

# IIP Recuperació del Segon Parcial - ETSInf

28 de Gener de 2016. Durada: 2 hores i 30 minuts.

1. 6.5 punts Tens disponible la classe `Article`, que representa un article publicat en un número d'una revista. Cada article es defineix en base a tres elements: `títol` (`String`), `autors` (array d'`String`) i `nombre de planes` (`int`). El resum de la documentació d'esta classe és el que pots veure al costat.

**Es demana:** implementar la classe `Revista`, que representa un exemplar d'una revista amb el seus corresponents articles, utilitzant els atributs i mètodes definits a continuació.

**Recorda** que les constants definides han d'utilitzar-se sempre que es requereisca.

## Constructor Summary

### Constructors

#### Constructor and Description

##### `Article()`

Constructor per defecte que crea un article estandard amb títol "Sense títol", un únic autor "Anonim" i nombre de planes igual a 1

##### `Article(java.lang.String t, java.lang.String[] a, int np)`

Constructor general que crea un article de títol `t`, autors els de l'array `a` i `np` planes

## Method Summary

### All Methods

### Instance Methods

### Concrete Methods

#### Modifier and Type

#### Method and Description

`java.lang.String[]`

##### `getAutors()`

Torna els autors de l'article

`int`

##### `getNomPlanes()`

Torna el nombre de planes

`java.lang.String`

##### `getTitol()`

Torna el títol de l'article

`java.lang.String`

##### `toString()`

Torna un `String` amb el títol entre cometes i els autors (un, dos o mes) de l'article, com mostren els següents exemples:

"Aplicacio d'algorismes en ADE" Miguel Rebollo

"IIP: errors usuals" Nati Prieto i Marisa Llorens

"Concursos de programacio a la UPV" Jon Ander Gomez et al.

- a) (0.75 punts) Atributs:

- `MAX_ARTICLES`, constant de classe (`static`) que representa el màxim nombre d'articles que pot tindre la revista, amb valor 20.
- `MAX_PLANES`, constant de classe (`static`) que representa el màxim nombre de planes que pot tindre de la revista, amb valor 200.
- `nom`, `String` amb el nom de la revista.
- `num`, enter positiu amb el número (d'exemplar) de la revista.
- `nArticles`, enter en l'interval `[0, MAX_ARTICLES]` que representa el nombre actual d'articles de la revista.
- `nPlanes`, enter en l'interval `[0, MAX_PLANES]` que representa el nombre actual de planes de la revista.
- `articles`, array d'`Article`, de capacitat `MAX_ARTICLES`, per emmagatzemar els articles de la revista, disposats seqüencialment en posicions consecutives de l'array, des de la 0 fins a la `nArticles - 1` inclusivament.

- b) (0.5 punts) Un constructor que, donats un nom (`String`) i un número (`int` positiu), crea un exemplar d'una revista amb el nom i número donats, sense continguts (sense articles i amb nombre de planes igual a 0).

- c) (1 punt) Un mètode amb perfil:

```
public Article cercar(String t)
```

que, donat un títol `t`, retorna l'`Article` de títol `t` a la revista o `null` si la revista no conté tal article.

d) (0.75 punts) Un mètode amb perfil:

```
public boolean afegir(Article a)
```

que afegeix l'article `a` a la revista si es pot, és a dir, si la revista resultant aconsegueix les condicions de màxim nombre d'articles i màxim nombre de planes. Pots suposar que l'article donat no està a la revista. El mètode tornarà `true` si s'ha afegit i `false` en cas contrari.

e) (1 punt) Un mètode amb perfil:

```
public Article mesCurt()
```

que retorna l'article amb menor nombre de planes; en cas que distints articles presenten el mateix nombre mínim de planes, ha de tornar el que apareix primer; pots suposar que hi ha com a mínim un article a la revista.

