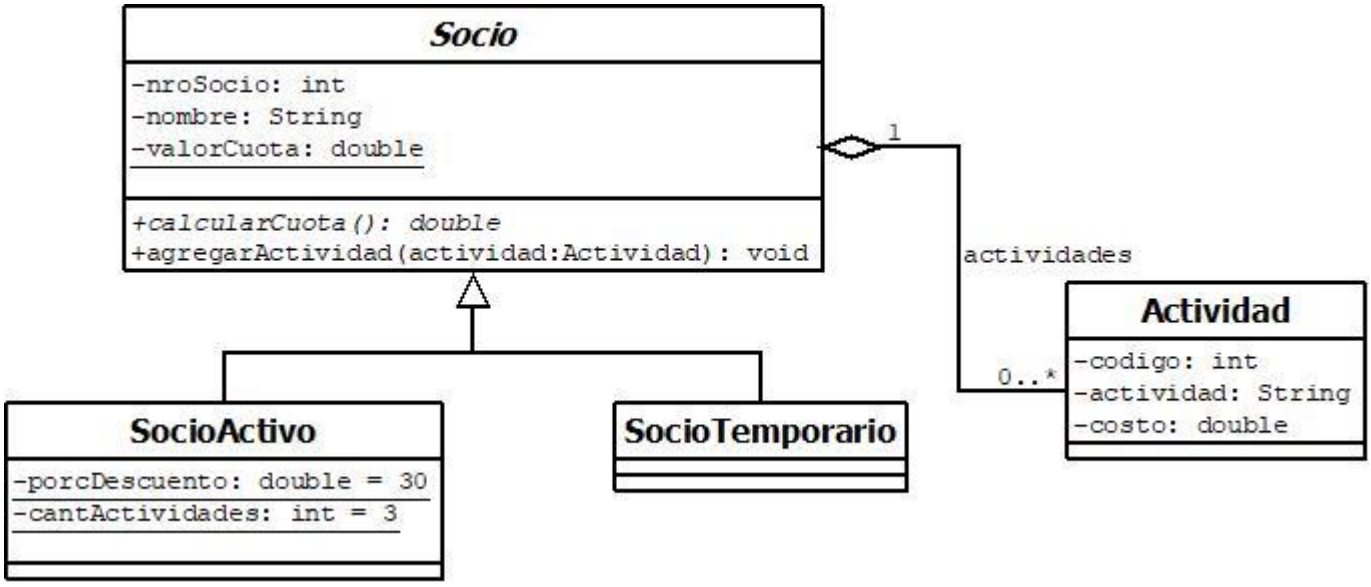
1° PARCIAL DE ALGORITMICA Y PROGRAMACION II - 2022

1) Desarrollar un sistema que administra el cobro de las cuotas sociales de un Club. El Club tiene dos tipos de socios: activos y temporarios. El valor de la cuota de socio es la misma para cualquier tipo de socio. Cada actividad que realiza (futbol, básquet, pileta, etc.) tiene un costo adicional. El importe a cobrar a cada socio es la suma del valor de la cuota social más el costo de las actividades que realiza. Los socios activos que realizan tres o más actividades tienen un descuento de un 30% del total a abonar.

Diagrama de Clases



Crear todas las clases con sus atributos y constructores. Implementar sus métodos.

2) Realizar un programa de prueba donde se crea un arreglo de socios activos y temporarios que realizan distintas actividades.

a) Recorrer el arreglo mostrando el nombre y la cuota que debe abonar cada socio.
b) Mostrar la sumatoria de todas las cuotas que debe abonar los socios del Club.

3) Dada una lista simplemente enlazada, agregar el siguiente método y realizar un programa que pruebe los diferentes casos que se pueden presentar.

```
/**
 * Retorna una nueva lista incluyendo los elementos alternados
 *
 * @param odd si es true incluye los elementos que están en la ubicación impar
 *            si es false incluye los elementos que están en la ubicación par
 *
 * @return retorna una nueva lista con los elementos alternados
 *
 * Por ejemplo dada la siguiente lista: {A, B, C, D}
 *
 * alternate(true) => {A, C}
 * alternate(false) => {B, D}
 *
 * Nota: este método no modifica la lista original
 *
 */
public SinglyLinkedList<E> alternate(boolean odd)
```

4) Dada una lista doblemente enlazada, agregar el siguiente método y realizar un programa que pruebe los diferentes casos que se pueden presentar.

```
/**
 * Intercambia dos elementos de la lista
 *
 * @param x: ubicación del primer elemento a intercambiar
 * @param y: ubicación del segundo elemento a intercambiar
 *
 * Por ejemplo dada la siguiente lista: {A, B, C, D}
 *
 * swap(0,3) => {D, B, C, A}
 * swap(2,1) => {A, C, B, D}
 * swap(2,2) => {A, B, C, D} (no realiza ningún cambio)
 *
 * Si los índices están fuera de rango lanza la excepción
 * IndexOutOfBoundsException
 *
 * Nota: este método intercambia los elementos de una lista dada. No crea una
 * lista nueva
 *
 */
public void swap(int x, int y)
```

TEORIA

1. Explique los conceptos de abstracción, encapsulamiento e interfaz. Indique qué provee Java para cada uno de ellos.
2. ¿Qué tipos de recursividad conoce? Indique que implica cada uno de ellos si se analiza su complejidad.
3. ¿Qué es recursividad de cola y qué puede decir respecto de eliminar la misma?
4. Realice un análisis comparativo de las estructuras de datos fundamentales: Lista enlazada simple, doblemente enlazada y circular.

Nota: *se tendrá en cuenta la redacción correcta de las respuestas, es decir, que se entiendan los conceptos y que sintácticamente estén bien construidas.*

IMPORTANTE:

1. Para la parte práctica del parcial, subir solamente los archivos con extensión .java.

2. Para la parte teórica del parcial, subir un solo archivo de Word. (con extensión doc o .docx)