

## R01 Recursión: Factorial

$$n! = 1 \cdot 2 \cdot 3 \cdots \cdot n$$

long fact = 1;  $\rightarrow$  se guarda el resultado y comienza en 1  
para 1 <= i < n  $\rightarrow$  obtener el valor ya q  $n \times i = n$

esto ya simula  $1 \times 2 \times 3 \times \dots \times n$

además del caso base

if ( $n \leq 1$ ) return 1;

Por lo q si:  $n = 1 \rightarrow$  devolver 1

$n = 0 \rightarrow$  devolver 1

en el caso recursivo:

return  $n * factorialRecursivo(n-1)$

lo q se ve matemáticamente como:  $n! = n \times (n-1)!$

## R02 Recursión: Suma

Sumar b veces  $t_1$  a a con  $b > 0$  y restar 1 b veces -1 o  
0, si  $b < 0$ , si  $b = 0$  entonces la suma solo es a

int resultado = a  $\rightarrow$  guarda el valor inicial

Para  $b > 0$ :  $\rightarrow$  repite "b" veces.

for (int i = 0; i < b; i++) {

  resultado += t1

}

$\rightarrow$  cada vuelta suma +1

para  $b < 0$        $\rightarrow$  Repite 16 veces  
 for ( int i=0; i>b; i-- ) {  
 resultado --,  
 }       $\hookrightarrow$  cada vuelta resta 1

El proceso es el mismo para while y do-while pero cambiando el orden en la estructura.

R03 Recursión: Multiplicación  
 $a + a + a + \dots$  (b veces)

private int multiplicacionFor(int a, int b)  
 int resultado = 0  $\rightarrow$  la multiplicación inicia en 0 porque se irá sumando  
 for (int i = 0; i < b; i++) {  
 resultado += a       $\hookrightarrow$  repite el ciclo b veces  
 }       $\hookrightarrow$  cada vuelta suma a  
 return resultado;

El proceso es el mismo para while y do-while pero cambiando el orden en la estructura

R04 Recursión: Potencia

$$a^b = \underbrace{a \cdot a \cdot a \cdot \dots \cdot a}_{b \text{ veces}}$$

private long potenciaFor(int a, int b)

long resultado = 1;  $\rightarrow$  considera una 1

for (int i = 0; i < b; i++)  $\rightarrow$  repite el ciclo b veces

resultado \*= a

}

$\hookrightarrow$  cada vuelta multiplica el resultado mismo

return resultado; hasta las b veces que se repite el ciclo.

}

El proceso sigue lo mismo para el while

private long potenciaDoWhile(int a, int b) {

if (b == 0) return 1;  $\rightarrow$  (como do-while se ejecuta al menos una vez igual que b con cero para evitar que bucle infinito vez mas).

long resultado = 1;

do {

resultado \*= a;

i++;

$\hookrightarrow$  cada vuelta multiplica el resultado mismo

} while (i < b)

$\rightarrow$  se repite el ciclo b veces.

return resultado;

}

R05; Cálculo Progresivo

Imprimir: 1, 2, 3, ..., n

$a_k = k$  para  $k = 1, 2, 3, \dots, n$

sols. invertido 1 en cada paso.

```

private static void cicleFor (int n) {
    for (int i=1; i<=n; i++) {
        Inicio de loop  $\xrightarrow{\text{condición para seguir}}$  Inicio de ciclo
        centro System.out.print (i + " ");
        }  $\xrightarrow{\text{actualizar ciclo}}$ 
    }
}

```

(El proc. sería el mismo combinando únicamente la estructura para while y do-while considerando que do-while se ejecuta al menor una vez.

## A06. Cálculo regresivo

Iniciar  $n, n-1, n-2, \dots, 1, 0$

$$d_k = n-k \text{ para } k=0, 1, 2, \dots, n$$

repetir una vez el ciclo hasta

```

private static void cicleFor (int n) {
    for (int i=n; i>0; i--) {
        Inicio  $\xrightarrow{\text{condición}}$  decremento
        centro System.out.print (i + " ");
    }
}

```

Para el while es lo mismo salvo que declarando i a parte, la condición int i>n estaría mal y el decremento dentro del bucle, y para do-while lo mismo considerando que al menor se ejecutará una vez.