# ESCUELA COLOMBIANA DE INGENIERÍA PROGRAMACIÓN ORIENTADA A OBJETOS PROYECTO INICIAL Ciclo No 1 2018-02

### **PROYECTO INICIAL**

El proyecto inicial tiene como propósito desarrollar una aplicación que permita simular la situación planteada en el **Problema C** de la maratón de programación internacional 2018 **Conquer The World** 

#### **PRIMER CICLO**

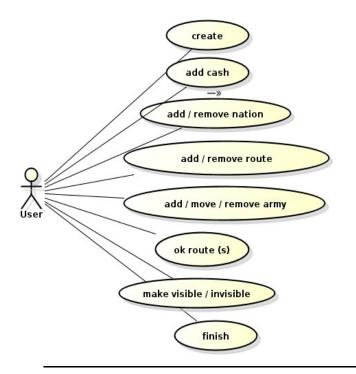
Los requisitos para el primer ciclo de desarrollo están indicados a continuación. No olviden que siempre hay un requisito implícito: el de **EXTENSIBILIDAD**.

En esta entrega NO deben resolver el problema de la maratón sólo deben construir el simulador .

### **REQUISITOS FUNCIONALES**

El simulador debe permitir:

- 1. Crear un mundo
- 2. Adicionar efectivo al presupuesto de batalla
- 3. Adicionar y eliminar una nación
- 4. Adicionar y eliminar rutas
- 5. Adicionar, eliminar y mover ejércitos
- 6. Validar si las rutas cumplen los requisitos
- 7. Hacer visible o invisible el simulador (debe poder funcionar invisible)
- 8. Terminar el simulador



**create.** Requisito 1.

add cash. Requisito 2.

add / remove nation. Requisito 3.

add / remove route. Requisito 4.

**add / move / remove army.** Requisito 5.

**ok route (s).** Requisito 6.

make visible / invisible. Requisito 7.

finish. Requisito 8.

# ConquerWorld

- + \_(maxX : int, maxY : int) : ConquerWorld
- + addCash(value : int) : void
- + addNation(shape: String, area: int, color: String, position: int[], armiesNeeded: int): void
- + addRoute(nations : String[], cost : int) : void
- + addArmy(nation : String) : void
- + addArmies(nations : String[]) : void
- + removeNation(color: String): void
- + removeRoute(nations : String[]) : void
- + removeArmies(nation : String) : void
- + moveArmy(fromNation : String, toNation : String) : void
- + okRoute(nations : String[]) : boolean
- + okRoutes(): boolean
- + makeVisible(): void
- + makeInvisible(): void
- + finish(): void
- + ok(): boolean

# **REQUISITOS DE USABILIDAD**

- 1. Las naciones se deben graficar usando diferentes colores y un distintivo para indicar si la nación está subyugada. Mínimo: 20 colores y 5 formas visibles.
- 2. Las rutas que se pueden recorrer con el presupuesto disponible deben tener un distintivo visual.
- 3. El presupuesto con el que se cuenta para la batalla se debe representar gráficamente. (No en números)
- 4. El simulador debe poder funcionar en modo visible o invisible, al iniciar es invisible
- 5. Si el usuario comete algún error se le debe presentar un mensaje especial, sólo si el simulador es visible.

# **REQUISITOS DE DISEÑO Y CONSTRUCCIÓN**

- En su desarrollo debe respetar las decisiones de diseño presentes en este diagrama de clases para la clase principal. El método **ok** retorna si la última operación se pudo realizar o no.
- Las clases se deben construir reutilizando los componentes del proyecto shapes que sean necesarios.
- El paquete shapes puede ser extendido, si se requieren otras funcionalidades. Incluyan en la retrospectiva las extensiones y su justificación.
- Las clases deben tener la documentación estándar de java. No olvidar revisar la documentación generada.
- Las clases se deben construir en Bluej. El nombre del nuevo proyecto debe ser conquerWorld

### **REQUISITOS DE ENTREGA**

Los productos los deben publicar en el espacio preparado en moodle en un archivo .zip con un nombre igual a la concatenación de los apellidos de los autores, ordenados alfabéticamente.

# Es necesario incluir la retrospectiva.

- 1. ¿Cuáles fueron los mini-ciclos definidos? Justifíquenlos.
- 2. ¿Cuál es el estado actual del laboratorio en términos de mini-ciclos? ¿por qué?
- 3. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)
- 4. ¿Cuál consideran fue el mayor logro? ¿Por qué?
- 5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo?
- 6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?
- Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué?