**Objetivos específicos**

OE1.1. Analizar problemas mediante la especificación, a través de contratos, de entradas, salidas, ejemplos y casos de prueba.

OE1.2. Modelar información relevante a la solución del problema empleando variables, constantes, tipos de datos primitivos y cadenas de texto.

OE1.3. Resolver problemas utilizando estructuras de control: instrucciones secuenciales, subrutinas, condicionales y repetitivas.

OE1.4. Utilizar operadores (de asignación, aritméticos, relacionales, de cadenas y lógicos), estructuras contenedoras lineales de tamaño fijo (de tipos de datos primitivos) y cadenas de texto en la construcción de soluciones.

OE1.5. Utilizar un ambiente de desarrollo (incluyendo la compilación y ejecución de programas desde consola) y un espacio de trabajo predefinido, para construir la solución de un problema.

OE1.6. Codificar en lenguaje Java la solución a un problema a partir de los contratos de solución propuestos en las etapas de análisis y diseño.

OE1.7. Utilizar objetos e invocar métodos estáticos de clases del API de Java en la construcción de soluciones implementadas con interfaces gráficas por consola.

OE1.8. Interpretar y resolver errores producidos en tiempo de ejecución (ej.: posición por fuera del rango en una estructura contenedora, llamados u operaciones con objetos que no han sido construidos, etc.).

OE1.9. Instalar y configurar las herramientas Git y GitHub para definir los repositorios local y remoto para albergar y hacer control de versiones de las soluciones propuestas.

**Actividades**

Lleve a cabo las siguientes actividades de cada una de las etapas de desarrollo de software:

1. Análisis del problema
   1. Definición del diagrama de flujo general que represente la solución planteada.
   2. Definición de los contratos de las subrutinas o métodos.
2. Implementación en Java. Incluya en la implementación, los comentarios descriptivos sobre el código java escrito por usted.
3. Generación del API de la solución usando JavaDoc.

**Entregables**

Usted debe subir su proyecto como un repositorio de github a través de Github Classroom. Tenga en cuenta que su proyecto debe tener la siguiente estructura:

DamageCalculator /

src/

bin/

doc/

Dentro de los directorios src/ y bin/ estarán presentes estos directorios, representando cada uno de sus paquetes:

ui/

model/

Contenido en los directorios:

* El directorio src (source code) contiene los archivos .java dentro del directorio ui (por ahora).
* El directorio bin (binary files) contiene los archivos .class en el directorio ui.
* El directorio doc contiene toda la información de análisis del problema y los archivos html generados por el javadoc

A continuación, encontrará un enunciado que narra de forma detallada la situación problemática que se espera usted solucione.

**Enunciado: Calculadora de daño y reacciones en el juego Genshin Impact**

**Genshin Impact**, es un juego de rol de acción gratuito desarrollado por HoyoVerse. El juego presenta un entorno de mundo abierto de fantasía y un sistema de combate basado en acción que utiliza magia elemental, tipos de armas y cambios de personajes.

Alicia, es una jugadora apasionada del juego que ha decidido participar en las ***“Olimpiadas de mejor daño provocado en Genshin Impact ”*** y ha escuchado que ahora usted es un/una entusiasta que usa Java. Alicia le ha contactado para **desarrollar una calculadora de daño que le permita poder experimentar como podría alcanzar el daño más alto en el juego** según las siguientes especificaciones:

Durante un combate, los personajes pueden causar daño a través de diferentes reacciones dependiendo del uso que el jugador le otorgue a la energía elemental. Estas reacciones se dividen en **transformativas, amplificativas y aditivas**; y son capaces de otorgar multiplicadores logrando escalar el daño que provocan los personajes.

Cada personaje cuenta con estadísticas diferentes pero las más importantes relacionadas al daño son **la maestría elemental (ME), la probabilidad de daño crítico (PDC) y la cantidad porcentual de daño crítico (DC).** Además, para las resacciones transformativas el daño también se ve influenciado por el **multiplicador de resistencia** de los enemigos. El daño base de un personaje se calcula de la siguiente manera:

A continuación se explica en detalle la condición para que se libere cada tipo de reacción, su fórmula y su lista de los multiplicadores.

**Transformativas:** En este tipo de reacciones se incluyen la mayoría de las reacciones (sobrecarga, torbellino, etc.). El daño de estas reacciones NO puede ser crítico y depende directamente de la maestría elemental, transformando así el daño resultante a través de la reacción misma.

donde,

Los***Multiplicadores Transformativos*** son los siguientes: 0.25 para quemadura, 0.50 para superconductor, 0.6 para torbellino, 1.2 para electro cargado, 1.5 para cristalización, 2.0 para sobrecargado, 2.0 para florecimiento, 3.0 para sobre florecimiento y super quemadura.

