

Casting i Aleatòris

1.- Càsting de variables primitives.

En ocasions és necessari **convertir** una variable (o una expressió en general) **d'un tipus a un d'altre**. Este procediment s'anomena **CASTING** o Conversió de Tipus primitius. També existeix Casting o Conversió d'objectes, que ja veurem en altres temes.

Recordem que els tipus de variables primitives o dades simples suportats per Java són:

- Per a números **enters**: **byte, short, int, long**.
- Per a números **reals**: **float, double**.
- Per a **caràcters**: **char**.
- Per valors **lògics**: **boolean**.

Per tant, fer un càsting és passar d'un tipus **int** a tipus **float**, o d'un **double** a **short**.

A l'hora de reslitzar un càsting hem de tindre en compte que:

- **NO** podem fer un càsting entre una variable **boolean** y qualsevol altra variable primitiva
- **SI** podem fer un càsting entre una variable **char** i una variable que emmagatzeme números **enters** (**byte, short, int, long**).

Int int char boolean float double short

Dins del càsting de variables primitives trobem dos tipus:

- **Càsting implícit**: No necessita escriure codi, es tracta d'una conversió ampla (**widening casting**), es a dir, quan coloquem un valor **xicotet** en un contenidor **gran**.

```
int num1 = 100;

long num2 = num1; // Un int cap en un long
long num2 = 100; // 100 en un long

byte num4 = 0;
short num5 = num4; // Un byte cap en un short
```

- **Càsting explícit**: Sí és necessari escriure codi. Es tracta d'una conversió estreta (**narrowing casting**), es a dir, quan coloquem un valor **gran** en un contenidor **xicotet**. (OJO!! fer este casting pot suposar la perdua de dades)

```
int num1 = 100;

short num2 = (short) num1; // Ací fa falta un casting explícit:
                          // short té menor capacitat que int

float num3 = 3.1415f;
double num4 = 3.141516;
int num5;

num5 = (int) num3; // int té menor capacitat que float
num5 = (int) num4; // int té menor capacitat que double
```

La forma general de realitzar un casting és:

(tipus) valor_a_convertir

Exemple amb (char). Recorda que sols podem fer castinf entre **char** i una variable que emmagatzeme números **enters** (**byte**, **short**, **int**, **long**).

```
public static void main(String[] args) {

    // 65 -> codi ascii de A
    // casting de int a char

    int num1 = 65;
    char lletra = (char) num1;

    System.out.println(lletra);
    System.out.println((char) 65);

    // casting de char a int

    char lletra2='A';
    int num2 = (int)lletra2;

    System.out.println(num2);
    System.out.println((int)'A');

}
```

```
<terminated> casting [Java Application]
A
A
65
65
```

NOTA IMPORTANT (float):

A l'exemple anterior trobem l'expressió: **float** num3 = 3.1415f;

En Java sempre que escrivim un número en decimal (amb coma) l'interpreta automàticament com un double (sempre), si posem:

float num3 = 3.1415; el compilador avisa d'un error, per això hi ha que fer un càsting explícit per que compile, que es pot fer de dos maneres:

```
float num = (float)3.1415; // casting explícit
float num = 3.1415f;      // casting abreujat (sols en floats)
```

(OJO!! el casting pot suposar la perda de dades)

```
public static void main(String[] args) {

    int num1 = 1000000;
    short num2 = (short) num1;

    System.out.println("El valor de num2 és: " + num2 );

}
```

```
<terminated> casting [Java Application]
El valor de num2 és: 16960
```



Si recordem un **short** té un rang entre -32768 i 32767, per tant, al realitzar el casting amb un valor més gran del que pot representar (**int** num1 = 1000000;) el resultat és incongruent. (num = 16960)

(OJO!! el casting pot suposar la perda de dades)

```
public static void main(String[] args) {
    int x = 2;
    int y = 9;
    double divisio;

    divisio = (double) y / (double) x;



    System.out.println("El resultat de la di-
    visión és: " + divisio);
}
```

