¡Por aquí ha pasado un duende!

Primera Convocatoria

Programación II VJ1208 31/05/2018

Nombre: **DNI:**

1. (4 puntos) Los duendes son unas criaturas invisibles y traviesas. Escribe las siguientes clases y métodos para ayudar a desarrollar un videojuego con duendes siguiendo los criterios expuestos en la asignatura para desarrollar programas orientados a objetos (herencia, privado/público, getters/setters/propiedades, constructores, comentarios).

a. Un `campo de juego` ocupa desde la coordenada (`0`, `0`) hasta (`XMax`,`YMax`).

b. Un `Duende` contiene información sobre su `posición` (en coordenadas `x` e `y`), `nivel de energía` (que es un entero) y un `campo de juego`. Un duende se crea con un nivel de energía de 100 unidades y necesita que se le proporcione un campo de juego para ubicarse inicialmente en una posición aleatoria válida. Además, tiene un método *`Actuar`* que devuelve la cadena "", y otro método *`Moverse`* que modifica su posición actual en X y en Y aleatoriamente entre -3 y 3 pasos sin salirse del campo y reduciendo en una unidad su nivel de energía.

c. Un duende `Revoltoso` es un tipo de duende que sobrescribe el método *`Actuar`* para devolver aleatoriamente una de las siguientes cadenas: "ji, ji", "¡ahuuu!", o "¡Hey!" si su nivel de energía es mayor que cero. En otro caso devuelve "".

d. Un duende `Coleccionista` es un tipo de duende que posee una lista de objetos de la clase `Tesoro` (no hay que desarrollar esta clase). También posee un método *`Añadir`* (tampoco hay que desarrollar este método) que recoge y almacena en su lista de tesoros un Tesoro si existe uno cerca de su posición. Un duende `Coleccionista` también debe sobrescribir el método *`Actuar`* de forma que elimine aleatoriamente de la lista un tesoro y devuelva una representación de ese Tesoro en forma de cadena. Si en la lista no hay tesoros se devuelve "".

e. Desarrolla un método `Golpear` que recibe una lista de duendes y reduce el nivel de energía en `n` unidades para aquellos duendes de esa lista que están a menos de `n` unidades de una `posición` también recibida. Un duende está a menos de `n` unidades de una posición (`x0`,`y0`) si su `x` está a menos de `n` unidades de `x0` y su `y` está a menos de `n` unidades de `y0`. Si, como consecuencia, el nivel de energía de algunos duendes de esa lista pasa a ser negativo, éstos han de ser eliminados.

f. Desarrolla un método `MensajeColeccionistas` que recibe un array de duendes y devuelve una única cadena con el resultado acumulado de ejecutar el método *`Actuar`* para cada duende Coleccionista de dicho array.

Razonar la respuesta o la respuesta no será puntuada

```
public class List<T> {
   public void Add(T item) {....}
   public void RemoveAt(int index) { …. }
   public int IndexOf(T item) { …. }
   public bool Remove(T item) { …. }
   public int Count {get;}
…
}
```

Razonar la respuesta o la respuesta no será puntuada

Razonar la respuesta o la respuesta no será puntuada

Razonar la respuesta o la respuesta no será puntuada

4

Razonar la respuesta o la respuesta no será puntuada

Razonar la respuesta o la respuesta no será puntuada

2. (1.5 puntos) Escribir un método NDuendesMayorEnergía que reciba como parámetros a un entero N y a un array de duendes y devuelva otro array de duendes formado por hasta los N duendes del array recibido que tengan mayor nivel de energía.

Razonar la respuesta o la respuesta no será puntuada

3. (1.5 puntos) a) Calcula la función de coste del siguiente método contando como operación elemental cada acceso a la matriz duendesVecinos. ¿Cuál es su orden? Describe el proceso de obtención de la respuesta.

```
public void UbicaDuendes(int contMax) { //duendesVecinos bool[N,N]
    int cont = 0;
    for(int i = 0; i < N; i++) {
        for(int j = 0; j < N; j++) {
            if(duendesVecinos[i, j]) cont++;
        }
    }
    if(cont < contMax)
        for(int j = 0; j < N; j++)
            for(int i = 0; i < j; i++)
                for(int k = 0; k < N; k++)
                   duendesVecinos[i, k] = duendesVecinos[k,j] ||
                                          duendesVecinos[i,j];
}
```

b) Contesta a las siguientes cuestiones:

1. Cuando se calcula el coste de la suma acumulada de los elementos de un array el mejor caso es cuando el array tiene un solo elemento y el peor caso es cuando el array tiene muchos elementos.

Razonar la respuesta o la respuesta no será puntuada

2. Un método m1 cuyo coste en el mejor caso es O(N*log*N) es mejor que otro método que resuelva la misma tarea cuyo coste en el peor caso es O(N^3).

4. (1.5 puntos) Escribir un método ***recursivo*** DevolverRevoltosos que recibiendo una lista de Duendes (que puede estar formada por Revoltosos o Coleccionistas) devuelva una nueva lista solo con los duendes Revoltosos de la lista recibida.

Razonar la respuesta o la respuesta no será puntuada

5. (1.5 puntos) Dada la siguiente declaración de clase:

```
public class NodoDuende {
  public Duende d;
  public NodoDuende anterior;
  public NodoDuende(Duende d) {
     this.d = d;
     this.anterior = null;
  }
}
```

Escribe otra clase `PilaNodosDuendes` que presente la funcionalidad básica de una estructura dinámica de datos tipo pila, utilice la clase `NodoDuende` y contenga los métodos `public Duende Pop()`, que devuelve el duende que corresponde de la `PilaNodosDuendes` y `public void Push(Duende d)`, que añade el duende pasado como parámetro a la `PilaNodosDuendes`. En una `PilaNodosDuendes` también se puede conocer el número de duendes que contiene.

Razonar la respuesta o la respuesta no será puntuada

Razonar la respuesta o la respuesta no será puntuada