1.00 Clase 3

Tipos de datos básicos en Java, Estructuras de control

Tipos de datos en Java

- **8 tipos de datos primitivos o integrados:**
  - **4 tipos enteros (*byte, short, int, long*).**
  - **2 tipos de coma flotante (*float*, *double*).**
  - **Booleano (*boolean*).**
  - **Carácter (*char*).**
- **No son objetos.**
- **Se definen de forma (casi) idéntica en cada máquina en la que se ejecuta Java, algo que no ocurre en otros lenguajes de programación.**
- **Java es un lenguaje de programación fuertemente tipado:**
  - **Cada variable debe tener un tipo declarado.**

Tipos de datos en Java

Tipo	Tamaño (en bits)	Rango
byte	8	-128 a 127
short	16	-32,768 a 32,767
int	32	-2,147,483,648 a 2,147,483,647
long	64	-9,223,372,036,854,775,808L a 9,223,372,036,854,775,807L
float	32	+/- 3.4E+38F (6-7 dígitos importantes)
double	64	+/- 1.8E+308 (15 dígitos importantes)
char	16	Conjunto de caracteres Unicode ISO
boolean	1	verdadero o falso

¿Qué tipo de dato utilizaría?

- Qué utilizaría para almacenar:
  - La velocidad de la luz.
  - Su calificación en el curso 1.00.
  - Su nota media de este trimestre.
  - El número de frigoríficos de una habitación.
  - Localización de un punto en la pantalla.
  - 2^{65}
  - 234,77 dólares.
  - La mitad de 234,77 dólares.
  - Bits por segundo transmitidos por un módem.

¿Qué tipo de datos utilizaría?

- **Lo que utilizaría para almacenar:**
  - **La velocidad de la luz.** ***double***
  - **Su nota en el curso 1.00.** ***char***
  - **Su nota media este trimestre.** ***double/float***
  - **El número de frigoríficos de una habitación.** ***int***
  - **Localización de un punto en la pantalla.** ***float/int***
  - **2^{65}** ***BigInteger***
  - **234.77 dólares.** ***double/int***
  - **La mitad de 234.77 dólares.** ***double/int***
  - **Bits por segundo transmitidos por un módem.** ***int/float***

Cómo utilizar los tipos de datos de Java

```
public class DataTypes  {
    public static void main(String[]  args) {
        boolean isReal=true;     // Los nombres son sensibles a
                                 // mayúsculas y minúsculas,
                                 // deben empezar por una letra y
                                 // pueden contener números,_,$
        byte d= 122;             // Deben ser inferiores a 127
        short e= -29000;         // Deben ser inferiores a 32767
        int  f= 100000;          // Deben ser inferiores a 2100 mill.
        long g= 999999999999L;   // Deben poner L al final
        float h= 234.99F;        // Deben ser < 3E38; F al final
        double i= 55E100;
        char  cvalue= '4';        // char '4' no es el entero 4

        //Las cadenas (strings) son objetos, no primitivos.
        //Ejemplo: String nombre= "Claudius";
    }
}
```

Operadores aritméticos

Tabla por orden de prioridad (mayor precedencia al principio)

Operadores	Significado	Asociatividad
++	incremento	Derecha a izquierda
--	decremento	
+ (unario)	unario + (x = +a)	
- (unario)	unario - (x = -a)	
*	multiplicación	Izquierda a derecha
/	división	
%	Resto (mod)	
+	suma	Izquierda a derecha
-	resta	

Cómo utilizar operadores aritméticos

```
public class DataType2 {
    public static void main(String[]  args) {

        int  j, k, m;
        int  d= 123;
        j= d--;                 // j vale 122 y d vale 123
        System.out.println("j= " + j);
        k= ++d;                 // k vale 124 y d vale 124
        System.out.println("k= " + k);
        m= --d;                 // m vale 123 y d vale 123
        System.out.println("m= " + m);

        m= k % j;               // operador Resto para los tipos int
                                // k=124 y j=122, por tanto, m= 2
        System.out.println("m= " + m);
        j= 5; k= 3; m= j/k;     // División entera: m= 1

        System.out.println("m= " + m);
        System.exit(0);

    }
}
```

Operadores lógicos

- **Producen resultados de tipo booleano.**
- **En las comparaciones se utilizan 8 operadores:**

Igual	==	Distinto	!=
Menor que	<	Menor o igual que	<=
Mayor que	>	Mayor o igual que	>=
Conjunción lógica (Y)	&&	Disyunción lógica (O)	||

  - **Existen también operadores *bitwise* que utilizaremos próximamente.**
- **Ejemplo:**

```
double c= 0.0, b= 3.0;
if (c != 0.0 && b/c > 5.0) System.out.println("Boo");
// Nunca utilice == (igual) con float/double (es un mal ejemplo)
// Evaluación por corto circuito: salir tras una subexpresión falsa
```

Operadores de asignación

- **Asignación no es lo mismo que igualdad:**
  - **El operador = no es igual al operador ==**
- **Las asignaciones son expresiones:**

```
int x, y;
x= y= 5;    // Es lo mismo que x = (y= 5); asociación de dcha. a izq.
```

