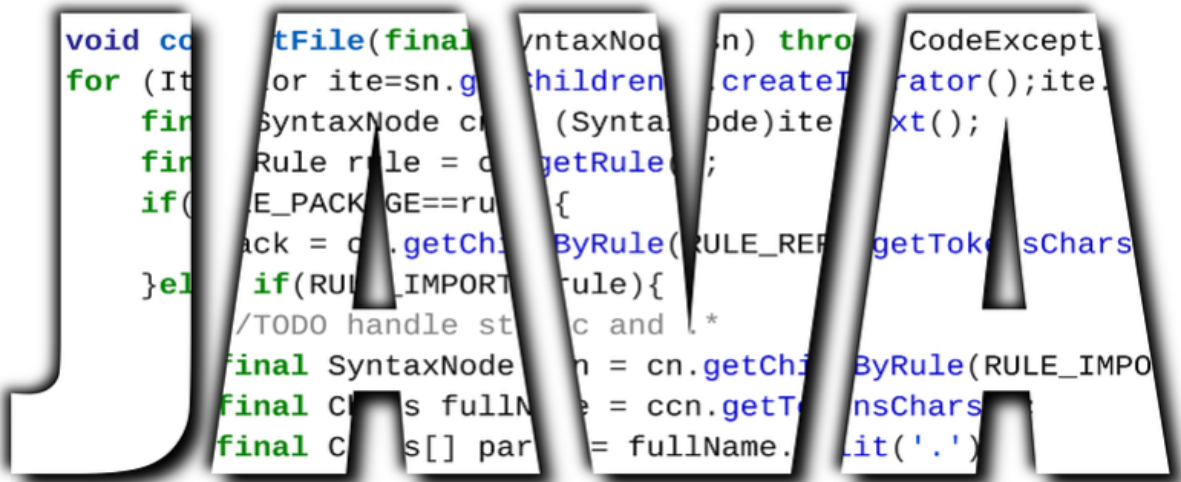


# 09 - Procediments i funcions en java



Ámbito y tipo de dato del  
valor que retornará la función

Parámetros de  
entrada

private int sumar(int numero1, int numero2)

{

Nombre de  
función

int suma = numero1 + numero2;

Instrucciones

return suma;

Valor de retorno

}

Nom: Jonathan Valle  
Curs: 1r DAW  
Any: 2022/2023

# Index

<b>09 - Procediments i funcions en java</b>	<b>1</b>
Longitud de cadena	3
Normalització	3
Tornar valor sencer equivalent	4
Múltiple de 3	4
Permutarà	4
Valor aleatori entre 2 valors	5
Número aleatori entre 1 i 6	6
Valor aleatori entre 2 i 12	6
Número de xifres que conté	6
Parell o no parell (booleana)	7
NIF	7
Ordenar Vector	8
Pedra, paper i tisora	8
100 números aleatoris entre 1 i 6	9

MÒDUL M03	EXERCICIS DEL MÒDUL M03 DELS CICLES FORMATIUS DE GRAU SUPERIOR DE ASIX, DAM I DAW	EXERCICIS
-----------	---	-----------

# MÒDUL M03 - ACTIVITAT 9

## Mètodes en java. Funcions (i procediments)

Als exercicis que figuren a continuació, es demana únicament les funcions, tot i que quan les proveu, és convenient que feu algun programa per cridar-les amb els paràmetres adequats. Fins i tot podeu llegir els paràmetres del teclat.

### 1. Longitud de cadena

1º) – Per introduir-nos a l'ús de funcions, farem una molt senzilla amb el prototipus següent: **static int len (String cad)**, al que passarem una cadena al paràmetre **cad**, i ens retornarà la seva longitud com a **int**. Fixeu-vos que ja tenim el mètode **length()** que fa exactament això, de forma que l'únic que haurem de fer des de la nostra funció és cridar-lo i retornar el valor proporcionat per aquest. Només caldrà que posem: **return cad.length();** dintre de la nostra funció. Fixeu-vos també que declarem la funció com a **static** perquè pugui ser cridada des del mètode **main**, que és **static**. Totes les funcions que farem en aquesta UF les declararem com **static**.

```
V.13 15/01/2022-14:02:26 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció comptar els caràcters d'un String
static int len(String cad){
    return cad.length();
}
```