Console  
 <terminated> casting [Java Application] C:\Pro
 El resultat de la división és 4.5

```
public static void main(String[] args) {
    int x = 2;
    int y = 9;
    double divisio;

    divisio = y / x;

    System.out.println("El resultat de la
    división és " + divisio);
}
```

Console  
 <terminated> casting [Java Application] C:\Progr
 El resultat de la división és 4.0

Primer tenim:

```
divisio = (double) y / (double) x;
```

(**double**) y converteix el valor d' y en un **double**, és a dir, en un número amb decimals i (**double**) x fa el mateix amb x, per tant, l'operació de divisió quedaria com 9.0 / 2.0. En tractar-se d'una divisió entre nombres decimals, Java entén que el resultat ha de tindre també decimals i usa tota la precisió disponible en el tipus. El resultat que es guarda en la variable **divisio** és 4.5 que és el valor correcte.

En el segon cas tenim:

```
divisio = y / x;
```

Ara la divisió és 9 / 2. Java veu una divisió entre dos nombres enters, així que el resultat que ofereix és un altre nombre enter després de menysprear els decimals si n'hi haguera. El resultat que dona Java és la divisió sencera, és a dir, 4. El següent pas és una assignació, tindríem **divisio** = 4;. Com la variable **divisio** és de tipus **double**, el valor 4 es guarda en realitat com 4.0;

!!!COSAS DEL JAVA!!!

Recorda : Fes el *casting* per a conservar els decimals en les divisions amb enters

La divisió entre **dos números enters** es un altre número enter. Per a conservar els decimals de la divisió hi ha que fer un *casting* a **float** o **double**.

2.- RANDOM- Números Aleatoris.

6. Números aleatoris

Els números aleatoris s'utilitzen amb freqüència en programació per a emular el comportament d'algun fenomen natural, el resultat d'un joc d'atzar o, en general, per a generar qualsevol valor que a priori no es pot predeir.

Per exemple, es poden utilitzar números aleatoris per a generar tirades de daus de tal forma que, per endavant, no es pot saber el resultat. Abans de tirar un dau no sabem si eixirà un 3 o un 5; es tractarà doncs d'un número imprevisible; el que sí que sabem és que eixirà un número entre l'1 i el 6, és a dir, podem delimitar el rang dels valors que obtindrem de manera aleatòria.

- Generació de números aleatoris amb i sense decimals.

Per a generar valors aleatoris utilitzarem Math.random(). Aquesta funció genera un número amb decimals (de tipus double) en l'interval (0 – 1], és a dir, genera un número major o igual que 0 i menor que 1.

El següent programa genera **deu números aleatoris**:

```
public class Aleatoris1 {

    public static void main(String[] args) {

        System.out.println("Deu números aleatoris:\n");

        for (int i = 1; i < 11; i++) {

            System.out.println(Math.random());

        }

    }

}
```

```
<terminated> Aleatoris1 [Java Application] C:\Program Files\Java\jdk-9.0.4\bin\java.exe
Diez números aleatorios

0.17473738174501852
0.6922222680981228
0.6828491821798811
0.8446665847689957
0.5516120435419309
0.6423106628729469
0.28126952501104496
0.0340944875144773
0.33733038585090624
0.0044205912925116
```

Si executem varies voltes el programa podrem observar que **cada volta ixen números diferents**, encara que sempre estan compresos entre 0 i 1 (incloent el 0, no aplega mai a 1).

Pensaràs que no és molt útil generar números aleatoris entre 0 i 1 si el que volem és per exemple traure una carta a l'atzar de la baralla espanyola; però en realitat un nombre decimal entre 0 i 1 és l'única cosa que ens fa falta **per a generar qualsevol tipus de valor aleatori sempre que es manipule etse número de la forma adecuada**.

