**TABLA DE ESPECIFICACIÓN DEL PROBLEMA DE INGENIERÍA DE SOFTWARE, identificando los siguientes elementos**

|  |  |
| --- | --- |
| CLIENTE | Empresa de videojuegos |
| USUARIO | Usuario administrador  Usuario jugador |
| REQUERIMIENTOS FUNCIONALES | * + - 1. Registro de un jugador       2. Registro de un nivel       3. Registro de un tesoro       4. Registro de un enemigo       5. Asociación de un tesoro a un nivel       6. Asociación de un enemigo a un nivel       7. Modificar el puntaje de un jugador.       8. Identificar el nivel para un jugador.       9. Informar los tesoros y enemigos (separados por coma) de un nivel dado por el usuario       10. Informar la cantidad encontrada de un tesoro en todos los niveles, es decir si el usuario quiere saber cuántos diamantes existen en todos los niveles.       11. Informar la cantidad encontrada de un tipo de enemigo en todos los niveles.       12. Informar el tesoro más repetido en todos los niveles.       13. Informar el enemigo que otorga mayor puntaje y el nivel donde se ubica.       14. Informar la cantidad de consonantes encontradas en los nombres de los enemigos del juego.       15. Informar el top 5 de los jugadores de acuerdo al puntaje.       16. Selección de la resolución de pantalla.       17. Clasificación de la dificultad de un nivel. |
| CONTEXTO DEL PROBLEMA | Una empresa de videojuegos necesita construir un videojuego que consta de 10 niveles en los que el jugador recoge tesoros y pelea con enemigos. |
| REQUERIMIENTOS NO FUNCIONALES | **RNF1**. El despliegue de los tesoros y enemigos de un nivel, en la aplicación web no tarde más de 2 seg.  **RNF2**. El sistema debe funcione}ar tanto como aplicación web como en una app para móviles. |

**Tabla de análisis de requerimientos funcionales (Nota: Una tabla por cada requerimiento funcional)**

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF1. Registro de un jugador | | |
| Resumen | El sistema debe permitir el registro de jugadores nuevos. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| nickname | String |  |
| name | String |  |
| Actividades generales necesarias para obtener los resultados | Lectura de datos.  Verificación de que el nuevo nickname no coincida con el nickname de algún jugador registrado.  Se instancia un objeto de tipo Player con los datos dados por el usuario y los valores predeterminados de puntaje y vidas  Se almacena este objeto en el array de tipo Player en el controller | | |
| Resultado o postcondición | Registro del jugador en el array de jugadores del controller. | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de alerta | String | Registro de jugador |

**Tabla de análisis de requerimientos funcionales (Nota: Una tabla por cada requerimiento funcional)**

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF2. Registro de un nivel | | |
| Resumen | El sistema debe permitir el registro de un nivel del videojuego, se deben generar 10 niveles de acuerdo a un puntaje inicial y una constante de crecimiento dadas por el usuario | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| requiredScore | int |  |
| growthConstant | double |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Lectura de datos requeridos 2. Se instancia 10 objetos de tipo Level de acuerdo al puntaje inicial dado por el usuario y la constante de crecimiento, cada uno con un identificador generado automáticamente 3. Se guardan los objetos en el array de tipo Level del controller. | | |
| Resultado o postcondición | Registro de 10 niveles del juego | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de registro de nivel | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF3. Registro de un tesoro | | |
| Resumen | El Sistema debe permitir el registro de los tesoros, estos son el mecanismo para el personaje de aumentar el puntaje y subir de nivel. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| name | String |  |
| linkUrl | String |  |
| remPoints | int |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Lectura de datos requeridos 2. Se instancia el objeto de tipo Treasure 3. Se guardan el objetos en el array de tipo Treasure del controller. | | |
| Resultado o postcondición | Registro de un tesoro | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de registro de un tesoro | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF4. Registro de un enemigo | | |