### 2. Normalització

2º) – Funció a la que passem una cadena contenint un nom de persona i la retornarà normalitzada. La normalització consisteix en posar el primer caràcter de la cadena en majúscules, i la resta en minúscules. El prototipus de la funció haurà de ser quelcom semblant a: **static String normalitza(String nom)**. Per fer aquesta funció, no es necessari fer servir bucles, tot i que també es poden utilitzar, ja que la classe **String** té una funció que ho fa, buscar-la.

```
V.13 15/01/2022-14:02:11 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció per normalitzar la cadena
static String normalitza(String cad){
    char arr[] = cad.toCharArray();
    arr[0] = Character.toUpperCase(arr[0]);
    return new String(arr);
}
```

### 3. Tornar valor sencer equivalent

3º) – Funció **static int trunca(double x)**, a la que passarem un valor **double x** i retornarà el valor enter equivalent, on s'han truncat els decimals (s'han tret tots els decimals que pugui tenir el valor **x**). Fixeu-vos que la funció rep un **double** com a paràmetre (que tindrà decimals), i retorna un **int** (que no els té).

```
V.13 15/01/2022-14:03:09
Valle Alfaro Jonathan

// Funció retornar valor sencer
static int trunca(double x){
    return (int)x;
}
```

### 4. Multiple de 3

4º) – Funció **static long multiple3Proper(long x)** a la que passem un valor **x** de tipus **long**, i ens retorna el múltiple de tres més proper en aquest (també un valor de tipus **long**). Evidentment, si el valor **x** ja fos múltiple de 3, retornarà aquest valor (**return x**).

```
V.13 15/01/2022-14:18:02 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció multiple de 3 Proper
static long multiple3Proper(int x){
    if(x % 3 == 0){
        return x;
    }else if(x % 3 == 1){
        return multiple3Proper(x-1);
    }else {
        return multiple3Proper(x+1);
    }
}
```

### 5. Permutarà

5º) – Funció a la que passarem un vector d'enters de longitud **2** (dos elements únicament) i els permutarà, es a dir, posarà el primer element com a segon del vector i el segon element del vector com a primer. En **java**, el pas de paràmetres a les funcions és **per valor**, però si passem un vector, llavors passem el valor que té l'apuntador al vector, i mitjançant aquest, podem accedir als elements del vector per modificar-lo.

El codi d'aquest exercici seria semblant a:

```

public static void permuta(int v[]){
    int aux = v[0];
    v[0] = v[1];
    v[1] = aux;
}

public static void main(String[] args) {
    int v[] = { 10, 20};
    System.out.println("Vector abans: " + java.util.Arrays.toString(v));
    permuta(v);
    System.out.println("Vector després: " + java.util.Arrays.toString(v));
}

```

Fixeu-vos com es declara una funció que rep un vector d'enters com a paràmetre

I en executar-lo obtindríem:

```

run:
Vector abans: [10, 20]
Vector després: [20, 10]
BUILD SUCCESSFUL (total time: 0 seconds)

```

Fixeu-vos que a l'exemple s'ha declarat la funció com a **public**. Això vol dir que aquesta funció és visible i podria ser executada des de fora de la **classe** que la conté. Nosaltres, en principi, ometrem el modificador d'accés o fins i tot el podríem posar **private**.

```

V.13 15/01/2022-14:04:05
Valle Alfaro Jonathan

// Funció Permutar vector
static void permuta(int v[]){
    int aux = v[0];
    v[0] = v[1];
    v[1]=aux;
}

```

## 6. Valor aleatori entre 2 valors

6º) – Feu una funció que estarà definida com **static int aleatori (int n1, int n2)**, a la que passarem dos valors **n1** i **n2**, i ens retornarà un valor aleatori comprés entre aquests dos (**n1** i **n2** inclosos). Per això, primer de tot ens assegurarem que **n1** és el valor menor. Com ho farem? Mitjançant un condicional de l'estil **if (n1 > n2)** Dintre d'aquest condicional, permutarem **n1** i **n2**. Un cop sabem que **n1** és el menor valor, la fórmula per obtenir un valor aleatori entre **n1** i **n2** podria ser: **(int) (Math.random() \* (n2-n1+1)) + n1**. Feu un programa que demani a l'usuari dos valors enters per provar aquesta funció.

```

V.13 15/01/2022-14:04:51 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció valor aleatori entre n1 i n2
static int aleatori(int n1, int n2){
    if(n1<n2){
        n1=(int)(Math.random()*(n2-n1+1))+n1;
    }
    return n1;
}

