

## Funcionalidad

- Las clases Tweet y TweetBD funcionan correctamente.
- En la clase TweetBD se realizan copias de objetos siempre que sea necesario para que ningún atributo se pueda modificar desde fuera de la clase.

```
// Busca y devuelve el tweet más reciente de la BBDD
// Si no hay ningún tweet devolverá null
public Tweet tweetMasReciente() {
    Date masReciente = new Date();
    masReciente = tweets.get(0).getFecha();
    Tweet masRecienteTweet = null;
    for(int i = 1; i < getNumeroTweets() - 1; i++) {
        if(masReciente.before(tweets.get(i).getFecha())) {
            masReciente = tweets.get(i).getFecha();
            masRecienteTweet = new Tweet(tweets.get(i));
        }
    }
    return masRecienteTweet;
}
```

- Todas las opciones del menú se ejecutan correctamente y se comprueba el formato de la fecha. También se comprueba el usuario aunque no sería necesario .

```
Matcher comparaFormato;
String usuario = "";
//Si el usuario no es valido la pedirá otra vez
do {
    System.out.println(" Usuario: ");
    usuario = teclado.next();
    comparaFormato = formatoUsuario.matcher(usuario);
    if(!comparaFormato.matches()) {
        System.out.println("El usuario es inválido, tipo esperado
8 carácteres.");
    }
}while (!comparaFormato.matches());
usuario = "USER_" + usuario;

//Comparamos la fecha y la hora para que sea valida (\\d == a
[0-9])
Pattern formatoFecha = Pattern.compile("\\d{4}-\\d{2}-\\d{2}");
Pattern formatoHora = Pattern.compile("\\d{2}:\\d{2}:\\d{2}");
String fechaStr;
String hora;
//Si la fecha no es valida la pedirá otra vez
```

```

do {
    System.out.println("Introduce una fecha (aaaa-mm-dd): ");
    fechaStr = teclado.next();
    comparaFormato = formatoFecha.matcher(fechaStr);
    if(!comparaFormato.matches()) {
        System.out.println("La fecha es inválida");
    }
} while (!comparaFormato.matches());
//Si la hora no es valida la pedirá otra vez
do {
    System.out.println("Introduce una hora (hh:mm:ss): ");
    hora = teclado.next();
    comparaFormato = formatoHora.matcher(hora);
    if(!comparaFormato.matches()) {
        System.out.println("La hora es inválida");
    }
} while (!comparaFormato.matches());

```

- El programa está bien estructurado y no hay redundancias.
- Se utilizan 4 métodos con paso de parámetros y al menos uno de ellos usa la sentencia return. Además a uno de ellos se le pasará un objeto de una de las clases que hemos creado u otro lo devolverá.
- En el método tweets anteriores no me imprime la descripción, y he tenido que imprimirla con `System.out.print(tweetBusqueda.descripcion);`, en un ejercicio de concesionario me suena que se imprimía sin hacer el print, o quizás este equivocado.

```

1. public int borrarTweets(String usuario)
2. public Tweet getTweet(int i)
3. public TweetBD buscarTweets(String palabra)
4. public TweetBD tweetsAnteriores(Tweet esteTweet) {
    TweetBD tweetBDAnterior = new TweetBD("TWEETS ESCRITOS ANTES DE:
    " + esteTweet + "\n");
    .....
    return tweetBDAnterior;
}

```

- Durante todo el programa, los datos del teclado se leen mediante nextLine solo al pedir el tweet, en los otros casos no se ve necesario

```

//Pedimos el tweet
System.out.println(" Mensaje: ");
teclado.nextLine();
String mensaje = teclado.nextLine();

```

y se utiliza los métodos parseXxx para convertir algún dato a otro tipo.

```

fecha = formatoFecha.parse(fechaHora);

```

- Se utiliza un iterador

```
public int borrarTweets(String usuario) {
    Iterator<Tweet> it = tweets.iterator();
    int tweetsBorrados = 0;
    while(it.hasNext()) {
        Tweet tweet = it.next();
        if(tweet.getUsuario().equals(usuario)) {
            it.remove();
            tweetsBorrados++;
        }
    }

    return tweetsBorrados;
}
```

y un bucle for-each para recorrer el ArrayList.

```
public TweetBD tweetsAnteriores(Tweet esteTweet) {
    TweetBD tweetBDAnterior = new TweetBD("TWEETS ESCRITOS ANTES DE: "
        + esteTweet + "\n");
    //TweetBDAnterior.addTweet(getTweet(numTweet));
    for(Tweet miraTweet: tweets){
        if(miraTweet.esMasReciente(esteTweet)) {
            Tweet copiaTweet = new Tweet(miraTweet);
            tweetBDAnterior.addTweet(copiaTweet);
        }
    }
    System.out.print(tweetBDAnterior.descripcion);

    return tweetBDAnterior;
}
```

- Se utiliza la palabra reservada this, tanto para referenciar un atributo como un constructor.

```
public TweetBD(String descripcion) {
    this.descripcion = descripcion;
    tweets = new ArrayList<Tweet>();
}

public Tweet(String usuario, String fechaHora, String tweet) {
    this.usuario = usuario;
    //2010-03-03T14:19:51
    try {
        SimpleDateFormat formatoFecha = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss");
        fecha = formatoFecha.parse(fechaHora);
    } catch (ParseException e){
        System.out.println("El formato de la fecha es incorrecto, fecha esperada (yyyy-MM-dd HH:mm:ss): " + e.getMessage());
    }
    this.tweet = tweet;
}
```

- Se ha decido imprimir después de cada acción en un archivo nuevo, pero habría que darle un repaso y corregir en alguna opción.

Nota: 6

## Comentarios

- Se detecta código que se beneficiaría de un comentario o algunos comentarios son excesivos o inapropiados

Nota: 1

## Legibilidad

- El código está bien organizado, sigue la guía de estilo y es fácil de leer.

Nota: 1.5

Nota de la autoevaluación 8.5