Examen de Algorítmica y Complejidad (Plan 2014)

22 de enero de 2021

Nº matrícula:_____ **Nombre:** _____

Apellidos: _____

Problema. El juego de la rana (en España), juego del sapo (en Perú, Uruguay, Ecuador, Bolivia y Argentina), o simplemente rana (en Colombia); es un juego de lanzamiento de precisión múltiple donde se intenta introducir un determinado número de fichas, argollas o discos de metal en los múltiples agujeros que existen en la mesa del sapo o rana.

Si bien las partidas habitualmente consisten en 10 tiradas, donde el jugador que consigue la máxima puntuación gana, en este caso se ha decidido modificar el juego, y se trata de conseguir una *puntuación* concreta, usando el menor número de discos posibles. Dada la mesa de juego que se muestra en la imagen (que permite conseguir: {5,10,15,25,50} puntos por disco), se pide implementar un algoritmo que muestre como resultado a qué secciones de la mesa (agujeros) deben tirarse los discos para conseguir obtener la *puntuación* deseada, teniendo en cuenta que en el mismo agujero de la mesa pueden tirarse tantos discos como sea necesario.

Implementar un algoritmo en Java (*numDiscosGreedy*), basado en un esquema voraz, que, dada una *puntuación* concreta y una mesa de juego con valores desordenados, devuelva cuántos discos deben tirarse y a qué agujero, teniendo en cuenta que el número de discos a tirar debe ser el mínimo posible.

```
ArrayList<AgujeroRana> numDiscosGreedy(ArrayList<Integer> valoresRana,
                                       int puntuacion)

//Clase auxiliar: AgujeroRana
public class AgujeroRana {
    private int valor; //valor del agujero de la mesa
    private int cantidad; //número de discos introducidos en el agujero

    AgujeroRana (int valor, int cantidad){
        this.valor  = valor;
        this.cantidad = cantidad;
    }
    /* getters y setters */
}
```

```
/* El problema es equivalente al problema de las monedas (reparto mínimo de
monedas. Se implementa solución de O(N^2) */
```

```
ArrayList numDiscosGreedy (ArrayList valoresRana, int puntuacion){
   ArrayList solucion = new ArrayList();
   int valor;
   while ((puntuacion>0) &&(!valoresRana.isEmpty())) {
      valor = seleccionarCandidatoListaDesordenada(valoresRana);
      valoresRana.remove(Integer.valueOf(valor));
      if ((puntuacion/valor) >0){
         solucion.add(new AgujeroRana(valor, puntuacion/valor));
         puntuacion = puntuacion % valor;
      }
   }
   if (puntuacion==0) {
      return solucion;
   } else
      return null;
}


int seleccionarCandidatoListaDesordenada(ArrayList diana){
   //los valores de la Diana están desordenados: O(N)
   int mayor = Integer.MIN_VALUE;
   for (Integer disco : diana) {
      if ((mayor < disco))
         mayor = disco;
   }
   return mayor;
}
```