```

## 7. Número aleatori entre 1 i 6

7º) – Feu una funció que cridarem com **dau ()**, i cada cop que l'invoquem, ens retornarà un número **aleatori enter** entre **1 i 6**.

```
V.13 15/01/2022-14:05:30 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció valor aleatori entre 1 i 6
static void dau(){
    System.out.println((int)(Math.random()*6)+1);
}
```

## 8. Valor aleatori entre 2 i 12

8º) – Feu una funció que anomenareu **dosDaus ()** i que cada cop que la cridem ens retornarà un valor **aleatori enter** entre **2 i 12**. Podem fer servir la funció **dau()**, a la que cridareu dos cops, sumant després els dos resultats obtinguts. **Pregunta:** Podem simular la tirada de dos daus amb una funció independent **dosDaus()** que calculi un valor aleatori entre **2 i 12** amb **return (int) (Math.random()\*11)+2**; i seria equivalent a l'anterior?. Perquè no?, justifiqueu-lo.

```
V.13 15/01/2022-14:37:34 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

static int dau(){
    return (int)(Math.random()*6)+1;
}

// Funció valor aleatori entre 2 i 12
static int dosDau(){
    return dau() + dau();
}
```

## 9. Número de xifres que conté

9º) – Realitzeu una funció amb la definició **static int lenNum (int num)** al que li passem un número natural en **num** i ens retornarà el número de xifres que conté. **Suggerència:** Passeu primer el número a cadena, i després determineu la longitud d'aquesta amb el mètode **length()**.

```
V.13 15/01/2022-14:08:15 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció per comptar la longitud d'un número
static int lenNum(int num){
    String s = Integer.toString(num);
    return s.length();
}
```

## 10. Parell o no parell (booleana)

10º) – Feu una funció a la que passarem un valor i ens indicarà si aquest és **parell** en una variable **booleana**. La definició de la **funció** serà de l'estil: **esParell (valor)**, on **valor** és el número **enter** (de tipus **int** o **long**) que volem comprovar si és o no **parell**, i la funció retornarà **true** si **valor** és un número parell, i **false** si no ho és. Un número **n** és parell si **n % 2 == 0**. Per fer aquesta funció, només caldrà que posem una línia al seu interior de l'estil: **return valor % 2 == 0**.

```
V.13 15/01/2022-14:08:50 192.168.1.130
Valle Alfaro Jonathan LAPTOP-5DAAC478

// Funció booleana per dir si és Parell o no
static boolean esParell(int valor){
    return valor%2==0;
}
```

## 11. NIF

11º) – Fer una funció **static String nif(int dni)** a la que li passarem la part numèrica del **NIF** d'una persona, 8 dígits i a partir d'aquesta informació, el programa determinarà la lletra corresponent i ens retornarà una cadena amb el **NIF** complet (el número i a continuació la lletra).

Per obtenir la lletra del **NIF**, a partir de la variable **dni**, que manté el valor del número del **NIF** fem:

**"TRWAGMYFPDXBNJZSQVHLCKE".charAt(dni % 23);** // Retorna un **char** El valor retornat per aquesta expressió és la lletra del **NIF** que necessitem. Recordeu que la funció retorna el **NIF**, i no només la lletra. Per fer això, haureu de passar la variable **dni** a cadena i concatenar-la a la lletra així obtinguda. Podeu fer: **Integer.toString(dni) + ...**

```
V.13 15/01/2022-14:56:34 Valle Alfaro Jonathan 192.168.1.130 LAPTOP-5DAAC478

// Funció retornar lletra del DNI
static String nif(int dni){
    return dni + String.valueOf("TRWAGMYFPDXBNJZSQVHLCKE".charAt(dni % 23));
}

public static void main(String[] args) {
    int x=0;
    Scanner sc = new Scanner(System.in);

    while(String.valueOf(x).length()!=8){
        try{
            System.out.println("Introduce 8 cifras");
            x = Integer.parseInt(sc.nextLine());
        }catch(Exception error){
            System.out.println("no has introducido 8 cifras ");
        }
    }
    System.out.println(nif(x));
}
```