f) (1.5 punts) Un mètode amb perfil:

```
public Article[] ambUnAutor()
```

que retorna un array d'`Article` amb aquells articles amb un únic autor. La longitud de l'array ha de ser 0 si, a la revista, no hi ha articles amb un únic autor.

g) (1 punt) Un mètode amb perfil:

```
public String indexDeContinguts()
```

que retorna un `String` amb la següent estructura:

- Primera línia: nom i número de la revista (com s'indica a l'exemple posterior).
- Resta de línies: descripció de cada article (donada pel mètode `toString()` d'`Article`), seguit d'un guionet (" - ") i de la seua plana inicial (com s'indica a l'exemple posterior); la primera plana del primer article és la plana 3.

Pots suposar que hi ha com a mínim un article a la revista.

Per exemple:

```
Programacio a la UPV - Numero 5
"IIP: errors usuals" Nati Prieto i Marisa Llorens - 3
"Concursos de programacio a la UPV" Jon Ander Gomez et al. - 15
"Una experiencia amb tablets per a programacio" Mabel Galiano - 22
"Formes diferents d'ensenyar-se if i if-else" Assump Casanova i Paco Marques - 34
"Retrocomputing: descripcio de llenguatges antics" Jorge Gonzalez i Quique Ramos - 51
"Qui es l'ultim? Fundaments d'estructures FIFO" Carlos Herrero et al. - 59
"Aplicacio d'algorismes en ADE" Miguel Rebollo - 78
```

## Solució:

```
/** Classe Revista: representa una revista amb diferents articles
 *
 * @author Examen IIP
 * @version Recuperacio segon parcial - Curs 2015-2016
 */
public class Revista {
    /** Nombre maxim d'articles que pot tenir una revista */
    public static final int MAX_ARTICLES = 20;
    /** Nombre maxim de planes que pot tenir una revista */
    public static final int MAX_PLANES = 200;

    private String nom;           // Nom de la revista
    private int num;              // Numero de l'exemplar
    private Article[] articles;   // Conjunt d'articles
    private int nArticles;        // Nombre d'articles
    private int nPlanes;          // Nombre de planes
```

```

/** Constructor que crea un nou exemplar amb numero {@code nu} de la
 * revista de nom {@code nr}, sense articles (0 articles, 0 planes)
 * @param nr String, nom de la revista
 * @param nu int, numero de l'exemplar (nu >= 0)
 */
public Revista(String nr, int nu) {
    nom = nr; num = nu;
    articles = new Article[MAX_ARTICLES];
    nArticles = 0; nPlanes = 0;
}

/** Torna l'article amb titol {@code t} de la revista, o {@code null}
 * si la revista no conte tal article
 * @param t String amb el titol
 * @return Article, article amb titol {@code t} o {@code null} si no esta
 */
public Article cercar(String t) {
    int i = 0;
    while (i < nArticles && !articles[i].getTitol().equals(t)) { i++; }
    if (i < nArticles) { return articles[i]; }
    return null;
}

/** Afegeix l'article {@code a} a la revista si es pot (la revista resultant
 * no te mes de MAX_ARTICLES articles ni mes de MAX_PLANES planes) <br>
 * PRECONDICIO: {@code a} no esta a la revista
 * @param a Article a afegir
 * @return boolean, {@code true} si s'ha afegit, {@code false} en altre cas
 */
public boolean afegir(Article a) {
    if (nArticles < MAX_ARTICLES
        && (nPlanes + a.getNomPlanes()) <= MAX_PLANES) {
        articles[nArticles++] = a;
        nPlanes += a.getNomPlanes();
        return true;
    }
    return false;
}

/** Torna l'article amb menor nombre de planes de la revista,
 * (el primer que apareix si hi ha mes d'un) <br>
 * PRECONDICIO: com a minim un article en l'exemplar de la revista
 * @return Article, el de menor nombre de planes en l'exemplar
 */
public Article mesCurt() {
    int posMin = 0, planesMin = articles[posMin].getNomPlanes();
    for (int i = 1; i < nArticles; i++) {
        int planesI = articles[i].getNomPlanes();
        if (planesI < planesMin) {
            posMin = i; planesMin = planesI;
        }
    }
    return articles[posMin];
}

/** Torna un array amb els articles de la revista amb un unic autor,
 * (la seua llargaria pot ser 0)
 * @return Article[], array amb articles amb un unic autor
 */
public Article[] ambUnAutor() {
    int num1Autor = 0;
    for (int i = 0; i < nArticles; i++) {
        if (articles[i].getAutors().length == 1) { num1Autor++; }
    }
    Article[] aux = new Article[num1Autor];
    int j = 0;
    for (int i = 0; i < nArticles && j < num1Autor; i++) {
        if (articles[i].getAutors().length == 1) {
            aux[j++] = articles[i];
        }
    }
    return aux;
}