| Resumen | El Sistema debe permitir el registro de un enemigo, que es aquel que disminuye el puntaje del jugador. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| name | String |  |
| monsterIndicative | Byte |  |
| negativePoints | int |  |
| remPoints | int |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Lectura de datos 2. Asignación de un tipo de acuerdo al monsterIndicative dado por el usuario. 3. Se crea un objeto de la clase Monster con los atributos dados por el usuario 4. Se guarda este objeto en el array de tipo Monster del controller | | |
| Resultado o postcondición | Registro de un monstruo | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de registro de un monstruo | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF5. Asociación de un tesoro a un nivel | | |
| Resumen | El Sistema debe permitir asociar un Tesoro previamente creado a un nivel determinado. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idLevel | int |  |
| idTreasure | int |  |
| amountTreasures | int |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se muestra en pantalla los objetos de la clase Level con su respectivo indicador. 2. Se muestra en pantalla los objetos de la clase Treasure con su respectivo indicador. 3. Lectura de los identificadores de posición de los objetos de la clase Level y Treasure elegidos por el usuario. 4. Lectura de la cantidad de un objeto Treasure que quiere un usuario en un nivel. 5. Se crean las copias del objeto Treasure. 6. Se almacenan las copias en el array de tipo Treasure del nivel escogido por el usuario. | | |
| Resultado o postcondición | Asociación de los objetos de la clase Treasure en el array que administra los tesoros de un nivel | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de asociación de tesoro | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF6. Asociación de un enemigo a un nivel | | |
| Resumen | El Sistema debe permitir asociar un enemigo previamente creado a un nivel determinado. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idLevel | int |  |
| idMonster | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se muestra en pantalla los objetos de la clase Level con su respectivo indicador. 2. Se muestra en pantalla los objetos de la clase Monster con su respectivo indicador. 3. Lectura de los identificadores de posición de los objetos de la clase Level y Monster elegidos por el usuario. 4. Se verifica que el objeto de la clase Monster no esté asociado previamente en el array de monstruos del nivel. 5. Se crea una copia del objeto tipo Monster seleccionado. 6. Se almacena la copia en el array que contiene los monstruos del objeto de la clase Level seleccionado por el usuario. | | |
| Resultado o postcondición | Asociación de la copia del objeto tipo Monster en el array que administra los monstruos del objeto de la clase Level escogido por el usuario | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de asociación de monstruo | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF7. Modificar el puntaje de un jugador. | | |
| Resumen | El sistema le debe permitir al administrador modificar el puntaje de un jugador. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idPlayer | int |  |
| newScore | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se muestra en pantalla el listado de jugadores con su indice 2. Lectura de datos 3. Se modifica el puntaje del jugador seleccionado con el valor digitado por el usuario | | |
| Resultado o postcondición | Modificación del puntaje del jugador | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de cambio de puntaje | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF8. Identificar nivel para un jugador | | |
| Resumen | El sistema debe permitir incrementar nivel para un jugador, en caso que no pueda incrementar el nivel, debe informar al usuario que puntaje requiere para subir. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idPlayer | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Lectura de datos 2. Se compara el atributo initialScore del objeto de la clase Player con el atributo requiredScore de cada nivel del arreglo de niveles del controller. 3. Cuando el atributo initialScore sea mayor o igual a el atributo requiredScore de un nivel, se asigna al objeto Player elegido por el usuario un nivel superior al nivel donde se cumplió la condición. 4. El nivel queda registrado en el atributo currentLevel del objeto de clase Player elegido por el usuario | | |
