the Elfs
This is Woodlands, home of the Elfs
This is the powerful Elves army composed of 2500 Elfs
------------------------------------------------------------
I am Aragorn, king of the Humans
This is Silmarillion, home of the Humans
This is the powerful Soldiers army composed of 4500 Humans
------------------------------------------------------------
```

Pauta Pregunta 1:



Nota de Pauta:

- Modelación de inventario (laboratorios, computadores, material y programas)
  - 1.5 puntos
- Modelación de personal (profesores, alumno, funcionario, relación facultad-persona, relación laboratorio-funcionario)
  - 1.5 puntos
- Política de permisos
  - 1.5 puntos
- Registro de acceso a laboratorios
  - 1.5 puntos

No es necesario especificar relaciones de composición o agregación, ni navegabilidad, pero los alumnos sí deben incluir cardinalidades. Descontar 0.2 puntos por cada cardinalidad incorrecta (máximo 1 punto de descuento).

Pauta Pregunta 2:

```ruby
class CipherDecorator
  def initialize(cipher, decorator)
    @cipher = cipher
    @decorator = decorator
  end

  def encrypt(string)
    encrypted = @cipher.encrypt(string)
    @decorator.encrypt(encrypted)
  end

  def decrypt(string)
    decrypted = @decorator.decrypt(string)
    @cipher.decrypt(decrypted)
  end
end


class Rot
  def initialize(n)
    @n = n
  end

  def encrypt(string)
    result = ''
    string.each_char do |char|
      if char == ' '
        result += char
      else
        encrypted_char = char.ord + @n
        encrypted_char -= 26 if encrypted_char > 'z'.ord
        result += encrypted_char.chr
      end
    end
    result
  end

  def decrypt(string)
    result = ''
    string.each_char do |char|
      if char == ' '
        result += char
      else
        encrypted_char = char.ord - @n
        encrypted_char += 26 if encrypted_char < 'a'.ord
        result += encrypted_char.chr
      end
    end
    result
  end
end
```

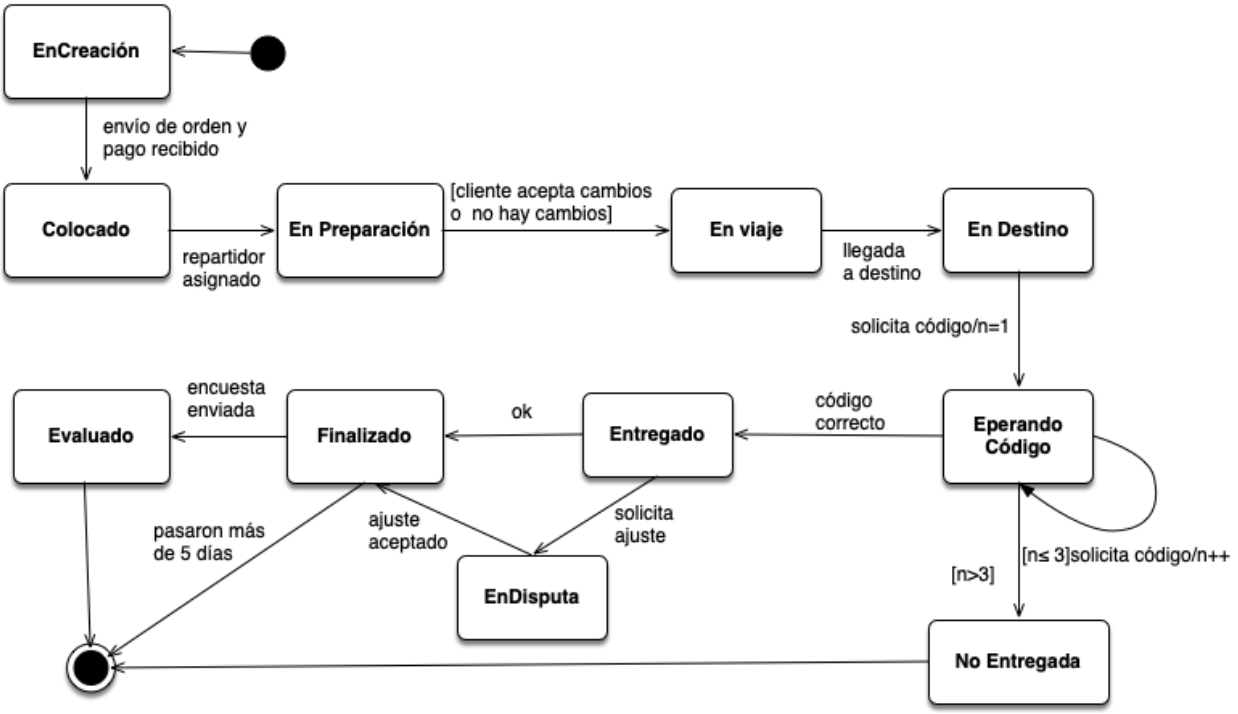
```ruby
class SimpleTranspose
  def initialize(n)
    @n = n
  end

