Quiz diagramas de clases.

Juan Manuel Díaz Moreno A00394477.

- Análisis del problema.

Cliente	Empresa de videojuegos.
Usuarios	Administrador del videojuego.
Requerimientos Funcionales	R1: Crear un jugador. R2: Registro de nivel. R3: Calcular el nivel de complejidad. R4: Registrar tesoro a un nivel. R5: Registrar enemigos a un nivel. R6: Modificar el puntaje de jugador. R7: Imprimir los tesoros y enemigos. R8: Incrementar nivel de jugador. R9: Imprimir la cantidad de un tipo de tesoro encontrado en todos los niveles. R10: Imprimir la cantidad de un tipo de enemigo encontrado en todos los niveles. R11: Imprimir el tesoro más repetido. R12: Imprimir la ubicación del enemigo con mayor puntuación. R13: Imprimir el top 5 de jugadores. R14: Imprimir las consonantes en los nombres de los enemigos.
Mundo del problema	• El juego contará con un máximo de 20 jugadores, 10 niveles, 50 tesoros y 25 enemigos. • El puntaje inicial de un jugador es 10, y sus vidas son 5. • Se enumeran los grados de complejidad de un nivel en alto, medio y bajo. • Los enemigos se clasifican en ogros, abstractos, jefes y mágicos.
Requerimientos no funcionales	R1: El despliegue de los tesoros en la aplicación web de debe tardar más de dos segundos. R2: Funcionamiento en aplicación web y móvil. R3: El juego tendrá una resolución de 640 x 480.

- Análisis de requerimientos funcionales.

Nombre o identificador	R1: Crear un jugador.		
Resumen	El sistema debe recibir y validar la información del jugador, de este modo, añadiéndolo al video juego junto al nivel asignado por sus puntos (puntaje inicial fijo) y el puntaje para pasar de nivel.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Nickname.	String	El nickname debe ser único.
	Nombre.	String	
Actividades generales necesarias para obtener los resultados	1. Recibir la información del jugador a crear. 2. Validar que el nickname es único en el videojuego. 3. Establecer el puntaje inicial del jugador (10 puntos) y sus 5 vidas. 4. Asignar un nivel al jugador teniendo en cuenta su puntaje (10 puntos). 5. Por medio del puntaje inicial se asignan los puntos necesarios para pasar de nivel.		
Resultado o postcondición	Jugador registrado.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de validación de creación.	String	Ingreso previo de datos por el usuario.
	Nivel del jugador.	String	Jugador creado.
	Puntaje para el siguiente nivel.	String	El jugador debe estar ubicado en un nivel.

Nombre o identificador	R2: Registro de nivel.		
Resumen	El sistema recibe los datos del nivel, una vez validados registra los tesoros y enemigos para definir el grado de complejidad del nivel.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Número de id.	String	Número de id único.
	Puntaje para pasar de nivel.	int	

	Tesoros.	String	Posiciones disponibles en el arreglo de 50.
	Enemigos.	String	Posiciones disponibles en el arreglo de 25.
Actividades generales necesarias para obtener los resultados	1. Recibir la información del nivel a crear. 2. Validar el número de identificación del nivel. 3. Registrar tesoros si estos no sobrepasan los 50. 4. Registrar enemigos si estos no sobrepasan los 25. 5. Establecer el grado de complejidad del nivel.		
Resultado o postcondición	Nivel registrado.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de validación.	String	Ingreso previo de datos por el usuario.

Nombre o identificador	R3: Calcular el nivel de complejidad.		
Resumen	El sistema compara los puntos que otorgan los tesoros y los enemigos de un nivel, de este modo, define el grado de complejidad (alto, medio, bajo) de este.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Tesoros.	String	
	Enemigos.	String	
Actividades generales necesarias para obtener los resultados	1. Recibe el total de tesoros y enemigos de un nivel. 2. Compara los puntos que otorgan los tesoros y enemigos del nivel. 3. Define que, si los puntos que otorgan los tesoros son mayores a los que otorgan los enemigos, el tipo de complejidad es bajo, si es igual es medio, y si los puntos de los enemigos son mayores que los tesoros la complejidad es alta. 4. Se aplica el tipo de complejidad al nivel.		
Resultado o postcondición	Grado de complejidad del nivel definido.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Tipo de complejidad.	String	Enemigos y tesoros registrado en el nivel.