**Amplificativas:** El daño de estas reacciones SI puede ser crítico y amplifican el daño base calculado según el personaje. Además de depender de la maestría elemental, la probabilidad crítica y el daño crítico.

donde,

Finalmente el daño de estas reacciones se calcula como:

Los ***Multiplicadores de reacción Amplificación***son los siguientes: 1.5 para vaporización y 2.0 para derretir.

**Aditivas:** El daño de estas reacciones SI puede ser crítico y añaden un porcentaje de daño adicional al daño base calculado según el personaje.

donde,

Los ***Multiplicadores Aditivos***son los siguientes: 1.15 para intensificación y 1.25 para propagación.

La lógica de daño anterior y sus respectivos cálculos fue extraída y adaptada de: <https://genshin-impact.fandom.com/wiki/Damage>

**Requerimientos funcionales**

Para poder ayudar a Alicia en su preparación para las olimpiadas, es necesario que usted realice las siguientes tareas dentro se su implementación:

1. Implementar las fórmulas de daño para CADA UNO de los tipos de reacciones: daño transformativo, daño amplificador y daño aditivo. **Nota:** debe tener en cuenta los cálculos secundarios que deben realizarse para completar estas fórmulas, como el cálculo del daño base y del porcentaje de bono de maestría elemental de cada reacción.
2. Calcular el daño realizado por un personaje o por un equipo (un equipo está compuesto de 4 personajes), según un tipo de reacción y según el multiplicador de resistencia de un enemigo. Este último debe ser generado de manera aleatoria en un rango entre 0.5 y 2 únicamente.
3. Implementar un sistema de memoria simple (usando arreglos), que le permita almacenar los valores de los últimos 10 daños registrados en el sistema para su posterior consulta.
4. Mostrar el daño más alto realizado en la calculadora hasta el momento.

**Fechas de entrega**

Entrega de Análisis y Diseño: Hasta el 19 de agosto de 2023 a las 21:59.

* Su entrega debe contener el análisis y diseño del programa a desarrollar:
  + Análisis: descripción, entradas, salidas y al menos un ejemplo.
  + Diseño: una primera versión del diagrama de flujo y los contratos de los métodos.

Primera entrega: Hasta el 26 de agosto de 2023 a las 21:59.

* Su entrega debe contener el análisis y diseño, y una primera versión de la documentación y la implementación del programa a desarrollar.

Segunda entrega: Hasta el 9 de septiembre de 2023 a las 21:59.

* Su entrega debe estar completa con todos los entregables de análisis, diseño, documentación y codificación.

**Rúbrica (**[**enlace**](https://docs.google.com/spreadsheets/d/1cyUmTMk1Bv9hdSC31F1hvFQGnGU42Mqowv3V1q6Pslw/edit#gid=0)**)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Análisis, Diseño y Documentación (30%)** | | | | **Codificación (70%)** | | | | | | | **T**  **o**  **t**  **a**  **l** | **Bono** | **D**  **e**  **f**  **i**  **n**  **i**  **t**  **i**  **v**  **a** |
| El programa cuenta con una descripción, entradas, salidas y al menos un ejemplo | Se presenta el diagrama de flujo general del programa | El programa tiene definidos los contratos de las subrutinas / métodos | Generación de la documentación en javadoc  (ver tabla en enlace) | Buenas prácticas  (ver tabla en enlace) | Compilación y Ejecución del Código  (ver tabla en enlace) | El programa muestra y permite seleccionar cada una de las fórmulas de daño para cada tipo de reacción: **daño transformativo, daño amplificador y daño aditivo.** Además estas están implementadas correctamente. | El programa implementa correctamente el sistema de memoria simple que almacena los cálculos más recientes hechos por el usuario. | El programa permite calcular el daño realizado por un personaje o un equipo | El programa permite consultar el daño más alto obtenido hasta el momento. | Se presenta gráficamente por consola un menú que permite al usuario interactuar con el programa. Este menú se despliega indefinidamente hasta que el usuario decide salir del programa seleccionando la opción correspondiente | El análisis (descripción, entradas, salidas y ejemplo) se encuentra en Inglés |
| **5%** | **10%** | **10%** | **5%** | **5%** | **5%** | **15%** | **15%** | **10%** | **10%** | **10%** | **100%** | **10%** | **110%** |