  def encrypt(string)
    delta = @n % string.length
    result = string[delta...string.length]
    result += string[0...delta]
    result
  end

  def decrypt(string)
    delta = @n % string.length
    result = string[(string.length - delta)...string.length]
    result += string[0...(string.length - delta)]
    result
  end
end

def generate_random_cipher
  cipher = random_cipher
  loop do
    random = rand(1..5)
    break if random == 1

    cipher = CipherDecorator.new(cipher, random_cipher)
  end
  cipher
end

def random_cipher
  n = rand(1..10)
  choice = rand(2)
  if choice == 0
    cipher = Rot.new(n)
  else
    cipher = SimpleTranspose.new(n)
  end
  cipher
end
```

Nota de Pauta:

- Algoritmo ROT
  - Encriptación: 1 punto
  - Desencriptación: 1 punto
- Algoritmo SimpleTranspose
  - Encriptación: 1 punto
  - Desencriptación: 1 punto
- Correcta implementación de patrón Decorator
  - 1 punto
- Implementación de generate_random_cipher
  - 1 punto

En caso de no encontrar el patrón Decorator implementado en ninguna parte, evaluar toda la pregunta con 0 puntos.

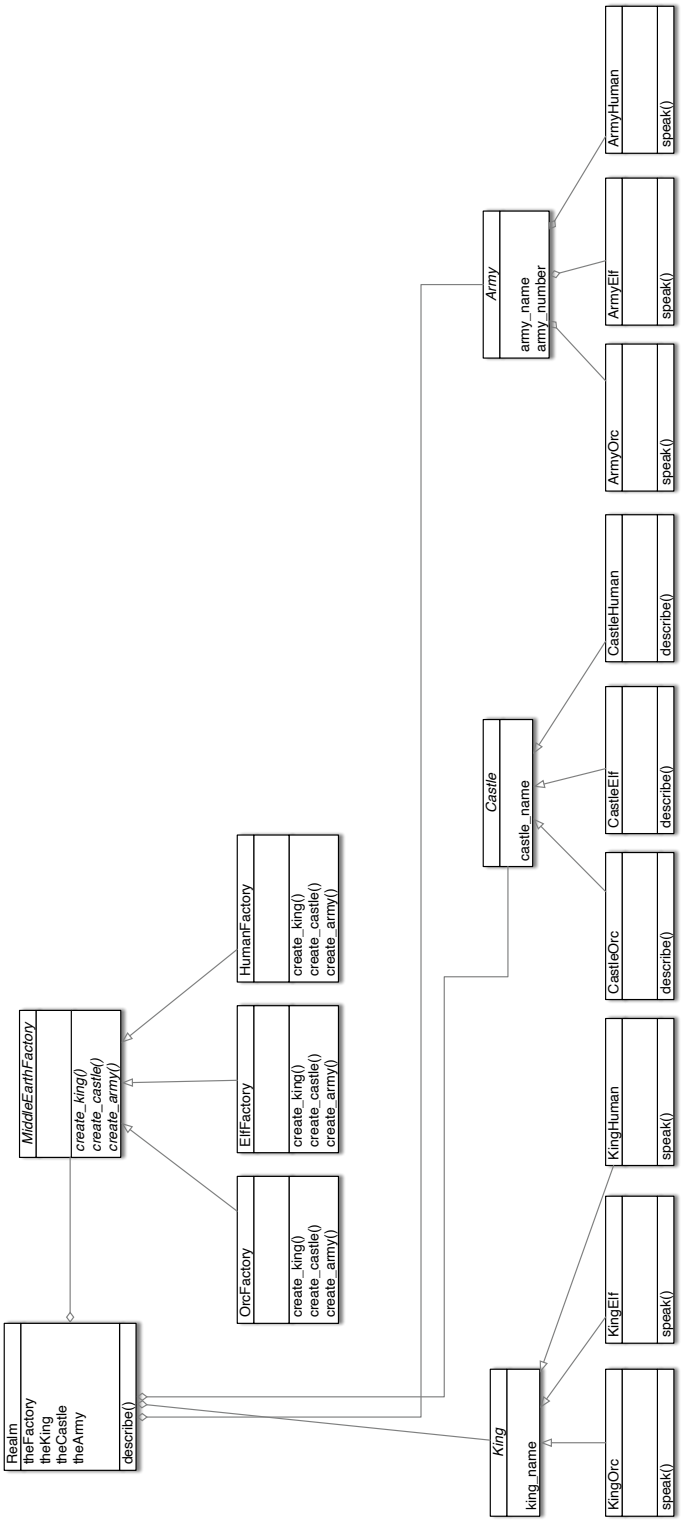
Pauta Pregunta 3:



Estados

EnCreación	El cliente está en proceso de armar el pedido, ha seleccionado productos, pero aún no ha completado el proceso no ha pagado
Colocado	El pedido ya ha sido colocado y aceptado por la empresa, pero aún no ha sido asignado a un repartidor
EnPreparación	El pedido está en proceso de armado. En este lapso puede que sea necesario llamar al cliente si es que es necesario hacer cambios o si hay productos que simplemente no pueden agregarse
EnViaje	El repartidor ya completó el pedido y ha iniciado el viaje hacia el destino indicado
EnDestino	El repartidor ha arribado al destino indicado por el cliente
EsperandoCódigo	El repartidor está esperando que el cliente le entregue un código válido. En este estado el repartidor puede solicitar reenvío del código al cliente hasta dos veces más.
Entregado	El código es válido y el pedido es entregado al cliente
NoEntregado	No fue posible validar el código y la mercadería no es entregada
EnDisputa	El cliente ha reclamado por mercadería faltante
Finalizado	O bien no hubo disputa o bien el ajuste solicitado ya ha sido ingresado y comunicado al cliente
Evaluado	La encuesta asociada al pedido ha sido respondida y enviada por el cliente y el pedido queda entonces evaluado

Pauta Pregunta 4:
a)



b)