Nombre o identificador	R4: Registro de tesoro a un nivel.		
Resumen	El sistema recibe los datos del tesoro y pregunta al usuario cuantos tesoros de un tipo quiere registrar, generando una posición aleatoria dependiendo de la resolución de pantalla para cada tesoro.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Nombre.	String	
	URL de la imagen.		
	Puntaje del tesoro.	double	
	Posición.	int	Generación de acuerdo a la resolución de pantalla.
Actividades generales necesarias para obtener los resultados	1. Recibir los datos ingresados por el usuario. 2. Valida que los tesoros a registrar puedan ser ubicados en una posición del arreglo cada uno (50 tesoros). 3. Pregunta al usuario cuantos tesoros de un tipo quiere registrar en el nivel. 4. Genera aleatoriamente la posición X y Y de cada tesoro.		
Resultado o postcondición	Tesoros registrados en el nivel.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de validación de creación.	String.	Ingreso previo de datos por el usuario.
	Posición de los tesoros registrados.	int	Tesoro registrado.

Nombre o identificador	R5: Registrar enemigos a un nivel.		
Resumen	El sistema registra los datos del enemigo para que sea generado aleatoriamente en el nivel, de tal modo que no se repita el enemigo, los tipos de enemigo son ogros, abstractos, jefes y mágicos.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Nombre.	String	Nombre de identificación único.
	Tipo de enemigo.	String	Establecido dentro de los cuatro tipos.

	Puntaje de derrota.	double	
	Puntaje de victoria.	double	
	Posición.	int	Generación de acuerdo a la resolución de pantalla.
Actividades generales necesarias para obtener los resultados	1. Recibir la información del enemigo. 2. Valida que los enemigos se puedan registra cada uno en una posición del arreglo (25 enemigos). 3. Valida que en nivel no se repitan los enemigos registrados. 4. Genera aleatoriamente la posición X y Y de cada enemigo.		
Resultado o postcondición	Enemigos registrados en el nivel.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de validación de creación.	String	Ingreso previo de datos por el usuario.
	Posición de los enemigos registrados.	int	Enemigos registrados.

Nombre o identificador	R6: Modificar el puntaje de jugador.		
Resumen	El sistema se ubica en el arreglo de jugadores para modificar el atributo del puntaje del jugador.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Nuevo puntaje.	double	
Actividades generales necesarias para obtener los resultados	1. Valida la información registrada por el usuario. 2. En el arreglo de jugadores elimina el atributo de puntaje actual. 3. Registra la nueva puntuación registrada por el usuario.		
Resultado o postcondición	Puntaje de jugador modificado.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de validación.	String	Previo registro de información.

Nombre o identificador	R7: Imprimir los tesoros y enemigos.		
Resumen	El sistema recibe un nivel que digita el usuario, busca este nivel en el arreglo y si se encuentra imprime los tesoros y enemigos de dicha posición.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Nivel	String	El nivel se encuentra dentro de los 10 del juego.
Actividades generales necesarias para obtener los resultados	1. Recibir la información dada por el usuario. 2. Validar si el nivel se encuentra en el juego. 3. Identificar el arreglo de tesoros y enemigos de la posición del nivel seleccionado por el usuario. 4. Crear un mensaje tipo String con los enemigos y tesoros separados por una coma.		
Resultado o postcondición	Tesoros y enemigos de un nivel informados en pantalla.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de tesoros y enemigos encontrados.	String.	Nivel encontrado en el juego.

Nombre o identificador	R8: Incrementar nivel de jugador.		
Resumen	El sistema incrementa el nivel al jugador validando los puntos que tiene actualmente el usuario, e informando los necesarios para el siguiente nivel.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Nivel.	String	El nivel se encuentra dentro de los 10 del juego.
Actividades generales necesarias para obtener los resultados	1. Valida los puntos actuales del jugador. 2. Si los puntos son los suficientes cambia de nivel al jugador e informa por medio de un mensaje, si los puntos no son los suficientes informa el puntaje requerido para subir.		
Resultado o postcondición	Jugador subido de nivel.		

Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de validación.	String	Previo registro de jugador.