## 12. Ordenar Vector

12º) - Feu una funció **static int[] ordenaTres(int n1, int n2, int n3)** a la que passarem tres valors enters **n1**, **n2** i **n3** i ens retornarà un vector on aquests tres valors apareixeran ordenats de menor a major. Fixeu-vos en la forma d'indicar que la funció retorna un vector. Recordeu que hi ha un mètode per ordenar objectes. Un cop emmagatzemat els tres valors al vector **v**, podem ordenar-lo amb una sola instrucció: **java.util.Arrays.sort(v)**;

Després haurem de retornar el vector ordenat. Si cridem a la funció com es mostra a continuació, el resultat haurà de ser el que es veu a sota:

```
public static void main(String[] args) {  
    System.out.println("Vector ordenat: " + java.util.Arrays.toString(ordenaTres(75,10,30)));  
}
```

```
V.13 15/01/2022-14:09:59 192.168.1.130  
Valle Alfaro Jonathan LAPTOP-5DAAC478  
  
// Funció per ordenar un vector  
static int[] ordenaTest(int n1, int n2, int n3){  
    int v[] = {n1,n2,n3};  
    Arrays.sort(v);  
    return v;  
}
```

## 13. Pedra, paper i tisora

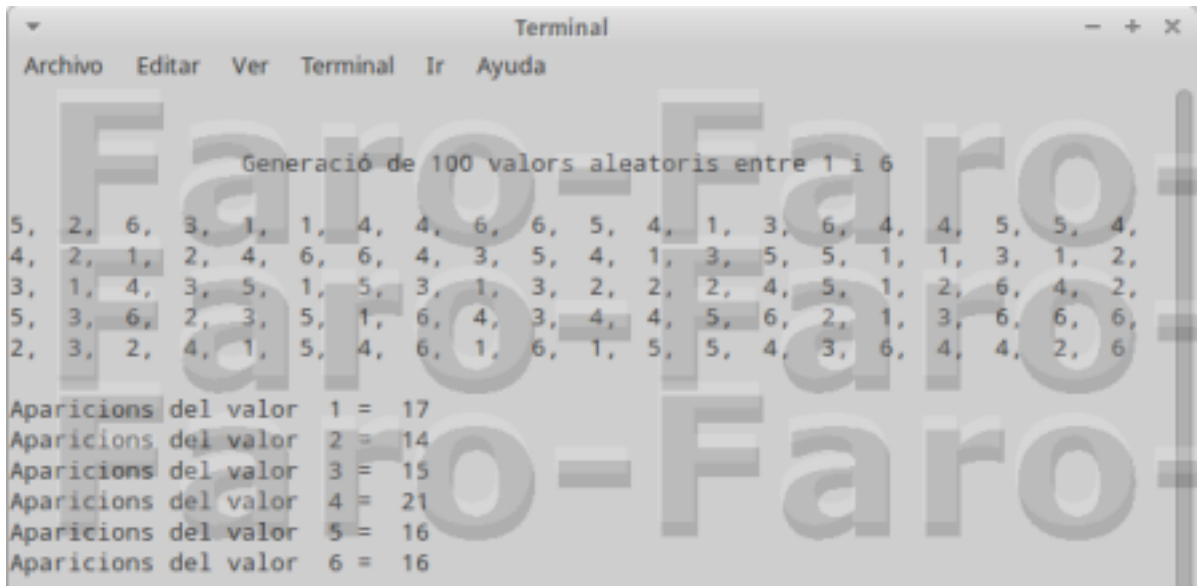
13º) – Feu una funció per jugar a **pedra**, **paper** i **tisora**. Cada cop que la invoquem, la funció ens retornarà la seva elecció entre aquestes tres opcions de forma aleatòria. La definició d'aquesta funció podria ser semblant a: **pedraPaperTisora ()**. La funció ens retornarà o be “pedra”, o be “paper”, o be “tisora”. Consulteu la documentació **Generació de números aleatoris en java** per saber com procedir. Us podeu definir un vector de cadenes amb aquests tres valors i retornar aleatòriament una cadena del vector i llavors la funció esdevé molt fàcil de programar.

```
V.13 15/01/2022-14:14:28 192.168.1.130  
Valle Alfaro Jonathan LAPTOP-5DAAC478  
  
// Funció de Pedra, Papel o Tijera  
static void pedraPaperTijera(){  
    String respostes[] = {"pedra", "paper", "tisora"};  
    int aleatori = (int)(Math.random()*3)+1;  
    switch(aleatori){  
        case 1:  
        System.out.println("pedra");  
        break;  
        case 2:  
        System.out.println("paper");  
        break;  
        case 3:  
        System.out.println("tisora");  
        break;  
    }  
}
```



