

Métodos

Escribe un programa que determine la cantidad de combinaciones que se pueden realizar tomando k elementos de un grupo de n elementos, es decir,

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

1 La solución larga

Una solución preliminar bastante larga (y eso que usamos for en vez de while) es

```
1  int n=con.readInt();
2  int k=con.readInt();
3  int nfactorial=1;
4  for (int i=1; i<=n; i++)
5      nfactorial=nfactorial*i;
6  int kfactorial=1;
7  for (int i=1; i<=k; i++)
8      kfactorial=kfactorial*i;
9  int nkfactorial=1;
10 for (int i=1; i<=n-k; i++)
11     nkfactorial=nkfactorial*i;
12 int combs=nfactorial/(kfactorial*nkfactorial);
13 con.println ("El número de combinaciones es "+combs);
```

Si te fijas, las líneas 3-5, 6-8 y 9-11 hacen lo mismo: calcular el factorial de un número (n, k y n-k respectivamente). Java permite definir métodos que llevan a cabo tareas específicas. A modo de ejemplo, si suponemos que existe un método de nombre factorial que recibe un número entero n y devuelve n!, entonces el programa queda

```
14 int n=con.readInt();
15 int k=con.readInt();
16 int combs=factorial(n)/(factorial(k)*factorial(n-k));
17 con.println ("El número de combinaciones es "+combs);
```

El programa no sólo ha quedado mucho más pequeño, sino que además es más fácil de entender y mantener.

2 El método factorial

El problema ahora es escribir el método factorial:

```
1  int factorial (int x) {
2      int producto=1;
3      for (int i=1; i<=x; i++)
4          producto=producto*i;
5      return producto;
6  }
```

2.1 Descripción

1 El encabezado del método especifica qué devuelve, su nombre y qué es lo que recibe. En este caso, se llama factorial, recibe un entero x y su resultado es un entero.

2-4 Se calcula el factorial con un ciclo for.

5 El resultado del método es el valor de la variable producto

3 Programa completo

```
7  public class Ejemplo {
8      Console con=new Console();
9      public static void main(String args[]) {
10         new Ejemplo();
11     }
12     int factorial (int x) {
13         ...
14     }
15     public Ejemplo () {
16         ...
17         con.println ("Número de combinaciones:"+comb);
18     }
19 }
```

4 Forma general de un método

Un método consta de dos partes principales: su encabezado y su cuerpo. El encabezado especifica en forma precisa su nombre, qué parámetros recibe y qué devuelve¹. El cuerpo de la función lleva a cabo el "trabajo sucio" y tiene como restricción que tiene que devolver un valor cuyo tipo sea el especificado en el encabezamiento. Suele terminar con la instrucción return, que devuelve el valor de la expresión que le acompaña a quien invocó al método, terminando de paso su ejecución.

¹Los métodos que no devuelven nada, se declaran como **void**

5 Invocación de un método

La forma general de invocación a un método es nombre(parámetros). Cuando devuelve un valor, hay que asignarlo a una variable o usarlo dentro de una expresión. ¿Qué hace Java al ver una invocación? Consideremos el ejemplo

```
int x=factorial(n-k);
```

1. Se evalúan las expresiones que se pasarán como parámetro. Si $n=5$ y $k=3$, entonces se calcula $n-k=2$
2. Se copia el valor de la expresión (es este caso 2) en el parámetro correspondiente de la función
3. Se ejecutan las instrucciones de la función
4. Se recibe el resultado de la función

6 Paso de parámetros

El paso de los parámetros en Java es por valor, es decir, los valores que aparecen en la invocación se copian en las variables que aparecen en la declaración. Esto quiere decir que si la variable del método se modifica, la variable en la invocación permanece inalterada ¿Qué hace el siguiente programa cuando `z` es invocado?

```
1 void incrementa(int x, int cuanto) {
2     x=x+cuanto;
3 }
4 void z () {
5     int p=5;
6     incrementa(p,3);
7     con.println (p);
8 }
```

La respuesta es que escribe 5 en la pantalla, y no 8 como podría suponerse: primero, se le asigna 5 a `p` y se llama a `incrementa` con los valores (5,3). El valor 5 se copia en una nueva variable '`x`'. El valor 3 se copia en '`cuanto`'. '`x`' se incrementa en 3 (x es 8, p es 5). Se escribe el valor de `p`, que es 5 ¿Cómo sería el programa escrito correctamente?

```
int incrementa(int x, int cuanto) {
    x=x+cuanto;
    return x;
}
```

```
void z () {
    int p=5;
    p=incrementa(p,3);
    con.println (p);
}
```

6.1 Errores comunes

Los errores más comunes están relacionados con el paso y devolución de valores:

6.1.1 No hacer nada con el valor devuelto:

```
int n=con.readInt();
factorial(n);
```

Lo correcto es

```
int n=con.readInt();
int f=factorial(n);
con.println (f);
```

6.1.2 Leer los parámetros dentro del método

Este es sin duda el error más común y el conceptualmente más grave. Consiste en leer del teclado los valores de los parámetros, siendo que estos se reciben del programa principal:

```
int suma(int a, int b) {
    a=con.readInt();
    b=con.readInt();
    return a+b;
}
```

Lo correcto es leer los datos (si es que son leídos) en una parte específica del programa, antes de invocar a los métodos.

```
int suma(int a, int b) {
    return a+b;
}
void principal () {
    int a=con.readInt();
    con.println (suma(a,5));
}
```