Nombre o identificador	R9: Imprimir la cantidad de un tipo de tesoro encontrado en todos los niveles.		
Resumen	El sistema recibe un tipo de tesoro a buscar en todos los arreglos de tesoros de todos los arreglos de niveles, informando la cantidad encontrada en pantalla.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Tipo de tesoro.	String	El tesoro se encuentra en alguno de los 10 niveles.
Actividades generales necesarias para obtener los resultados	1. Recibe la información dada por el usuario. 2. Valida si el tipo de tesoro ingresado por el usuario se encuentra en los niveles. 3. Por cada posición del arreglo de niveles evalúa si se encuentra el tipo de tesoro buscado en el arreglo de tesoros. 4. Informa la cantidad del tipo tesoro encontrada.		
Resultado o postcondición	Número de tesoros informado en pantalla.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Cantidad de tesoros encontrados.	int	Tesoro registrado al menos en un nivel.
Nombre o identificador	R10: Imprimir la cantidad de un tipo de enemigo en todos los niveles.		
Resumen	El sistema recibe un tipo de enemigo a buscar en todos los arreglos de enemigos de todos los arreglos de niveles, informando la cantidad encontrada en pantalla.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Tipo de enemigo.	String	Pertenece a la numeración de enemigos del juego.
Actividades generales necesarias para obtener los resultados	1. Recibe la información dada por el usuario. 2. Valida si el tipo de enemigo ingresado por el usuario se encuentra en los niveles. 3. Por cada posición del arreglo de niveles evalúa si se encuentra el tipo de enemigo buscado en el arreglo de enemigos. 4. Informa la cantidad del tipo enemigo encontrada.		

Resultado o postcondición	Número de enemigos informado en pantalla.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Cantidad de enemigos encontrados.	int	Enemigo registrado en al menos un nivel.

Nombre o identificador	R11: Imprimir el tesoro más repetido.		
Resumen	El sistema pasa por las posiciones del arreglo de niveles buscando en el arreglo de tesoros el que más se ha registrado para mostrarlo en pantalla.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Tesoros.	String	Tesoros registrados en los niveles.
Actividades generales necesarias para obtener los resultados	1. Pasa por cada posición del arreglo de niveles. 2. En cada posición busca el arreglo de tesoros para informar el que más se ha registrado.		
Resultado o postcondición	Tesoro más repetido informado en pantalla.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Tesoro más repetido.	String	Ingreso previo de información.
Nombre o identificador	R12: Imprimir la ubicación del enemigo con mayor puntuación.		
Resumen	El sistema pasa por las posiciones del arreglo de niveles buscando en el arreglo de enemigos el que otorga una mayor puntuación para mostrar en pantalla.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Enemigos.	String	Enemigos registrados en los niveles.
Actividades generales necesarias para obtener los resultados	1. Pasa por cada posición del arreglo de niveles. 2. En cada posición busca el arreglo de enemigos para informar el que otorga al jugador una mayor puntuación.		
Resultado o postcondición	Enemigo que otorga mayor puntuación informado en pantalla.		

Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Enemigo que otorga mayor puntuación.	String	Ingreso previo de información.

Nombre o identificador	R13: Imprimir el top 5 de jugadores.		
Resumen	El sistema pasa por el arreglo de jugadores, tomando a los cinco jugadores con mayor puntuación y ordenándolos de manera ascendente.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Jugadores.	String	Jugadores registrados en el video juego.
Actividades generales necesarias para obtener los resultados	1. Valida que existan al menos cinco jugadores registrados en el video juego. 2. Selecciona a los cinco que tengan una mayor puntuación. 3. Ordena a los 5 jugadores de manera ascendente y en un mensaje tipo String.		
Resultado o postcondición	Top 5 de jugadores informados en pantalla.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Mensaje de jugadores.	String	Jugadores encontrados en el video juego.

Nombre o identificador	R14: Imprimir las consonantes en los nombres de los enemigos.		
Resumen	El sistema pasa por las posiciones del arreglo de niveles, buscando en el arreglo de enemigos el tipo (nombre) de cada uno, tomando sus consonantes y mostrándolas en pantalla.		
Entradas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Enemigos.	String	Enemigos registrados en los niveles.
Actividades generales necesarias para obtener los resultados	1. Valida que existan enemigos en los niveles. 2. En cada posición del arreglo de enemigos toma las consonantes del atributo del nombre de cada enemigo. 3. Imprime en pantalla las consonantes de dichos enemigos.		