## 14. 100 números aleatoris entre 1 i 6

14º) – Fent servir la funció **Dau** de l'exercici 4, de **Funcions**, feu un programa que generarà aleatòriament **100** valors **enters** entre **1** i **6**, i després ens informarà de la quantitat d'aparicions de cada valor. La sortida del programa serà de l'estil:



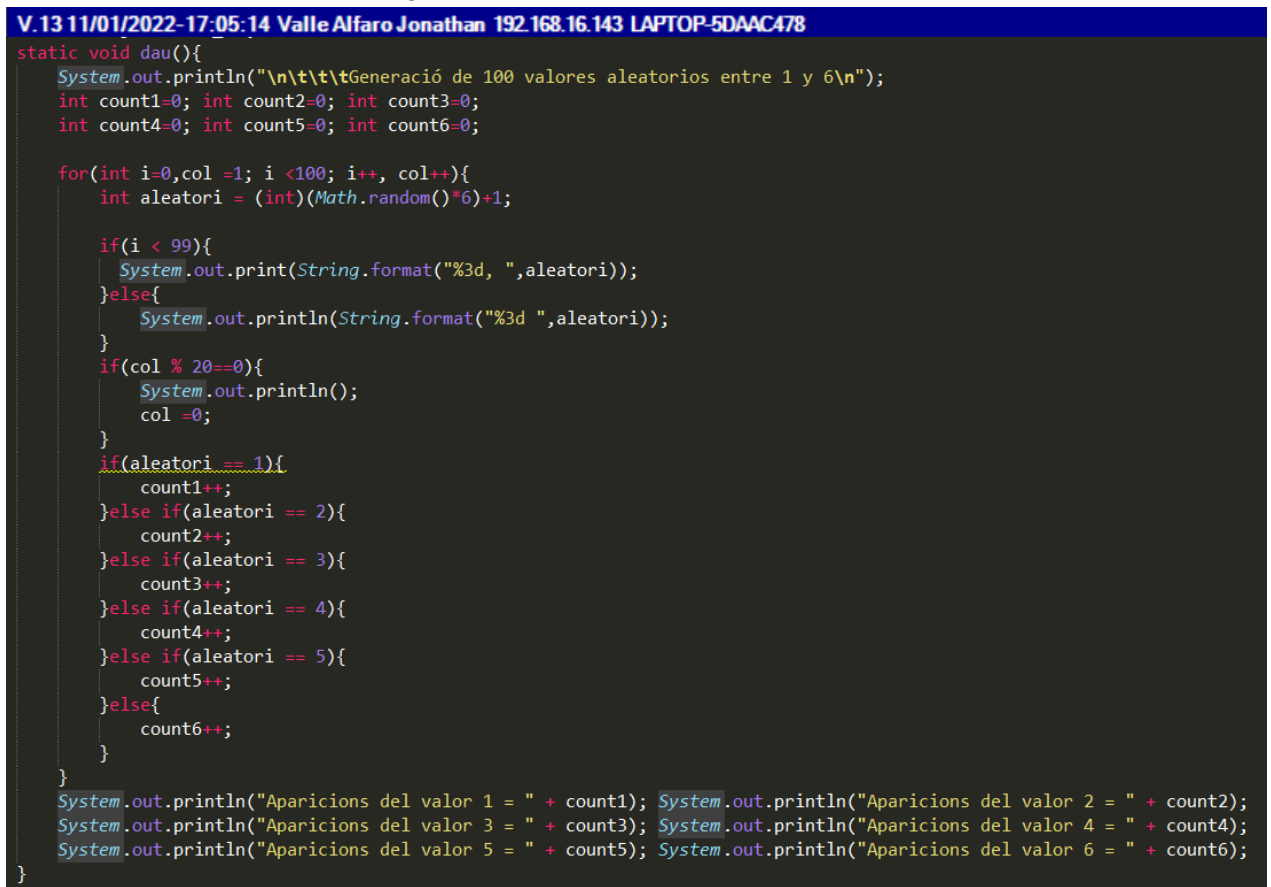
```
Terminal
Archivo  Editar  Ver  Terminal  Ir  Ayuda

Generació de 100 valors aleatoris entre 1 i 6

5, 2, 6, 3, 1, 1, 4, 4, 6, 6, 5, 4, 1, 3, 6, 4, 4, 5, 5, 4,
4, 2, 1, 2, 4, 6, 6, 4, 3, 5, 4, 1, 3, 5, 5, 1, 1, 3, 1, 2,
3, 1, 4, 3, 5, 1, 5, 3, 1, 3, 2, 2, 2, 4, 5, 1, 2, 6, 4, 2,
5, 3, 6, 2, 3, 5, 1, 6, 4, 3, 4, 4, 5, 6, 2, 1, 3, 6, 6, 6,
2, 3, 2, 4, 1, 5, 4, 6, 1, 6, 1, 5, 5, 4, 3, 6, 4, 4, 2, 6

Aparicions del valor 1 = 17
Aparicions del valor 2 = 14
Aparicions del valor 3 = 15
Aparicions del valor 4 = 21
Aparicions del valor 5 = 16
Aparicions del valor 6 = 16
```