| Resultado o postcondición | Identificación y asignación del nivel correspondiente de un jugador según su puntaje y el puntaje requerido de los niveles | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de nivel de jugador | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF9. Informar los tesoros y enemigos de un nivel | | |
| Resumen | El sistema debe informar los tesoros y enemigos de un nivel dado por el usuario | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idLevel | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se muestra en pantalla los niveles del juego con su índice 2. Lectura de datos 3. Se recorre la lista de enemigos del objeto Level y se compara con los objetos de clase Monster en la lista del controller. 4. Se recorre la lista de tesoros del objeto Level y se compara con los objetos de clase Treasure en la lista del controller. 5. Se forma un mensaje con el nombre del tesoro y su respectiva cantidad en el nivel, con el nombre del enemigo y su respectiva cantidad en el nivel | | |
| Resultado o postcondición | Informe los tesoros y enemigos de un nivel | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Informe los tesoros y enemigos | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF10. Informar la cantidad encontrada de un tesoro en todos los niveles | | |
| Resumen | Informar la cantidad encontrada de un tesoro en todos los niveles, es decir si el usuario quiere saber cuántos diamantes existen en todos los niveles. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idTreasure | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se muestra en pantalla la lista de tesoros del juego. 2. Lectura de datos. 3. Se recorre los arreglos que administran los tesoros de cada objeto tipo Level contando la cantidad del tesoro escogido por el usuario 4. Se almacena este contador. | | |
| Resultado o postcondición | Informe de la cantidad encontrada de un tesoro en todos los niveles | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje con la cantidad de un tesoro | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF11. Informar la cantidad encontrada de un tipo de enemigo en todos los niveles. | | |
| Resumen | Informar la cantidad encontrada de un tipo de enemigo en todos los niveles, es decir si el usuario quiere saber cuántos ogros existen en todos los niveles. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| idEnemyType | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Lectura de datos 2. Se recorre los arrays que administran los enemigos de cada objeto tipo Level almacenados en el controller 3. Se lleva un conteo de los objetos tipo Monster que coincida con el tipo seleccionado por el usuario 4. Se genera un mensaje con el informe de la cantidad de objetos de la clase Monster por el tipo elegido | | |
| Resultado o postcondición | Informe de cantidad de objetos de la clase Monster por el atributo type elegido en todos los niveles. | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Informe de cantidad de enemigos por tipo elegido | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF12. Informar el tesoro más repetido en todos los niveles. | | |
| Resumen | Informar el tesoro más repetido en todos los niveles. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. El sistema toma un objeto Treasure como el más repetido. 2. Se entra en una estructura repetitiva que compara las cantidades de determinado objeto Treasure en todos los niveles con la cantidad del objeto Treasure tomado como el más repetido. 3. Una vez finalizada todas las iteraciones, se obtiene el objeto Treasure más repetido. | | |
| Resultado o postcondición | Informe del tesoro más repetido de todos los niveles | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| mostrepeatedTreasure | int |  |
| Informe del tesoro más repetido | String |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF13. Informar el enemigo que otorga mayor puntaje y el nivel donde se ubica. | | |
| Resumen | El Sistema debe informar el enemigo que otorga mayor puntaje y el nivel donde se ubica. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se recorre el arreglo de objetos clase Monster del controller comparando su atributo remPoints para haller el enemigo que otorga mayor puntaje. 2. Una vez obtenido el objeto clase Monster que más remunera, se recorren todos los arreglos que administran enemigos en cada nivel para hallar en donde se encuentra este determinado enemigo. 3. Se forma el mensaje con el nombre del objeto tipo Monster y los niveles en que se encuentra | | |
