Laboratorio 6
Programación 1

Primer Semestre 2017

1. Escribir una clase que permita representar al personaje de un videojuego. Un personaje tiene un nombre (o nickname), un porcentaje de vida (o salud), un poder (su nombre, por ejemplo, “patada giratoria”), y una medida de daño (número entero entre 0 y 100).

    1. Implementar un método que permite imprimir la información de un personaje utilizando la instrucción print()

    2. Implementar un método que permite atacar a otro personaje (que se recibe cómo parámetro). El ataque del personaje (p1) le quita vida al personaje que es atacado (p2), utilizando la siguiente función:

        ```
        nueva_vida(p2) = vida_actual(p2) - medida_de_daño(p1)
        ```

    3. Implementar un método que indica (devolviendo True) si un personaje está con vida (salud > 0)

    4. Crear 3 personajes llamados pj1, pj2 y pj3 (con el porcentaje de salud y poder que ustedes desee), pj1 debe atacar a pj2 y pj3

2. En programación orientada a objetos uno de los pilares fundamentales es el encapsulamiento. Es buena práctica utilizar métodos de acceso a las propiedades (conocidos comúnmente como getters y setters). Agregar getters y setters a todas las propiedades de las clases anteriores. Los setters deben mantener la consistencia de la información, por ejemplo no se debe permitir modificar la salud de un personaje y que la misma sea negativa.

3. Implementar una clase para identidades de personas. La clase debe tener los componentes usuales para la identificación de las personas como el número de la cédula, una dirección, nombres y apellidos, número de teléfono, etc. Se debe incluir métodos de acceso a todas las propiedades.

4. Implementar una clase modelando una cuenta bancaria. La cuenta bancaria tiene un saldo, un número de cuenta, e información sobre el dueño, que es una instancia de la clase anterior (ejercicio 3).

    La clase debe implementar los métodos de hacer un depósito o una extracción de dinero. El método de extracción de dinero debe rechazar la transacción si no hay fondos suficientes en la cuenta.

También desarrollar una función transferencia que mueve un monto entre una y otra cuenta (Para esto se puede realizar una extracción seguido por un depósito). Esos métodos y funciones deben devolver un valor lógico indicando si se fue posible ejecutar la transacción.

5. Crear una clase Hora con atributos para las horas, los minutos y los segundos de una hora.

    1. Construir un constructor sin parámetros con el 00:00:00 como valores por omisión.
    2. Construir un método “validar” que comprobará si la hora es correcta; si no lo es la ajustará al valor por omisión.
    3. Construir métodos que permitan acceder y modificar las horas, los minutos y los segundos de un objeto Hora (getters y setters).
    4. Construir un método que permita representar en un string la hora: mostrará la hora: (07hs. 03ms. 21s.).
    5. Construir un método “aSegundos” que devolverá el número de segundos transcurridos desde las 0 horas.
    6. Construir un método “deSegundos(int)” que hará que la hora sea la correspondiente a haber transcurrido, desde las 0 horas, los segundos que se indiquen como parámetro.
    7. Construir un método “segundosDesde(hora)” que devolverá el número de segundos entre la hora y la hora proporcionada como parámetro.
    8. Construir un método “siguiente” que pasará al segundo siguiente.
    9. Construir un método “anterior” que pasará al segundo anterior.
    10. Construir un método “copia” que devolverá un objeto clonado de la hora.