

Estructura de Datos

Entrega 02 Proyecto RISK

Miguel Angel Rico Llanos

Mateo Maldonado Diaz

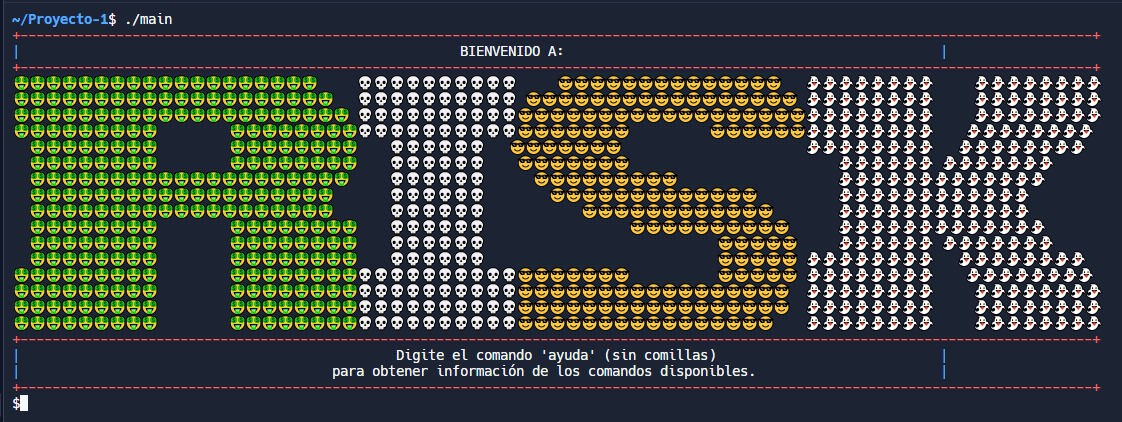
Edward Andres Quintero Palma

John Corredor

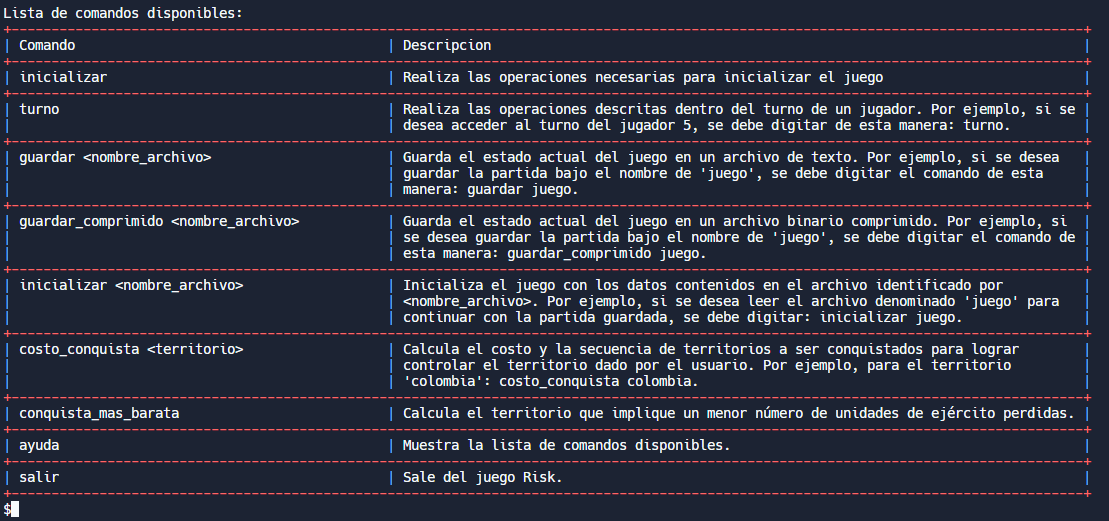
17/10/2023

1. **Descripción de Entradas y Salidas**
   1. **Interfaz**

El primer apartado del proyecto hace referencia a la interfaz, es decir al inicio de todo el código, al inicializar el código se obtiene por pantalla lo siguiente:



Este apartado recibe por consola la palabra ‘ayuda’ lo cual muestra el siguiente menú.



En este menú se muestran todos los comandos y operaciones que realizará el programa. Este menú recibe por consola el nombre del comando a realizar, y posteriormente se reciben y muestran distintos datos y menús dependiendo el comando seleccionado, a continuación se hará la explicación de una serie de comandos.

* 1. **Comando Inicializar**

Este comando realiza las operaciones necesarias para inicializar el juego. Inicialmente, el comando pregunta la cantidad de jugadores, para cada jugador su nombre, y luego, por turnos, preguntar a cada jugador en qué territorio desea ubicar sus unidades de ejército. En cada turno, el jugador sólo puede indicar un único territorio a ocupar.

Texto

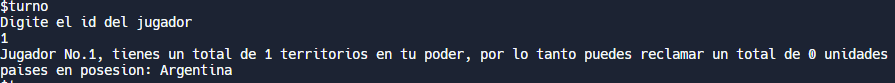
Descripción generada automáticamente

Como se puede evidenciar, el comando inicializar muestra en pantalla “Digite la cantidad de jugadores”, esto recibe por consola un entero, posteriormente se pide que se ingresen los nombres de los jugadores, esto recibe un string (Una cadena de caracteres), continuando con el comando, le solicita a cada jugador que ingrese el id del continente que desea escoger para inicializar su juego y posteriormente el pais que desea escoger en dicho continente, estos dos apartados reciben por pantalla un entero, para acabar con el comando se finaliza con un mensaje recapitulando la información recibida y arroja un mensaje de éxito de que el juego se inicializo correctamente.