Per exemple, si volem generar **valors aleatoris entre 0 i 10 (incloent el 0 i sense arribar a 10)** simplement haurem de córrer la coma un lloc o, cosa que és el mateix, **multiplicar per 10**.

```
public class Aleatoris2 {

    public static void main(String[] args) {

        System.out.println("20 números aleatoris entre 0 y 10");
        System.out.println(" sense aplegar a 10 (amb decimals):");

        for (int i = 1; i <= 20; i++) {

            System.out.println(Math.random()*10);

        }

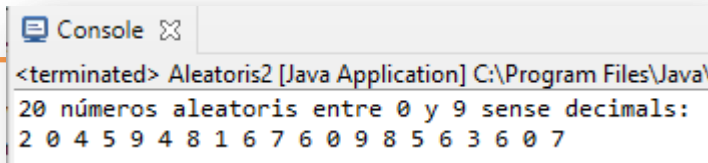
    }

}
```

```
<terminated> Aleatoris2 [Java Application] C:\Program Files\Java\jdk-9.0.4\bin\java.exe
20 números aleatoris entre 0 y 10
 sense aplegar a 10 (amb decimals):
6.208795976840991
2.186081009989901
8.1054371487536
1.2032716874175198
3.767190573451896
2.1182156731128887
8.73076157093791
1.0594263212876798
2.948832268966245
4.066261631310689
7.986814390140209
6.80566245202203
1.0971109049385386
3.7860451459074476
7.642863299983987
4.894027315591719
2.3141549600440126
4.755796576389622
6.4516165484726855
4.268653772542499
```

Si volem generar números **enters** entre **0 i 9**, sols tenim que fer un **casting** per a convertir valors de tipus double a valors de tipus int. (**int**) `valor_double`. Al passar un valor float o doble a **int** perd tots els decimals.

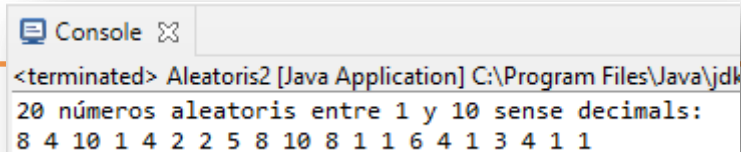
```
public class Aleatoris3 {
    public static void main(String[] args) {
        System.out.println("20 números aleatoris entre 0 y 9 sense decimals:");
        for (int i = 1; i <= 20; i++) {
            System.out.print((int) (Math.random()*10) + " ");
        }
    }
}
```



```
<terminated> Aleatoris2 [Java Application] C:\Program Files\Java\
20 números aleatoris entre 0 y 9 sense decimals:
2 0 4 5 9 4 8 1 6 7 6 0 9 8 5 6 3 6 0 7
```

I si en lloc de generar nombres enters entre **0 i 9** volem generar números entre **1 i 10**? En este cas, simplement caldria sumar 1 al número generat, d'aquesta manera es "desplacen un pas" els valors generats a l'atzar, de tal forma que el **mínim** valor que es produeix seria **el 0 + 1 = 1** i el **màxim** seria **9 + 1 = 10**.

```
public class Aleatoris4 {
    public static void main(String[] args) {
        System.out.println("20 números aleatoris entre 1 y 10 sense decimals:");
        for (int i = 1; i <= 20; i++) {
            System.out.print((int) (Math.random()*10+1) + " ");
        }
    }
}
```



```
<terminated> Aleatoris2 [Java Application] C:\Program Files\Java\jdk
20 números aleatoris entre 1 y 10 sense decimals:
8 4 10 1 4 2 2 5 8 10 8 1 1 6 4 1 3 4 1 1
```