- **Existen formas abreviadas:**

```
int  x= 5, y= 3;
x += y;          // Es lo mismo que x= x + y;
```

- **Entre las formas se incluyen: +=, -=, *=, /=, &=, ^=, |=, %=**

Ejercicios

- **Calcule el porcentaje de estudiantes de postgrado (licenciados) que hay en el curso 1.00.**

```
int  estudiantes= 240;
int  licenciados= 35;
_____________________;
```

- **Represente 15*i correctamente:**

```
int  i= 100000000 + 100000000;
_____________________;
```

- **Escriba una expresión para comprobar si *int x* es mayor que *double y*, si *x* es menor que y^2 y si *x* es distinto de x^2:**

```
_____________________;   // Declare x, y
if (_____________________;   // Escriba la expresión lógica
```

- **Incremente *int z* con *int* a:**

```
_____________________;    // Declare a, z
                          // Incremente z con a
```

Ejercicios

- **Calcule el porcentaje de estudiantes de postgrado (licenciados) del curso 1.00:**

```
int  estudiantes= 240;
int  licenciados= 35;
double pctLic= licenciados/(double) estudiantes;
```

- **Represente 15*i correctamente:**

```
int  i= 100000000 + 100000000;
long j= 15*i;
```

- **Escriba una expresión para comprobar si *int x* es mayor que *double y*, si *x* es menor que y^2 y si *x* es distinto de x^2:**

```
int  x; double y;
if (x > y && x < y*y && x != x*x) …
```

- **Incremente *int z* con *int a*:**

```
int  a;
int  z += a;
```

Estructuras de control: Ramificación

Forma general	Ejemplo

if (condición)
  sentencia;
	if (x == y)
	  a = 20;
	if (x ==z) {
	  b = 10;
	  c = 20; }

if (condición)
  sentencia 1;
else
  sentencia 2;
	if (x == y) {
	  a = 10;
	  b = 20; }
	else
	  x = y;

if (condición1)
  sentencia1;
…
else if (condiciónN)
  sentenciaN;
else
  sentencia;
	if (x > 60)
	  y = 20;
	else if (x < 30) {
	  z += y;
	  y = 25; }
	else
	  y= 40;

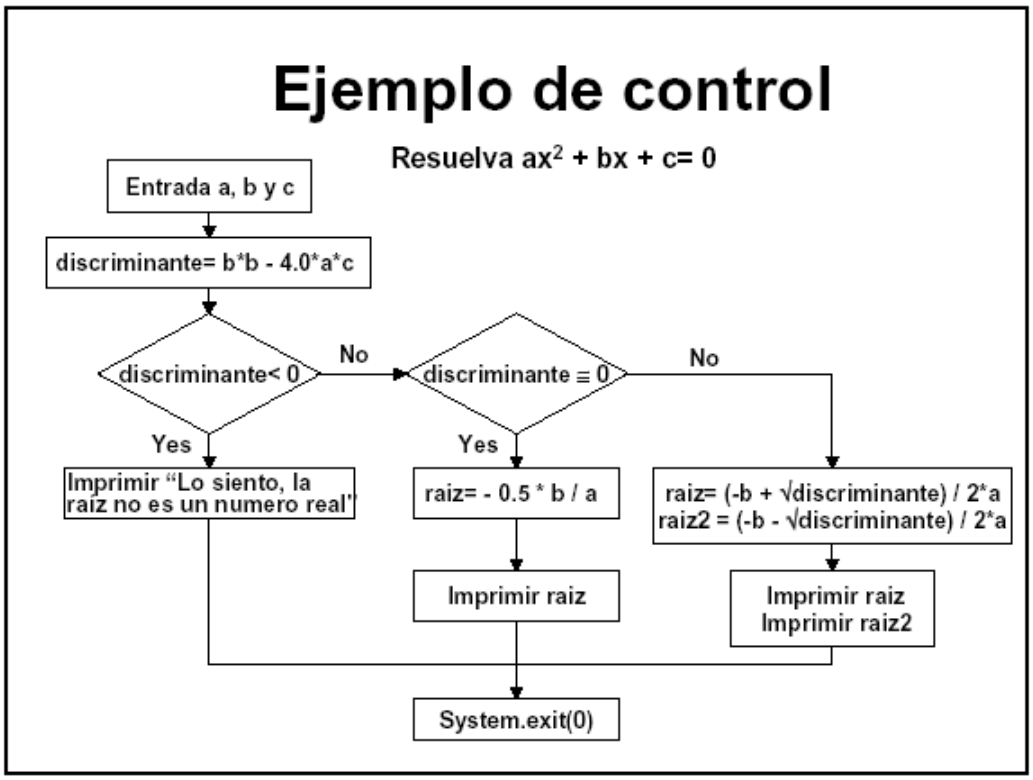
Estructuras de control: Ramificación

- **Una sentencia se puede reemplazar por una serie o conjunto de sentencia entre llaves: { }**
- **Una cláusula *else* enlaza con su sentencia *`if`* más cercana.**
  - **Si desea asociarla con un *if* más lejano, añada enunciados *`if`* intermedios entre { }.**

```
if (x > 3) {
 if (y < 7) z= 5; }
else           // Este else va enlazado con if (x > 3)
 z = y;
```