Fixeu-vos que no tots els valors apareixen el mateix nombre de vegades (cosa que d'altra banda seria impossible, ja que **100** no és divisible entre **6**). A l'exemple, el valor que menys vegades apareix és el **2**, mentre que el **4** és el que més. Evidentment, cada cop que provem el programa, el resultats seran diferents. **Nota important:** No és necessari utilitzar un vector de **100** elements per fer aquest programa.



```
V.13 11/01/2022-17:05:14 Valle Alfaro Jonathan 192.168.16.143 LAPTOP-5DAAC478

static void dau(){
    System.out.println("\n\t\t\t\t\tGeneració de 100 valores aleatorios entre 1 y 6\n");
    int count1=0; int count2=0; int count3=0;
    int count4=0; int count5=0; int count6=0;

    for(int i=0,col =1; i <100; i++, col++){
        int aleatori = (int)(Math.random()*6)+1;

        if(i < 99){
            System.out.print(String.format("%3d ",aleatori));
        }else{
            System.out.println(String.format("%3d ",aleatori));
        }
        if(col % 20==0){
            System.out.println();
            col =0;
        }
        if(aleatori == 1){
            count1++;
        }else if(aleatori == 2){
            count2++;
        }else if(aleatori == 3){
            count3++;
        }else if(aleatori == 4){
            count4++;
        }else if(aleatori == 5){
            count5++;
        }else{
            count6++;
        }
    }
    System.out.println("Aparicions del valor 1 = " + count1); System.out.println("Aparicions del valor 2 = " + count2);
    System.out.println("Aparicions del valor 3 = " + count3); System.out.println("Aparicions del valor 4 = " + count4);
    System.out.println("Aparicions del valor 5 = " + count5); System.out.println("Aparicions del valor 6 = " + count6);
}
```

## EXERCICIS OPCIONALS

Aquests exercicis estan destinats a aquells alumnes que presenten un nivell d'aprenentatge superior i serveixen per adaptar els diferents ritmes de treball a l'aula. Recordeu que els anteriors son obligatoris, i aquests no. No deixeu de fer exercicis obligatoris per fer-ne d'opcionals. Aquests exercicis, no cal que els lliureu, però és convenient que intenteu fer-los si teniu temps. A partir d'ara, tots els exercicis hauríeu de plantejar-los en base a funcions i procediments, que puguin facilitar-vos la feina.

15º) – Realitzeu una funció **static int tallesRoba (String lletres)**, al que li passarem les lletres de la talla en **lletres** i ens retornarà un enter amb la talla numèrica equivalent, o un **0** si hi ha un error a **lletres**. Les conversions que permetrà el programa seran “XS” (Extra petita) = **34**, “S” (Petita) = **36**, “M” (Mitjana) = **40**, “G” (Gran) = **42**, “XL” ó “1X” (Extra Gran) = **46**, “XXL” ó “2X” = **52**, “XXXL” ó “3X” = **56**. La forma més senzilla de fer això en java és fer servir un diccionari, on les claus serien les lletres de la talla. Podeu consultar com utilitzar els diccionaris en java a: <http://programacionextrema.com/2015/11/06/diccionarios-en-java/> Volem que el programa ens retorni **0** si busquem una clau que no existeixi al diccionari (introducció d'una denominació de talla incorrecta), altres valors de retorn no ens serveixen. Al java, si busquem una clau que no existeix en un diccionari es retorna **none**. Per saber si una clau existeix en un diccionari podem executar per exemple:

