|  |  |  |
| --- | --- | --- |
| **Nombre**: Diego Álvarez Gamboa  Maria Fernanda Villegas Casco | | **Matrícula**: 2758345  2817368 |
| **Nombre del curso:**  Programación Orientada a Objetos | **Nombre del profesor**:  [María Suei Chong Sol](https://cursos.tecmilenio.mx/courses/14745/users/27690) | |
| **Módulo**: modulo 3 | **Actividad**: Villa virtual | |
| **Fecha**: 7 de mayo del 2020 | | |
| **Bibliografía:** | | |

Procedimiento:

1. Identifica las clases, objetos y métodos que se requieren.
2. Diseña el diagrama de clases que represente la relación entre clases, objetos y métodos en la villa virtual.
3. Genera el código en lenguaje Java que represente el diagrama de clases.

Descripción:

Existe una villa virtual que ha sido hechizada por un mago poderoso. Este mago obtiene su poder del polvo de hadas que mantiene en cautiverio. El poder de un mago se contabiliza de manera numérica y aumenta en una unidad por cada 10 hadas que capture.

El papel del jugador principal es construir casas para rescatar un hada y llevarla a vivir ahí. Cada vez que un hada es rescatada, el poder del mago disminuye. Por cada 10 hadas que rescates, se resta una unidad de poder.

El jugador principal tiene un nivel de vida que aumenta cada vez que se rescatan 10 hadas y disminuye cuando es golpeado por el ogro que protege al mago.

Resultados:

Diagrama UML

Imagen que contiene texto

Descripción generada automáticamente

Análisis:

Nuestro juego inicia preguntando los atributos del nombre del personaje principal (jugador) y del Ogro. De ahí le explica al jugador cómo funciona, que se empieza con una barra de vida de 100, mientras que hay 12 hadas repartidas en 4 casas diferentes, pero capturadas por el mago. El juego empieza cuando el jugador escoge una categoría: 1.

* La categoría uno son sumas y restas que se generan en un método el cual tiene como retorno un punto( si la suma o resta que digita el usuario es correcta se da, o en caso de ser incorrecta se restan 2), ahí pasa por condicionales para haber si se suma o resta la vida del jugador o del mago, así como si se capturan o rescatan hadas. El ogro es invocado cuando el resido entre 8 de la vida del mago es 0, para que aparezca de manera aleatoria a ojos del usuario. En este nivel, se rescata de hada en hada y la vida baja de 5 en 5
* La categoría 2 son puras multiplicaciones y divisiones, las cuales también se obtienen gracias a los números random, y únicamente se toma en las divisiones los números antes del punto en caso de tener decimales. Sucede igual que en la opción uno en cuanto al retorno, los puntos y la vida. Aquí te baja más la vida, de 10 en 10 Pero se recuperan o pierden de 6 en 6 las hadas Cada 6 hadas, se hace una casa.
* La categoría 3 son derivadas, si contesta bien ganas pero si contesta mal, pierdes.

Al final se muestra tu vida, las veces que te atacó el ogro, el poder final del mago y un gracias

SumasRestas()

|  |  |  |
| --- | --- | --- |
| Entrada | Procedimento | Salida |
| \_\_ | Se crean 2 números random del 1 al 2  Random Aleatorio= new Random();  int SoR =1 + Aleatorio.nextInt(2);  Si da 1{ Se suman 2 números aleatorios  int num1 =1 + Aleatorio.nextInt(1000);  int num2 =1 + Aleatorio.nextInt(100);  suma= num1+num2;  }  Si no { Se restan 2 números aleatorios  int num1 =1 + Aleatorio.nextInt(1000);  int num2 =1 + Aleatorio.nextInt(100);  resta= num1- num2;}  Si el usuario digita el resultado correcto, se regresa un punto, de no ser así se restan 2 | Retorna Punto  Punto= 1  O  Punto= -1 |

MultiyDivision()

|  |  |  |
| --- | --- | --- |
| Entrada | Procedimento | Salida |
| \_\_ | Se crean 2 números random del 1 al 2  Random Aleatorio= new Random();  int SoR =1 + Aleatorio.nextInt(2);  Si da 1{ Se multiplican 2 números aleatorios  int num1 =1 + Aleatorio.nextInt(1000);  int num2 =1 + Aleatorio.nextInt(100);  multi= num1\*num2;  }  Si no { Se dividen 2 números aleatorios  int num1 =1 + Aleatorio.nextInt(1000);  int num2 =1 + Aleatorio.nextInt(100);  divi= num1/ num2;}  Si el usuario digita el resultado correcto, se regresa un punto, de no ser así se restan 2 | Retorna Punto  Punto= 1  O  Punto= -1 |

Derivadas()

|  |  |  |
| --- | --- | --- |
| Entrada | Procedimento | Salida |
| \_\_ | Se crean 2 números random del 1 al 2  Random Aleatorio= new Random();  int SoR =1 + Aleatorio.nextInt(2);  Se guarda en una variable como derivada, multiplicando el num 1\* 1  int der=(num1\*1);  Si el usuario digita el resultado correcto, se regresa un punto, de no ser así se restan 2 | Retorna Punto  Punto= 1  O  Punto= -1 |

Una captura de pantalla de una red social

Descripción generada automáticamente

Una captura de pantalla de una red social

Descripción generada automáticamente

Una captura de pantalla de una red social

Descripción generada automáticamente

Conclusiones:

Mafer: Gracias a este videojuego pude poner en práctica mis conocimientos, recordé lo esencial en Java, vi cómo se relacionan nuevamente entre las herencias las clases. Plasmé algo más concreto a un lenguaje de programación. Aprendí también que hay que cerrar los proyectos y no solamente en la ventana derecha, si no, también en la izquierda, recordé la importancia del orden y de los comentarios para poder hacer un poco más fácil el proceso de la creación

Diego: Al realizar este proyecto fue complicado ya que de principio no recordaba bien como hacer las cosas en java y me confundía y las ponía como si lo estuviera haciendo en Python, el análisis para el diagrama UML fue bastante complicado hay que no especifica al 100 todas los personajes, atributos y comportamientos, pero una vez definidos se realizó con más rapidez y surgieron otras ideas.