| Resultado o postcondición | Informe del enemigo que otorga mayor puntaje y el nivel donde se ubica. | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Informe del enemigo que otorga mayor puntaje y el nivel donde se ubica. | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF14. Informar la cantidad de consonantes encontradas en los nombres de los enemigos del juego. | | |
| Resumen | El Sistema debe indicar la cantidad de consonantes encontradas en los nombres de los enemigos del juego. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se recorre el arreglo que contiene los objetos tipo Monster del controller. 2. Se obtiene de cada enemigo su nombre y se genera un arreglo de tipo char con cada carácter del nombre del enemigo. 3. Se hace una comparación de cada elemento del arreglo tipo char creado con lista de vocales y espacio. 4. En caso de que no coincida se aumenta un contador. 5. Se forma el mensaje con el nombre del objeto de clase Monster y la cantidad de consonantes que tiene. | | |
| Resultado o postcondición | Informe de la cantidad de consonantes encontradas en los enemigos del juego | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Informe de la cantidad de consonantes encontradas en los enemigos del juego | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF15. Informar el top 5 de los jugadores de acuerdo al puntaje. | | |
| Resumen | El Sistema debe informar el top 5 de los jugadores del juego en base a su puntaje | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se recorre el arreglo de objetos tipo Player del controller comparando su atributo de puntaje. 2. Se entra en estructura repetitiva donde se obtiene el mayor score, y en cada iteración se excluye al anterior. 3. Los 5 mejores puntajes se guardan en orden en un arreglo de tipo int. | | |
| Resultado o postcondición | Arreglo con el top 5 de jugadores según el puntaje | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Informe del top 5 de jugadores | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF16. Selección de la resolución de pantalla. | | |
| Resumen | El sistema debe preguntar al usuario cual es la resolución sobre la cual se ejecutará el juego. Ejemplo (El usuario escoge la resolución QHD) y esta información será utilizada al momento de generar randómicamente las posiciones en pantalla. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| resolutionIndicative | int |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se muestra en pantalla las distintas resoluciones 2. Lectura de datos 3. Asignación de la resolución escogida a través de una enumeración en un atributo del controller. | | |
| Resultado o postcondición | Selección de la resolución de pantalla | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje de asignación de resolución | String |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre o identificador | RF17. Clasificación de la dificultad de un nivel. | | |
| Resumen | Para cada nivel, se define el nivel de complejidad (alto, medio, bajo) de acuerdo a los tesoros y los enemigos registrados, si los puntos que otorgan los tesoros es mayor que los puntos de los enemigos es nivel bajo, si es igual es medio, y si son más los puntos de los enemigos que los puntos de los tesoros es alto. | | |
| Entradas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Actividades generales necesarias para obtener los resultados | 1. Se obtiene una sumatoria de los puntos que otorgan los tesoros de un determinado nivel. 2. Se obtiene una sumatoria de los puntos que otorgan los monstros de un determinado nivel. 3. Se comparan ambas sumatorias, bajo el criterio de si los puntos que otorgan los tesoros es mayor que los puntos de los enemigos es nivel bajo, si es igual es medio, y si son más los puntos de los enemigos que los puntos de los tesoros es alto, y de acuerdo a la comparación se asigna la dificultad al nivel. 4. La dificultad del nivel se guarda en el atributo del objeto de la clase Level. | | |
| Resultado o postcondición | Clasificación de la dificultad de un nivel.  Guardado de esta dificultad como un atributo del objeto clase Level | | |
| Salidas | Nombre entrada | Tipo de dato | Condición de selección o repetición |