**if (tallesDic.containsKey(talla))**

16º) – Feu una funció anomenada **static String triangle (int a, int b, int c)**, al que passarem en **a, b, c** les longituds dels costats d'un triangle i ens retornarà una **cadena** indicant el tipus de triangle en funció de la longitud dels seus costats. Els possibles valors a retornar son: “**No és un triangle**”, si els valors de **a, b i c** no permeten formar un triangle, “**equilàter**” (tots els costats iguals, es a dir, **a = b = c**), “**isòsceles**” (dos costats iguals i un de diferent) i “**escalè**” (tots tres costats de longitud diferent).

<b>MÒDUL M03</b>	<b>Realitzat per : José Javier Faro Rodríguez</b>	<b>FULL 4 - 5</b>
------------------	---	-------------------

<b>FUNCIONS</b>	<b>CICLE FORMATIU DE GRAU SUPERIOR DE ADMINISTRACIÓ DE SISTEMES INFORMÀTICS EN XARXA</b>	<b>DATA : 07/12/21</b>
-----------------	--	------------------------

17º) – Feu una funció **static double descompte (double preu, double descompte)**, al que li passem el preu d'un article en €, i el % descompte a aplicar (exemple **5, 10**) i ens retorna el preu de venda un cop aplicat el descompte sol·licitat. **preu** i **descompte** han de ser valors en coma flotant i el valor de retorn també.

18º) – Feu una funció **static int numLloses (double llarg\_llosa, double ampla\_llosa, double area)** a la que passem la grandària de les lloses que fem servir (**ampla i alt**), i l'àrea de la zona que volem enllosar i ens calculi quantes ens fan falta suposant que no es perd material. Totes les dades hauran d'estar a les mateixes unitats evidentment, i el valor de retorn haurà de ser un número enter, arrodonit per sobre. Per exemple, si el càlcul ens dona que necessitem **127,1** lloses, el programa haurà de retornar **128**. El que heu de fer és calcular l'àrea d'una llosa (costat per costat) i dividir l'àrea total entre l'àrea de la llosa per saber quantes es necessiten. Com es fa en java per arrodonir per sobre?. Consulteu la teoria.

19º) – Realitzeu una funció **facturaTelefonica (durada, cost\_establiment, cost\_minut)**, al que passem la duració d'una trucada telefònica en minuts, el cost d'establiment i el cost per minut que ens factura la companyia i ens retornarà el cost total d'aquesta trucada en euros.

20º) – Programeu la funció **quansSegons (dies, hores, minuts, segons)** al que li passem un

període de temps en **dies**, **hores**, **minuts** i **segons** i ens retorna aquest temps convertit a segons. Recordeu que un dia son **24** hores, una hora son **60** minuts i un minut son **60** segons.

21º) – Feu una funció per realitzar operacions binàries (amb operadors binaris). La funció s'anomenarà **opera (operand1, operador, operand2)**. **operand1** i **operand2** son els valors numèrics dels operands, on **operand1** és el que s'escriuria a la **esquerra** de l'operador, mentre que **operador** és l'operador. Els valors possibles de l'operador son: **“+”**, **“-”**, **“\*”**, **“/”**, (suma resta producte i quocient), **“\”** ó **“Div”** ó **“/”** – Divisió entera i **“%”** ó **“Mod”** – Resta de la divisió entera. Es a dir, la funció reconeix **9** operadors. Per exemple, si la cridem com **opera(5, “+”, 8)**, aquesta retornarà el valor **13**, ja que en calcular **5 + 8 = 13**. No cal que considereu que la funció provocarà una excepció en el cas de la **divisió** i la **divisió sencera** si **operand2 = 0**. Penseu que els operadors son cadenes, i que al java, no podem comparar la igualtat de dues cadenes amb l'operador **==**. Penseu també que l'operador **“/”** volem que retorni un número en coma flotant encara que els operands siguin sencers (el java no ho fa).

<b>MÒDUL M03</b>	<b>Realitzat per : José Javier Faro Rodríguez</b>	<b>FULL 5 - 5</b>
------------------	---	-------------------