Resultado o postcondición	Consonantes de los enemigos informados en pantalla.		
Salidas	Nombre entrada	Tipo de dato	Condición de selección o repetición
	Consonantes de los nombres.	String	Ingreso previo de información.

- Tablas de trazabilidad.

Requerimiento funcional	Nombre de la clase	Nombre del método
R1: Crear un jugador.	Clase Player	Player(nickname: String, name: String, initialScore: double, initialLives: int)
	Clase Player	validateNickName(nickname: String)
	Clase VideoGame	assingLevel(initialScore: double)
	Clase VideoGame	createrPlayer(Player: players)

Requerimiento funcional	Nombre de la clase	Nombre del método
R2: Registro de nivel.	Clase Level	Level(idNumber : String, pointsForNextLevel : int)
	Clase Level	addTreasure()
	Clase Level	addEnemy()
	Clase Level	calculateGradeOfComplexity()
	Clase VideoGame	createLevel(idNumber : String, pointsForNextLevel : int)

Requerimiento funcional	Nombre de la clase	Nombre del método
R3: Calcular el nivel de complejidad.	Clase Level	calculateGradeOfComplexity()

Requerimiento funcional	Nombre de la clase	Nombre del método
R4: Registrar tesoro a un nivel.	Clase Treasure	Treasure(evelTreasure : int, pictureURL : String, positionPixels : String, foundScore : int)
	Clase Level	addTreasure()
	Clase VideoGame	createTotalTreasure()
	Clase VideoGame	generatePosition()

Requerimiento funcional	Nombre de la clase	Nombre del método
R5: Registrar enemigos a un nivel.	Clase Enemy	Enemy(name : String, type : String, score : double, pixelsPosition : double)
	Clase VideoGame	validateEnemyRepetition(name: String)
	Clase VideoGame	createEnemy(name : String, type : String, score : double, pixelsPosition : double)
	Clase VideoGame	generatePosition()

Requerimiento funcional	Nombre de la clase	Nombre del método
R6: Modificar el puntaje de jugador.	Clase VideoGame	deletePlayerScore(score: double)
	Clase VideoGame	changePlayerScore()

Requerimiento funcional	Nombre de la clase	Nombre del método
R7: Imprimir los tesoros y enemigos.	Clase VideoGame	printTreasuresAndEnemys(Treasure treasure, Enemy enemy)
	Clase Treasue	getTreasure()
	Clase Enemy	getEnemy()

Requerimiento funcional	Nombre de la clase	Nombre del método
R8: Incrementar nivel de jugador.	Clase Player	calculateLevelToPlay(Level)
	Clase VideoGame	increaseLevel(initialScore)

Requerimiento funcional	Nombre de la clase	Nombre del método
R9: Imprimir la cantidad de un tipo de tesoro encontrado en todos los niveles.	Clase VideoGame	searchTreasureByName(name: String)
	Clase VideoGame	printTypeTreasure()

Requerimiento funcional	Nombre de la clase	Nombre del método
R10: Imprimir la cantidad de un tipo de enemigo encontrado en todos los niveles.	Clase VideoGame	searchEnemyByName(name: String)
	Clase VideoGame	printTypeEnemy()

Requerimiento funcional	Nombre de la clase	Nombre del método
R11: Imprimir el tesoro más repetido.	Clase VideoGame	getMaxTreasure()

Requerimiento funcional	Nombre de la clase	Nombre del método
R12: Imprimir la ubicación del enemigo con mayor puntuación.	Clase VideoGame	getMaxScoreEnemy()
	Clase Enemy	getEnemyPosition(position: int)
	Clase VideoGame	printPosition()

Requerimiento funcional	Nombre de la clase	Nombre del método
R13: Imprimir el top 5 de jugadores.	Clase VideoGame	printTopPlayers(Player player)
	Clase Player	getPlayersScore()

Requerimiento funcional	Nombre de la clase	Nombre del método
R14: Imprimir las consonantes en los nombres de los enemigos.	Clase VideoGame	printConsonantsEnemys(name: String)