```
class MiddleEarthFactory
  def create_king
    puts "You should implement this method in the concrete factory"
  end
  def create_castle
    puts "You should implement this method in the concrete factory"
  end
  def create_army
    puts "You should implement this method in the concrete factory"
  end
end

class OrcFactory < MiddleEarthFactory
  def create_king(name)
    KingOrc.new(name)
  end
  def create_castle(name)
    CastleOrc.new(name)
  end
  def create_army(name, number)
    ArmyOrc.new(name, number)
  end
end

class ElfFactory < MiddleEarthFactory
  def create_king(name)
    KingElf.new(name)
  end
  def create_castle(name)
    CastleElf.new(name)
  end
  def create_army(name, number)
    ArmyElf.new(name, number)
  end
end

class HumanFactory < MiddleEarthFactory
  def create_king(name)
    KingHuman.new(name)
  end
  def create_castle(name)
    CastleHuman.new(name)
  end
  def create_army(name, number)
    ArmyHuman.new(name, number)
  end
end

class King
  def initialize (name)
    @king_name = name
  end
end

class KingOrc < King
  def speak
    puts "I am #{@king_name}, king of the Orcs"
  end
```

```
end
class KingElf < King
  def speak
    puts "I am #{@king_name}, king of the Elfs"
  end
end
class KingHuman < King
  def speak
    puts "I am #{@king_name}, king of the Humans"
  end
end

class Castle
  def initialize (name)
    @castle_name = name
  end
end

class CastleOrc < Castle
  def describe
    puts "This is #{@castle_name}, home of the Orcs"
  end
end
class CastleElf < Castle
  def describe
    puts "This is #{@castle_name}, home of the Elfs"
  end
end
class CastleHuman < Castle
  def describe
    puts "This is #{@castle_name}, home of the Humans"
  end
end

class Army
  def initialize (name, number)
    @army_name = name
    @army_number = number
  end
end

class ArmyOrc < Army
  def describe
    puts "This is the powerful #{@army_name} army composed of #{@army_number} Orcs"
  end
end
class ArmyElf < Army
  def describe
    puts "This is the powerful #{@army_name} army composed of #{@army_number} Elfs"
  end
end
class ArmyHuman < Army
  def describe
    puts "This is the powerful #{@army_name} army composed of #{@army_number} Humans"
  end
end
```

```
c)
class Realm
  def initialize (aFactory, kname, cname, aname, anumber)
    @theFactory = aFactory
    @theKing = @theFactory.create_king(kname)
    @theCastle = @theFactory.create_castle(cname)
    @theArmy = @theFactory.create_army(aname, anumber)
  end
  def describe
    @theKing.speak
    @theCastle.describe
    @theArmy.describe
    puts '-' * 60
  end
end


orcosRealm = Realm.new(OrcFactory.new, "Melkor", "Isengard", "Goblins", 20000)
orcosRealm.describe

elvesRealm = Realm.new(ElfFactory.new, "Thranduil", "Woodlands","Elves", 2500)
elvesRealm.describe

humansRealm = Realm.new(HumanFactory.new, "Aragorn", "Silmarillion","Soldiers", 4500)
humansRealm.describe
```

```
I am Melkor, king of the Orcs
This is Isengard, home of the Orcs
This is the powerful Goblins army composed of 20000 Orcs
------------------------------------------------------------
I am Thranduil, king of the Elfs
This is Woodlands, home of the Elfs
This is the powerful Elves army composed of 2500 Elfs
------------------------------------------------------------
I am Aragorn, king of the Humans
This is Silmarillion, home of the Humans
This is the powerful Soldiers army composed of 4500 Humans
------------------------------------------------------------
```