CONVENCIONES DE ESTILO

LEE ESTE DOCUMENTO RÁPIDAMENTE NO ES PARA ESTUDIAR. ES PARA SIMPLEMENTE QUE SEPAS QUE SON LAS CODE CONVENTIONS

El "estilo" se refiere a un conjunto de convenciones (no normas ya que no es obligatorio cumplirlas) para dar formato a los programas. Este formato "acordado" permite que los programas sean más fáciles de leer y por tanto de mantener. Las normas de "estilo" no son de obligatorio cumplimiento pero si observamos el código escrito por los mejores programadores java vemos que parece que siguen unas normas de estilo tácitas.

El documento para consultar cuestiones de estilo "oficial" es
http://www.oracle.com/technetwork/articles/javase/codeconvtoc-136057.html
de la web oficial de Oracle pero ten en cuenta que hay empresas que por diversas razones utilizan variantes propias que contradicen incluso a este documento. La mejor forma, a mi entender, de captar el "estilo" correcto es fijarse en los ejemplos "de confianza": libros de alta calidad como *piensa en java*, tutoriales web de Oracle java, etc…

el enlace anterior se puede romper en cualquier momento, para llegar a él haz búsqueda en google tipo "oracle java code conventions"

Por el momento me gustaría que te fijaras en las siguientes cuestiones:

- Sangrar adecuadamente cada nuevo bloque de sentencias.

Los sangrados es una sencilla y eficaz convención de la máxima importancia, ya que con poco esfuerzo, se gana mucha legibilidad en el código.

- La estructura recomendable a la hora de escribir una clase.

```
Comentarios de javadoc de la clase
class NombreClase {
        comentarios de funcionamiento de la clase
        variables estáticas
        variables de instancia públicas (debe evitarse su uso)
        variables de instancia protegidas
        variables de instancia privadas
        métodos constructores
        resto de métodos
}
```

Observa que para métodos no se especifican si se colocan antes los private o los public ya que se pueden mezclar. Para pensar en un orden u otro el criterio es la legibilidad del código.

- Sobre identificadores
  - Se deben elegir un nombres identificadores significativos y a ser posible breves. En cualquier caso se prefiere la claridad a la brevedad. Por ejemplo, es preferible el identificador,
    integralIndefinida
    que el de,
    intInd

  - Los identificadores de clases, se escriben en minúsculas pero deben empezar con mayúscula. Si el identificador consta de varias palabras colócalas juntas,

con la inicial de cada una en mayúsculas. Por ejemplo, serían identificadores apropiados,

```
class Cliente
class ClientePreferencial
```

- Los identificadores de variables, sea locales, de instancia, estáticas… deben escribirse en minúsculas. Si el identificador consta de varias palabras se colocan separadas por el carácter de subrayado o bien todas seguidas(mejor todas seguidas). En este último caso, la primera de ellas se escribe en minúsculas y la inicial de las demás en mayúsculas.

  Por ejemplo,

```
double valorFinal=0.0;
int cantidadFinal=0;
String nombre_cliente="Aurora"; //mejor nombreCliente
```

- Para las constantes, el identificador debe usarse todo mayúsculas. Si el identificador consta de varias palabras se separan con el carácter de subrayado. Ejemplos correctos serían,

```
final int MINIMO=100;
final double PRECISION=0.001;
final double ALTURA_MEDIA=29.5;
```

- En el caso de los métodos, el identificador debe ser preferentemente un verbo y debe usarse en minúsculas. Si el identificador contiene varias palabras, la inicial de todas las posteriores a la primera va en mayúsculas. Ejemplos válidos serían,

```
public void introducir(double valor){
- - - cuerpo del método - - -
}
public double obtenerMedia(){
- - - cuerpo del método - - -
}
```

- Colocar la apertura de bloque ({) al final de la línea inicial de la sentencia. El (}) colócarlo en línea aparte y alineado con el principio de la sentencia. Si la sentencia de control es un if-else la cláusula else debe comenzar en la línea siguiente al fin del bloque de la cláusula if. Por ejemplo,

```
if (i==j) {
  dato=5;
} else {
  dato=6;
}
en lugar de,
if (i==j)
{
  dato=5;
} else
{
  dato=6;
}
```

- Poner cada nuevo bloque de sentencias entre llaves aunque conste de una sola sentencia. (De esto ya hablamos mucho …)

U3_B9_E1: Aplica las normas de estilo para mejorar la legibilidad del código.

```
class coche{

int pasajeros;
int kpl;
int Autonomia(){
  return  deposito*kpl;
}
int deposito;
  double gasofaNecesaria(int kilometros)
  {
    return (double) kilometros/kpl;
  }
}
class Unidad3 {


  public static void main(String[] args) {
coche peugeot308 = new coche();
peugeot308.pasajeros=5;
peugeot308.deposito=60;
peugeot308.kpl=20;
System.out.println("pasajeros:" + peugeot308.pasajeros);
System.out.println("deposito:" + peugeot308.deposito);
System.out.println("kpl:" + peugeot308.kpl);
System.out.println("La autonomía es: " + peugeot308.Autonomia());
System.out.println("Para recorrer 100 kilometros se necesitan " + peugeot308.gasofaNecesaria(100) + " litros");
}
}
```

U3_B9_E2: Mejora la legibilidad del siguiente código aplicando normas de estilo

```
class Unidad3 {
public static void main(String[] args) {
for(int i=2;i<5;i++)
{
   System.out.println("tabla de multiplicar del  "+i);
   for(int j=0;j<10;j++)
       System.out.println("\t"+ i+"x"+j+"="+i*j);
}
}
}
```

Y por cierto, la aplicación de estas convenciones "básicas" como sangrados, colocaciones de llaves y nombres de identificadores se tendrán en cuenta en la corrección de exámenes.