```

```

/** Torna l'index de continguts de l'exemplar de la revista, indicant: <br>
 * - En la primera línia, nom i número de la revista <br>
 * - En cada línia subsegüent, una descripció de cada article (donada
 * pel mètode {@code toString()} d' {@code Article}), seguida de " - " i
 * de la seua plana inicial; el primer article comença a la plana 3 <br>
 * PRECONDICIO: com a mínim un article en l'exemplar de la revista
 * @return String, index de continguts de l'exemplar de la revista
 */
public String indexDeContinguts() {
    String res = nom + " - Número " + num + "\n";
    int planes = 3;
    for (int i = 0; i < nArticles; i++) {
        res += articles[i] + " - " + planes + "\n";
        planes = planes + articles[i].getNomPlanes();
    }
    return res;
}
}

```

2. **1.5 punts** **Es demana:** escriure un mètode de classe (**static**) que, donat un **String** **s**, mostre per pantalla un dibuix amb tantes línies com caràcters té l'**String**, amb dues diagonals que s'uneixen en la primera línia i divergeixen cap a l'última línia (això és, com una 'V' invertida), de manera que, en la línia *i*-èsima es mostre, en cada diagonal, el caràcter *i*-èsim de l'**String** donat. Pots suposar que la llargària de l'**String** **s** és major que 1.

P.e., per a **s** = "Java":

		J	J		
	a		a		
v				v	
a					a

**NOTA:** l'exemple es mostra dins d'una quadrícula per tal que pugues distingir millor els blancs del dibuix.

### Solució:

```

/** Mostra per pantalla un dibuix amb n línies formades
 * per dues diagonals amb les lletres de l'String s
 * (de llargària major que 1) formant una 'V' invertida
 */
public static void dibuixar(String s) {
    for (int i = 0; i < s.length(); i++) {
        for (int j = i + 1; j < s.length(); j++) {
            System.out.print(" ");
        }
        System.out.print(s.charAt(i));
        for (int j = 0; j < i * 2; j++) {
            System.out.print(" ");
        }
        System.out.println(s.charAt(i));
    }
}

```

3. **2 punts** **Es demana:** escriure un mètode de classe (**static**) que, donat un enter no negatiu, retorne un array de **boolean**, de longitud igual al nombre de dígits de l'enter donat, tal que l'element de la posició *i* de l'array és **true** quan el dígit de la posició *i* de l'enter (numerant d'esquerra a dreta) és múltiple de 3, i **false** en un altre cas.

Per exemple:

Per a 357 ha de tornar	0	1	2	Per a 1609 ha de tornar	0	1	2	3
	true	false	false		false	true	true	true

## Solució:

```
/** Retorna un array de boolean indicant si cada
 * digit de n (n>=0) es multiple de 3 o no
 */
public static boolean[] intABool(int n) {
    int s = 1, copia = n;
    while (copia > 9) {
        s++;
        copia = copia / 10;
    }

    boolean[] a = new boolean[s];
    while (s > 0) {
        s--;
        a[s] = ((n % 10) % 3) == 0;
        n = n / 10;
    }
    return a;
}
```