Ho posarem una poc més difícil. Ara generarem nombres enters entre **50 i 60 tots dos inclosos**. Primer multipliquem `Math.random()` per **11**, amb el que obtenim nombres decimals entre **0 i 10.9999...** (sense arribar mai fins a 11). Després desplacem este interval sumant **50** pel que obtenim nombres decimals entre 50 i 60.9999... Finalment, llevem els decimals fent **casting** i ja tenim nombres enters aleatoris entre 50 i 60 tots dos inclosos..

```

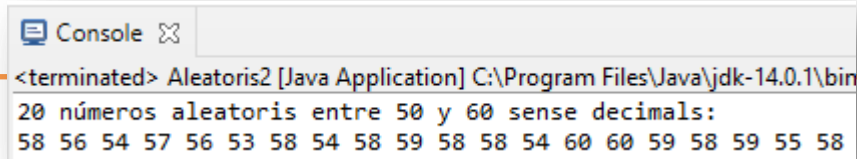
public class Aleatoris5 {

    public static void main(String[] args) {

        System.out.println("20 números aleatoris entre 50 y 60 sense decimals:");

        for (int i = 1; i <= 20; i++) {
            System.out.print(((int) (Math.random()*11)+50) + " ");
        }
    }
}

```



```

<terminated> Aleatoris2 [Java Application] C:\Program Files\Java\jdk-14.0.1\bin
20 números aleatoris entre 50 y 60 sense decimals:
58 56 54 57 56 53 58 54 58 59 58 58 54 60 60 59 58 59 55 58

```

En resum:

Códi Java	Número aleatori generat
<code>Math.random();</code>	Tipo double entre 0.0 y 1.0
<code>(int) (Math.random() * (N+1));</code>	Tipo int entre 0 y N
<code>(int) (MIN + Math.random() * (MAX - MIN + 1));</code>	Tipo int entre MIN y MAX

Exemples:

```

int x = (int) (Math.random() * (10+1)); // int aleatori entre 0 y 10

int y = (int) (20 + Math.random() * (30-20+1)); // int aleatori entre 20 y 30

```

- Generació de paraules de manera aleatòria d'un conjunt donat

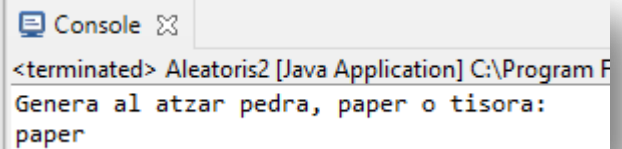
Hem vist com generar números aleatoris amb i sense decimals i en diferents intervals. Produïrem ara de manera aleatòria una paraula - pedra, paper o tisora - generant primer un nombre enter entre 0 i 2 i posteriorment fent correspondre una paraula a cada número.

```
public class Aleatoris6 {
    public static void main(String[] args) {

        System.out.println("Genera al atzar pedra, paper o tisora:");

        int ma = (int)(Math.random()*3);
        // genera un número al atzar entre 0 y 2 ambdós inclosos

        switch(ma) {
            case 0:
                System.out.println("pedra");
                break;
            case 1:
                System.out.println("paper");
                break;
            case 2:
                System.out.println("tisora");
                break;
            default:
                break;
        }
    }
}
```



Com podríem generar un dia de la setmana de manera aleatòria? En efecte, primer generem un número entre 1 i 7 tots dos inclusivament i després fem correspondre un dia de la setmana a cadascun dels números.

```
public class Aleatoris7 {
    public static void main(String[] args) {
        System.out.println("Mostra un día de la semana al atzar:");
        int dia = (int)(Math.random()*7) + 1; // genera un número aleatori entre el 1 i el 7

        switch(dia) {
            case 1:
                System.out.println("dilluns");
                break;
            case 2:
                System.out.println("dimarts");
                break;
            case 3:
                System.out.println("dimecres");
                break;
            case 4:
                System.out.println("dijous");
                break;
            case 5:
                System.out.println("divendres");
                break;
            case 6:
                System.out.println("dissapte");
                break;
            case 7:
                System.out.println("diumenge");
                break;
            default:
                break;
        }
    }
}
```

