Programación I

Parcial (turno noche) – 14/11/2019

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

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

a) Se puede mejorar la complejidad del método largo en listas enlazadas si se agrega como variable de instancia la cantidad de nodos.
b) Un tipo abstracto de datos (TAD) tiene siempre una única implementación.
c) Para invocar un método de clase es necesario haber creado previamente una instancia de esa clase.
d) Los algoritmos de ordenamiento por Burbujeo y Selección en peor caso tienen el mismo orden de complejidad $O(n)$.

Ejercicio 2 *(20 pts.)*

Escribir una función recursiva String eliminarVocalesYRevertir(String s) que toma un String s y devuelve una nueva cadena que resulta de eliminar de s todas las vocales y luego invertir sus caracteres. Por ejemplo:

- eliminarVocalesYRevertir(''toro'') debe devolver ''rt''.
- eliminarVocalesYRevertir(''risa.'') debe devolver ''sr.''.
- eliminarVocalesYRevertir(''mapa'') debe devolver ''pm''.
- eliminarVocalesYRevertir(''fadap'') debe devolver ''pdf''.
- eliminarVocalesYRevertir(''uia'') debe devolver ''''.

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 carácter.

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

Consideremos las clases Modulo7, Laboratorio, PC, Marca y Componente definidas como:

```
public class Modulo7
{
    Laboratorio[] labos;
}

public class Laboratorio
{
    int numero;
    int capacidad;
    PC[] computadoras;

}

public class PC
{
    String serial;
    String modelo;
    String OS;
    Componente[] componentes;
}
```

```
public class Componente
{
    String nombre;
    String tipo;
    Marca marca;
}


public class Marca
{
    String nombre;
    float calidad;
}
```

Estas clases modelan los laboratorios que posee el Modulo7. Cada laboratorio posee computadoras que están compuestas por componentes. Cada Componente es una determinada marca que tiene un estandar de calidad, por lo cual según la marca y especificaciones se le asigna un valor entre 1 y 5 a la calidad del componente. Para la clase Modulo7, se pide:

a) Escribir un método LinkedList<Componente> componentesPorMarca(Marca m) que devuelva una lista con todos los componentes del módulo 7 que coincidan con la marca recibida por parámetro. La lista no debe contener elementos repetidos.
b) Escribir un método **boolean** gamaAlta() que determina si existe una pc en el módulo 7 que sea de alta gama. Se considera que una pc es de alta gama si todos sus componentes tienen calidad mayor o igual a 4.
c) (*bonus track* 20 pts.) Escribir un método Marca marcaMasUsada() que devuelva la marca que más componentes provee al módulo 7. En caso de tener más de una Marca con estas condiciones, se debe devolver cualquiera de ellas.

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;
    ...
}
```

Para la clase ListaInt se pide escribir el método de instancia **void** descomponerPares() que por cada elemento par de la lista, este sea reemplazado por la mitad y se agregue un nodo nuevo que contenga como valor el doble del mismo. Por ejemplo:

- Si la lista es [1,12,3] debe quedar como [1,6,24,3]
- Si la lista es [2,10,12] debe quedar como [1,4,5,20,6,24]
- Si la lista es [8] debe quedar como [2,16]
- Si la lista es [ ] debe quedar como [ ]
- Si la lista es [1,3,7,11] debe quedar como [1,3,7,11]

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.