Departamento de Informática
Universidad Carlos III de Madrid

Grado en Ingeniería en Informática

Programación
Examen final
22 de Enero del 2016

Normas generales del examen

- El tiempo para realizar el examen es de **3 horas**
- Si se sale del aula, no se podrá volver a entrar durante el examen
- No se puede presentar el examen escrito a lápiz
- No olvides poner tu nombre, NIA y grupo
- No utilizar esta hoja para las respuestas

Problema 1 (5 puntos)

Un juego de dados al que pueden jugar varios jugadores a la vez consiste en lo siguiente. Todos los jugadores tienen un mismo número de dados. Los jugadores hacen su jugada tirando todos sus dados en la mesa. El ganador o ganadores de cada ronda se determinan de la siguiente forma:

1. Se seleccionan los dados de mayor valor de cada jugador.
2. El jugador con el dado más alto gana.
3. Si hay empate se repite 1. con el resto de dados.

Por ejemplo, si hay tres jugadores y tres dados, en una ronda con el siguiente resultado: Jugador 0 $\rightarrow$ 2 3 5; Jugador 1 $\rightarrow$ 5 3 1; Jugador 2 $\rightarrow$ 4 1 1, el ganador sería el Jugador 0, ya que los dados de mayor valor tienen valor 5, y corresponden a los Jugadores 0 y 1. El siguiente dado de mayor valor tiene un 3 para ambos jugadores, por lo que no sirve para desempatar. Finálmente, en el siguiente dado el Jugador 0 tiene un 2, mientras que el Jugador 1 tiene un 1, por lo que el Jugador 0 gana.

El objetivo es hacer un programa que **dados un número cualquiera de jugadores, un número cualquiera de dados y un número cualquiera de rondas** sirva para determinar qué jugador o jugadores han ganado más rondas. Para ello, el programa deberá ser capaz de: (1) simular las rondas del juego, es decir, las jugadas; (2) determinar qué jugador/es son los ganadores de una ronda; y (3) Determinar qué jugador/es han ganado la partida, es decir, un mayor número de rondas.

Se pide:

1. (1 punto) Explicar claramente el algoritmo o algoritmos que se van a utilizar para determinar quienes son los ganadores de una ronda. Se puede utilizar un ejemplo si es necesario.
2. Escribir un programa que resuelva el objetivo descrito. Sus funcionalidades de valorarán de la siguiente forma: (1 punto) Simulación de las jugadas; (2 puntos) Determinar ganadores de una ronda.; (1 punto) Determinar ganadores de la partida.

Aclaraciones:

- Se asumirá que el número de jugadores, el número de dados y el número de rondas están almacenados en tres variables de tipo entero y no es necesario pedirlos al usuario:

```
public static void main(String[] args) {
    int Njugadores = 3;
    int Ndados = 3;
    int Nrondas = 10;
    ...
```

- No se pueden utilizar librerías de Java, salvo para la generación de números aleatorios.
- El problema se puede resolver utilizando una única clase y métodos estáticos.
- Se valorará el diseño de métodos propuesto.

Problema 2 (5 puntos)

Una empresa de seguridad de software planea desarrollar un nuevo software para su trabajo diario. La empresa provee de servicios de detección y limpieza de virus a sus clientes. Algunas definiciones útiles son: Un virus es un software malicioso. Se caracteriza por su nombre, el sistema operativo en el que ejecuta, la lista de vulnerabilidades que aprovecha para infectar un programa (que se representa como una lista de números enteros), y si se puede o no limpiar sin pérdida de información. Un programa es un software legítimo. Está caracterizado por su nombre, el sistema operativo en que ejecuta y la lista de vulnerabilidades que tiene. Un cliente tiene un nombre y una lista de programas instalados. Un incidente implica un virus, un programa y un cliente, y representa el hecho de que cierto programa de un cliente ha sido infectado por un virus. Un incidente puede estar o no confirmado. Se confirma cuando es seguro que el programa está afectado por un virus.

Considerando la información descrita, **se pide**:

- (0,5 puntos) Crear las clases necesarias para representar los conceptos descritos e incluir los atributos correspondientes.
- (0,5 puntos) Crear un constructor completo para una de las clases y un ejemplo de método get y set para algún atributo. Asumiremos que el resto cuentan con el constructor completo y los métodos set y get necesarios para resolver los siguientes apartados.
- (0.5 puntos) Crear el método `incluirVulnerabilidad` que añade una nueva vulnerabilidad a la lista de vulnerabilidades aprovechadas por un virus. Si la vulnerabilidad ya existe, el método no hará nada y devolverá `false`. En otro caso, incluirá la vulnerabilidad en la lista y devolverá `true`.
- (0.5 puntos) Crear el método `aprovechaVulnerabilidad` que devuelve `true` si un virus aprovecha la vulnerabilidad recibida por parámetro. Asumiremos que un método similar denominado `tieneVulnerabilidad` existe para los programas.
- (1 punto) Crear el método `vectorDeAtaque` que dado un virus y un programa devuelva un array con las vulnerabilidades que tienen en común, en caso de que ambos ejecuten en el mismo sistema operativo. Si no hay vulnerabilidades en común o ejecutan en distinto sistema operativo el resultado será `null`.
- (1 punto) Crear el método `confirmarIncidente` que confirma un incidente que implica un cliente, un programa y un virus. Se debe chequear que el cliente es el dueño del programa, y que el programa tiene al menos una vulnerabilidad que el virus aprovecha.
- (0.5 puntos) Crear el método `toString` que devuelve un `String` con todos los datos de un incidente con el siguiente formato:

```
Confirmed: true
*Virus details:
Name: ILoveYou
Operating System: Windows
Vulnerabilities: 3 4
Can be cleaned: false
*Program details:
Name: Word
Operating System: Windows
Vulnerabilities: 1 4 6
```

- (0.5 puntos) Escribir un programa principal en el que se cree un array de incidentes. Debe contener 3 incidentes que impliquen a 3 programas, 1 virus y 2 clientes. Ejecutar la confirmación de los incidentes e imprimirlos.