| Mensaje con la dificultad de un nivel | String |  |
| levelDiff | Difficulty |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Requerimiento Funcional** | **Nombre de la Clase** | **Nombre del método** |
| RF1. Registrar de un jugador | Clase GameManager | regPlayer() |
| Clase GameController | registerPlayer(String nickname, String name) |
| Clase GameController | verifyNickname(String nicknameToVerify) |
| Clase Player | Player(String nickname, String name) |
| RF2. Registrar de un nivel | Clase GameManager | regLevel() |
| Clase GameController | registerLevel(int growthConstant, double requiredScore) |
| Clase Level | Level(int requiredScore) |
| RF3. Registro de un tesoro | Clase GameManager | regTreasure() |
| Clase GameController | registerTreasure(String name, String linkUrl, int remPoints) |
| Clase Treasure | Treasure(String name, String linkUrl, int remPoints) |
| RF4. Registro de un enemigo | Clase GameManager | regMonster() |
| Clase GameController | registerMonster(String name, byte monsterIndicative, int negativePoints, int remPoints) |
| Clase Monster | Monster(String name, EnemyType type, int negativePoints, int remPoints) |

|  |  |  |
| --- | --- | --- |
| RF5. Asociación de un tesoro a un nivel | Clase GameManager | associateTreasure() |
| Clase GameController | showLevels() |
| showTreasures() |
| associateTreasureToLevel(int idLevel, int idTreasure, int amountTreasures) |
| generatePosition(int) |
| Clase Levels | getListOfTreasuresInLevel() |
| Clase Treasure | Treasure (Treasure treasureToCopy) |
| RF6. Asociación de un enemigo a un nivel | Clase GameManager | associateMonster() |
| Clase GameController | showLevels() |
| showMonsters() |
| associateMonsterToLevel(int idLevel, int idMonster) |
| generatePosition(int) |
| Clase Level | getListOfTreasuresInLevel() |
| Clase Monster | Monster (Monster monster ToCopy) |
| RF7. Modificar el puntaje de un jugador | Clase GameManager | modifyPlayerScore () |
| Clase GameController | modifyPlayerScore(int idPlayer, int newScore) |
| Clase Player | setInitialScore(int initialScore) |
| RF8. Identificar el nivel de un jugador | Clase GameManager | identifyPlayerLevel() |
| Clase GameController | identifyPlayerLevel(int) |
| Clase Player | getInitialScore() |
| setCurrentLevel(int) |
| getCurrentLevel() |
| Clase Level | getCurrentLevel() |
| RF9. Informar los tesoros y enemigos en un nivel | Clase GameController | informMonstersTreasuresLevel(int) |
| Clase Level | getListOfMonstersInLevel() |
| Clase Monster | getName() |
| Clase Level | getListOfTreasuresInLevel |
| Clase Treasure | getName() |
| Clase GameManager | informTreasureMonstersInLevel () |
| RF10. Informar la cantidad encontrada de un tesoro en todos los niveles | Clase GameController | informAmountTreasures(int) |
| Clase Treasure | equals(Object) |
| Clase Levels | getListOfTreasuresInLevel() |
| Clase GameManager | informAmountTreasures() |
| GameController | showTreasures() |
| RF11. Informar la cantidad encontrada de un tipo de enemigo en todos los niveles, | Clase GameManager | informAmountMonstersType() |
| Clase GameController | informAmountMonstersType(int) |
| Clase Levels | getListOfMonstersInLevel() |
| Clase Monster | equals(Object) |
|  |  |  |
| RF12. Informar el tesoro más repetido en todos los niveles. | Clase GameController | informMostRepeatedTreasure() |
| informAmountTreasures(int) |
| Clase GameManager | informMostRepeatedTreasure() |
| RF13. Informar el enemigo que otorga mayor puntaje y el nivel donde se ubica. | Clase GameController | bestRemMonster() |
| Clase Monster | getRemPoints() |
| Clase Level | getListOfMonstersInLevel() |
| Clase Monster | equals() |
| RF14. Informar la cantidad de consonantes encontradas en los nombres de los enemigos del juego. | Clase GameController | numberConsonantsMonster() |
| Clase Monster | getName() |
| RF15. Informar el top 5 de los jugadores de acuerdo al puntaje | Clase GameController | informTopPlayers() |
| RF16. Selección de la resolución de pantalla. | Clase GameController | setResolution(int) |
| Clase GameManager | resolution() |
| Clase Resolution | values() |
| RF17. Clasificación de la dificultad de un nivel. | Clase GameController | setLevelDifficulty(int) |
| Clase Level | getListOfTreasuresInLevel() |
| getListOfMonstersInLevel() |
| Clase Treasure | getRemPoints() |
| Clase Monster | getRemPoints() |