Es decir, al final la salida de este comando es un valor booleano que indica si la inicialización del juego se realizó con éxito o no. Retorna `true` si la inicialización se realiza correctamente, y `false` si hay algún error en el proceso.

* 1. **Comando Turno**

Este comando, se encarga de realiza las operaciones de un jugador (obtener nuevas unidades, atacar y fortificar). De esta forma, el comando informa al jugador cuántas unidades adicionales puede reclamar, para luego preguntarle en cuáles de sus territorios las quiere asignar y en qué cantidad. A continuación, pregunta la configuración del ataque, desde cuál territorio y hacia cuál territorio, verificando las condiciones ya descritas. Luego informa los valores obtenidos con los dados, y la cantidad de unidades que se ganan o pierden. Este proceso se repite hasta que alguno de los dos territorios se quede sin unidades, o hasta que el atacante decida detenerse. Finalmente, el comando pregunta al jugador los territorios vecinos que desea seleccionar para la fortificación, así como la cantidad de unidades que se trasladarán de uno al otro.



Al ingresar el comando “turno” solicita el id del jugador, por lo tanto, recibe un entero por consola, posteriormente muestra la información de cuantos territorios tiene en su poder el jugador y cuantas unidades puede reclamar.

* 1. **Comando Salir**

Este último comando simplemente se encarga de termina la ejecución de la aplicación.



Este comando no recibe nada por consola y simplemente muestra un mensaje por pantalla agradeciendo por usar el programa.

1. **Descripción de TADs.**

* **TAD Jugador:**

Atributos:

Nombre: String

ID: Entero

País: String

CantPais: Entero

Países: Lista de Strings

Operaciones:

ObtenerNombre():

ObtenerID():

ObtenerPais():

ObtenerCantPais():

ObtenerPaises():

FijarNombre(n\_dato):

FijarID(n\_dato):

FijarPais(n\_dato):

FijarCantPais(n\_dato):

FijarPaises(n\_dato):

Calendario

Descripción generada automáticamente con confianza media

* TAD País:

Atributos:

Nombre: String

PaisesVecinos: Lista de Strings

Operaciones:

ObtenerNombre():

ObtenerPaisesVecinos():

FijarNombre(n\_dato):

FijarPaisesVecinos(n\_dato):

Texto

Descripción generada automáticamente con confianza media

* Fortificar:

Atributos:

TerritorioX: String

Unidades: Entero

Operaciones:

ObtenerTerritorioX():

ObtenerUnidades():

FijarTerritorioX(n\_dato):

FijarUnidades(n\_dato):

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

* **TAD Territorio:**

Atributos:

nombre: Cadena

unidades\_ejercito: Entero

Operaciones:

ObtenerNombre(): Cadena

ObtenerUnidadesEjercito(): Entero

EstablecerUnidadesEjercito(cantidad:Entero): void

Escala de tiempo

Descripción generada automáticamente con confianza media

1. **Funcionamiento General Comandos**

A continuación, se muestra un diagrama el cual es bastante claro respecto al funcionamiento de los comandos, esta muestra un paso a paso de las acciones que se realizan durante la ejecución de cada operación.

Diagrama

Descripción generada automáticamente

1. **Plan de Pruebas**

A continuación, se adjuntarán imágenes del funcionamiento del comando “Inicializar”, se mostrarán todas las opciones disponibles en el momento y el resultado por pantalla.

Así se vería una correcta inicialización del juego

Texto

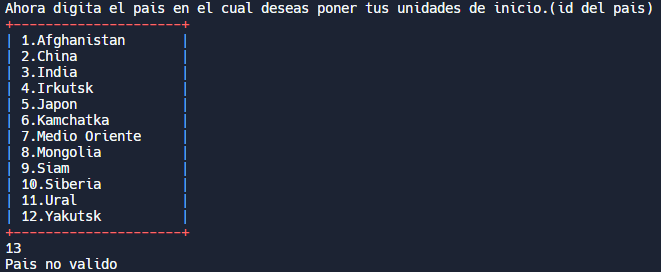
Descripción generada automáticamente

Si se ingresa una opción de continente no valida, muestra un mensaje de continente no valido y vuelve a mostrar el menú preguntando el continente nuevamente.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Si se ingresa una opción de país no valida, muestra un mensaje de país no valido y vuelve a mostrar el menú preguntando el país nuevamente.



Operación Guardar:

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Error al querer guardar una partida que no ha sido inicializada:



Carga correcta del archivo:

Texto

Descripción generada automáticamente

Error al intentar iniciar un juego al momento de haber cargado otro:

Texto

Descripción generada automáticamente

Error al momento de intentar abrir un archivo que no existe:

Texto

Descripción generada automáticamente

No fue posible llevar a cabo la implementación de "guardar\_comprimido" ya que no pudimos entender como implementar la codificación de Hauffman en código en c++, de igual manera se tendrá en cuenta para la tercera entrega.

**ACTUALIZACIÓN DE LA ENTREGA 1:**

Se corrigió el retorno booleano que presentaba fallas al momento de validar si la partida ya había sido inicializada, además de añadir las funciones correspondientes para la funcionalidad total de la función turno (Dichas funciones fueron aplicadas, pero aún no son implementadas)