Universidad de Costa Rica
Escuela de Ciencias de la Computación e Informática
CI-0112 Programación 1

Examen #2 de Cátedra

II-2019

Miércoles 27 de noviembre del 2019

Profesores:

- Francisco Arroyo
- Edgar Casasola
- Ricardo Gang
- Maureen Murillo
- Javier Vásquez

Observaciones generales:

1. Tiene 3 horas para entregar su solución del examen.
2. El examen es individual.
3. Es prohibido utilizar herramientas digitales o no para intercambio de documentos o comunicación.
4. El uso del celular durante la prueba implica la anulación del examen.
5. Firme la hoja de asistencia al finalizar el examen.

Observaciones específicas para examen realizado en computadora:

1. Puede utilizar cualquier material y código propio escrito por usted (prácticas, tareas, libros, apuntes) y material de Internet debidamente referenciado. Si no cumple con estas características se considerará fraude.
2. Como nombre de su proyecto utilice “ex2” junto con su carnet y nombre. Por ejemplo: ex2B82345JoseAraya.
3. En el comentario principal de cada clase, como autor indique su número de carné y su nombre.
4. Cada 20 minutos suba su solución en un archivo comprimido que contenga el código fuente al sitio de entrega del examen de su grupo.
5. En caso de fallo de energía eléctrica debe continuar su examen en papel. Se calificará la versión subida hasta ese momento a la página del grupo, así como la documentación que aporte en su cuaderno de examen.
6. La solución del examen debe subirse dentro del tiempo asignado a la página usada en su grupo. Al finalizar su examen y por motivos de seguridad guarde en su cuenta de correo, archivo drive o memoria USB una copia de la solución entregada.

Lineamientos generales de evaluación

En la evaluación de cada pregunta se tomarán en cuenta los siguientes aspectos:

1. Cumplimiento de buenas prácticas de programación.
2. Separación de responsabilidades de los objetos.
3. Correcta especificación de clases.
4. Uso correcto de constructores.
5. Uso correcto de la programación orientada a objetos.
6. Implementación correcta de la recursividad.
7. Uso correcto de arreglos de dos dimensiones.
8. Uso correcto de memoria dinámica para implementar las estructuras de datos de lista y árbol.
9. Uso correcto de estructuras de control.
10. Entrada/salida de datos.
11. Funcionalidad solicitada.

Enunciado

1) (50%) La **Mancha Par**

Escriba en Java una clase **Tablero** que posea un atributo que represente un arreglo de dos dimensiones (matriz) de enteros. Definimos como "**vecinas**" de una celda las cuatro celdas contiguas, que están ubicadas a su izquierda, derecha, arriba y abajo. Programe los siguientes métodos en su clase **Tablero**:

a) (5%) El constructor que reciba de parámetros sus dimensiones y la llene con números enteros aleatorios en un rango entre 1 y 8 inclusive.

b) (5%) Un método que muestre la matriz en un formato de dos dimensiones similar al ejemplo.

c) (35%) Un método **recursivo** que calcule "la mancha par":
int sumarManchaPar(int f, int c)
recibiendo como parámetros los indicadores de una fila y una columna de la matriz. Si la casilla indicada por los parámetros tiene un valor impar este método debe retornar cero; en caso contrario debe retornar la suma de sus celdas "**vecinas**" así como de las vecinas de éstas, que tengan un valor par. Para realizar la suma, este método debe considerar cada celda **solamente una vez**. Note que si el método no es recursivo, no tendrá puntos en esta respuesta.

d) (5%) Programe una clase principal que construya la matriz, la despliegue y encuentre la "mancha par" de todas sus casillas.

Ejemplo, si tenemos la siguiente matriz:

5	4	8	2	1
2	6	5	6	3
3	8	4	7	8
1	2	7	1	4

sumarManchaPar(0, 0) = 0
sumarManchaPar(1, 1) = 42

2) (50%) **Nombres en el mundo**

a) (15%) Defina la clase **ListaConContador** e implemente el método **agregue(String nombre)** que crea una lista enlazada y ordenada de nombres de personas sin repetir. Para cada nombre debe contabilizar la frecuencia observada del valor **nombre**, que se le da como parámetro.

b) (15%) Cree la clase **ArbolPaises** que en cada nodo posea el nombre de un país (sin repetir) y una lista de ocurrencias de nombres de personas (del tipo ListaConContador). Implemente el método **agregue(String nombrePersona, String nombrePais)**, que registre para el país recibido el nombre de la persona.

c) (5%) Cree el método **toString()** en la clase **ArbolPaises**, que muestre por país la lista de nombres con su frecuencia.

d) (10%) Cree el método **cuenteFrecuencia(String nombrePersona)** en la clase **ArbolPaises**, que halle la cantidad de veces que se repite el nombre en todo el árbol. Por ejemplo:

```
int frecuencia = arbol.cuenteFrecuencia(“Ana”);   // debe asignar un 5
```

e) (0%) En el controlador del punto b) agregue los siguientes datos:

```
arbol.agregue("Maria","Guatemala");
arbol.agregue("Ana","Peru");
arbol.agregue("Ana","Chile");
arbol.agregue("Ana","Chile");
arbol.agregue("Pedro","CR");
arbol.agregue("Juan","Guatemala");
arbol.agregue("Maria","CR");
arbol.agregue("Ana","Colombia");
arbol.agregue("Pedro","Nicaragua");
arbol.agregue("Juan","Peru");
arbol.agregue("Ana","Peru");
arbol.agregue(“Chepe”, “CR”);
arbol.agregue(“Luisa”, “USA”);
```

f) (5%) En el mismo controlador agregue las instrucciones necesarias para mostrar toda la información del árbol y la frecuencia de algunos nombre.