Programación I

Recuperatorio (turno tarde) – 27/11/2019

Ejercicio 1 *(20 pts. - 5 pts. cada item.)*

Discutir la veracidad de las siguientes afirmaciones, justificando claramente su respuesta.

a) El método que agrega un elemento al final de una lista enlazada tiene complejidad O(n) y en una lista doblemente enlazada tiene complejidad O(1).
b) Se produce aliasing cuando dos variables referencian a objetos que tienen los mismos valores en todas sus variables de instancia.
c) En un arreglo ordenado, el método de búsqueda binaria siempre termina antes que el método que busca un elemento recorriendo el arreglo desde el principio.
d) Un TAD puede tener más de una implementación.

Ejercicio 2 *(20 pts.)*

Escribir el método recursivo **public static** String repetirChar(String s, **char** c, **int** n), que devuelve la cadena con los mismos caracteres de s, donde el caracter c figura repetido n veces.

Por ejemplo:

- repetirChar("ameno",'a',3) debe devolver "aaameno".
- repetirChar("caerse",'e',4) debe devolver "caeeeerseeee".
- repetirChar("fino",'u',2) debe devolver "fino".

Se pide resolver **utilizando recursión**. Se pueden dar por hecha la función String resto(String s) que devuelve una cadena igual a s pero sin su primer caracter.

Ejercicio 3 *(35 pts. - item a. 15 pts. - item b. 20 pts.)*

Consideremos las clases Distribuidora, Producto, Proveedor y Precio definidas como:

```
public class Distribuidora {
    public Producto[] productos;
    ...
}
public class Producto {
    public String marca;
    public Proveedor[] proveedores;
    public Precio[] precios;
    public int stock;
    ...
}
```

```
public class Proveedor {
    public String direccion;
    public String[] barriosDeZonaDistribucion;
    ...
}

public class Precio {
    public double costoUnitario;
    public int descuento; //de 0 a 100
    ...
}
```

Estas clases modelan una distribuidora de bebidas sin alcohol. La distribuidora cuenta con un arreglo de los productos que comercializa. Cada Producto tiene una marca, el stock, un arreglo de proveedores y un arreglo de precios del mismo largo que el arreglo de proveedores que corresponde al precio de venta del producto según cada proveedor.

Para la clase Distribuidora, se pide:

a) Escribir un método **public** Set<Producto> preciosCuidados(**double** p) que devuelva el conjunto de productos que tengan al menos un precio con costoUnitario menor a p.
b) Escribir un método **public int** ofertasEnBarrio(**int** desc, String barrio) que devuelva la cantidad de productos que tienen todos sus precios con descuento mayor a desc y que todos sus proveedores tienen a barrio entre los barrios de su zona de distribucion.
c) (*bonus track* 20 pts.) Escribir un método **public** String barrioMasPopular() que devuelve el barrio que aparece la mayor cantidad de veces en los proveedores de los productos la distribuidora. En caso de existir más de un barrio con esta característica puede devolver cualquiera de ellos.

Ejercicio 4 *(25 pts.)*

Dadas las clases NodoInt y ListaInt cuyas variables de instancia son las siguientes:

```
public class NodoInt {
    int elemento;
    NodoInt siguiente;
}
```

```
public class ListaInt {
    NodoInt primero;
    ...
}
```

Se pide para esta clase escribir el método de instancia **public void** duplicar(**int** k) que modifique la lista agregando nodos de manera que los elementos iguales a k aparecen duplicados. Por ejemplo:

- Si la lista es [3,1,2,4,2] y k=2 la lista debe quedar [3,1,2,2,4,2,2].
- Si la lista es [3,1,2,4,2] y k=3 la lista debe quedar [3,3,1,2,4,2].
- Si la lista es [9,8,7,6] y k=4 la lista debe quedar [9,8,7,6].
- Si la lista es [] y k=3 la lista debe quedar [].

Se pide además que el método implementado **sea de orden lineal**, es decir, $O(n)$ donde n es la cantidad de elementos de la lista. Justificar la complejidad del mismo.