ADA	Febrero	Curso 2012/2013

En una gran empresa se vota entre todos los empleados tres propuestas (1,2,3) para solucionar un determinado problema laboral, Se escogerá una propuesta si la votación supera el 50% de los empleados que votan (las abstenciones son números naturales mayores de 3).

Sabiendo que los votos se recogen a medida que se producen las votaciones en una lista de enteros, se propone una vez haya finalizado la votación ***diseñar un algoritmo por la técnica de divide y vencerás*** que determine finalmente si hay propuesta que cumpla las condiciones iniciales o hay que volver a negociar (se devuelve un 0).

Por ejemplo, dada la siguiente lista de votaciones como entrada, el algoritmo devolvería "2"

5,1,2,2,1,3,2,1,2,2,2,3,2

Puede utilizar los siguientes métodos:

Integer elegirPivote(List<Integer> votos, Integer i, Integer j)

Este método elige un valor como pivote para una lista de votos entre los índices "i" y "j".

List<Integer> bh(List<Integer> votos, Integer p, Integer i, Integer j)

Este método implementa el algoritmo de la bandera holandesa para la lista de votos pasada como parámetro para un pivote "p" entre los índices "i" y "j". Dicho método devuelve una lista de enteros de tamaño dos que contiene los índices que indican las fronteras de los elementos menores, iguales y mayores que el pivote. Por ejemplo, para el caso anterior, si hacemos:

```
List<Integer> solBH = new ArrayList<Integer>();
p = elegirPivote(voto, 0 , numeros.size())  // si por ejemplo p fuera 2
solBH = bh(números,p,0,números.size());
```

Podríamos obtener usando los métodos anteriores una lista que divide el problema en 3 partes, una parte con los menores que el pivote, otra para los que son iguales que el pivote y otra última con los mayores.

*1,1,1,**2,2,2,2,2,2,2**,3,3,5*

Se PIDE:

1. Rellenar la siguiente ficha
2. Diseñar en el lenguaje Java utilizando una función que utilice la técnica de ***divide y vencerás la función que permite resolver el problema.*** El prototipo de dicha función es:

Integer BuscaPropuestaMayoritaria(List<Integer> votos)

Solución:

1. Rellenar la siguiente ficha

Ficha Examen	
Búsqueda del voto mayoritario	
Técnica: *Divide y Vencerás sin Memoria*	
Tamaño: *J-I*	
Propiedades Compartidas:	*d, List<Integer>* *n, d.size()*
Propiedades Individuales:	*i, entero en [0,d.size())* *j, entero en [i+1,d.size()]*
Solución: *Entero*	
Inicialización *votoMayoritario(d)= (vMayoR(d,0,d.size()) < 4) ? vMayoR(d, 0,d.size()) : 0*	
Generalización $vMayoR(d,i,j) = \begin{cases} 0, & j-i \leq \frac{n}{2} \\ d1[a], & j-i > \frac{n}{2}, b-a > \frac{n}{2} \\ vMayoR(d1,i,a), & j-i > \frac{n}{2}, b-a \leq \frac{n}{2}, a-i > \frac{n}{2} \\ vMayoR(d1,b,j), & j-i > \frac{n}{2}, b-a \leq \frac{n}{2}, j-b > \frac{n}{2} \\ 0, & en\ otro\ caso \end{cases}$ $p = ep(d,i,j)$ $(d1,a,b) = bh(d,i,j,p)$	
Recurrencia	$T(n) = n + \frac{1}{n}\sum_{s=0}^{n-1} T(s)$
Complejidad	$\Theta(n)$

3. Diseñar en el lenguaje Java utilizando una función que utilice la técnica de ***divide y vencerás la función que permite resolver el problema.*** El prototipo de dicha función es:

```
Integer BuscaPropuestaMayoritaria(List<Integer> votos)
{
    Integer r;
    r=vMayoR(d,0,d.size(),d.size());
   return(r < 4 ?r : 0);
}
```

```
public Integer vMayoR(List<Integer> d, Integer i, Integer j,Integer tam)
{
   Integer pivote,r,a,b;

   if(j-i<= tam/2)
         r=0;
   else {
      pivote=elegirPivote(d,i,j);
     List<Integer> solbh=bh(d,i,j,pivote);
     a=solbh.get(0);
     b=solbh.get(1);
     if(b-a>tam/2){
         r=pivote;
      }
     else {
           if(a-i > tam/2)
               r=vMayoR(d,i,a,tam);
          else if(j-b > tam/2)
               r=vMayoR(d,b,j,tam);
          else
              r=0;
       }
    }
    return(r);
}
```