- **Sentencia de comparación en forma de terna (tres sentencias en una)**
  - **(condición ? expr_si_es_verdadera : expr_si_es_falsa)**

```
boolean esCelsius= true;
double tC= 10.0;
double mostrarTemp= (esCelsius ? tC : 1.8*tC + 32.0);
```



Ejemplo de control

```
import  javax.swing.*;              // Para soportar la entrada simple
public class Control {                   // fórmula cuadrática
    public static void main(String[]  args) {
        final double TOL= 1E-15;       // Constante (utilice 'final')
        String entrada=  JOptionPane.showInputDialog("Introduzca a");
        double a= Double.parseDouble(entrada);
        entrada=  JOptionPane.showInputDialog("Introduzca b");
        double b= Double.parseDouble(entrada);
        entrada=  JOptionPane.showInputDialog("Introduzca c");
        double c= Double.parseDouble(entrada);
        double  discriminante= b*b -  4.0*a*c;
        if (  discriminante < 0)
            System.out.println("Lo siento, la raíz no es un número real");
        else if (Math.abs(discriminante) <= TOL) {
            double raiz= -0.5 * b / a;
            System.out.println("La raíz es" + raiz); }
        else {   // Redefinir 'raíz'; los bloques tienen sus propios ámbitos
            double raiz=(-b + Math.sqrt(discriminante))/ (2.0*a);
            double raiz2=(-b-  Math.sqrt(discriminante))/ (2.0*a);
            System.out.println("Raíces: " + raiz + " y " + raiz2;);  }
        System.exit(0);  } }
```

Ejemplo de control

- **El programa anterior presenta un error sutil e intencionado:**
  - **¿Puede verlo?**
  - **¿Es probable que lo encuentre al pasar las pruebas?**
  - **¿Es posible que lo encuentre utilizando el depurador y leyendo el código?**
- **Repare el error reorganizando el orden de las cláusulas *if-else*.**

Ejercicios de control

- **Ejercicio 1. Escriba el código de *main*()**
  - **Si la demanda d > las existencias s, aumente el precio p con a(d-s).**
  - **Si la demanda = las existencias, no haga nada.**
  - **Si la demanda d < las existencias s, baje el precio p con b(d-s).**
- **Ejercicio 2. Escriba el código de *main*()**
  - **Si el tiempo hoy está nublado, mañana lloverá.**
  - **Si el tiempo hoy es cálido y nublado, mañana será más cálido.**
  - **Si el tiempo hoy es soleado, mañana estará nublado.**
  - **Si el tiempo hoy es cálido y soleado, mañana será más frío y estará nublado.**
  - **Si hoy llueve, mañana saldrá el sol.**
  - **(Utilice *strings* para "nublado", "cálido", etc.).**

Soluciones al ejercicio de control

```
// Ejercicio 1. En Forte, declare e inicialice todas las variables.
if (d > s)
    p += a*(d-s);
else if (d < s)          // Se puede omitir "if (d < s)"
    p -= a*(d-s);        // Se puede omitir la cláusula de == dado
                         // que (d-s)=0
// Ejercicio 2. En Forte, declare e inicialice todas las variables.
if (tiempoHoy.equals("Soleado")) {
    tiempoMa= "Nublado";
    if (tempHoy.equals("Calido"))
        tempMa= "MasFrio";
    }
else if (tiempoHoy.equals("Nublado")) {
    tiempoMa= "Llueve";
    if (tempHoy.equals("Calido"))
        tempMa= "MasCalido";
}
else    // Lluvia
    tiempoMa= "Soleado";
```

Estructura de control: Iteración

Forma general	Ejemplo

while (condición)
  enunciado;
	while (x > 0) {
	  System.out.println("x= " + x);
	  x--;
	}

do
  enunciado;
while (condición);
// Siempre ejecuta el enunciado al
// menos una vez
	do {
	  System.out.println("x=" + x);
	  x--;
	} while (x > 0);

for (expr_inic; condic_fin; cuenta)
  enunciado;
	for (x= 20; x > 0; x--)
	  System.out.println("x=" + x);

Bucles *For*

```
for (expr_inic; condic_fin; cuenta)
   enunciado;
```

```
for (j= 0; j < 20; j++)
   z += j;
```

es equivalente a:

```
expr_inic;
while (condic_fin) {
  enunciado;
  cuenta;
}
```

```
j= 0;
while (j < 20) {
   z += j;
   j++;
}
```

Ejemplo 1 de iteración: Ln x

```
import  javax.swing.*;

public class Iteration {
    public static void main(String[]  args) {
        String entrada=  JOptionPane.showInputDialog("Introduzca x (0-2)");
        double x= Double.parseDouble(entrada);
        // Calcule 20 términos del
        // ln x= (x-1) -  (x-1)^2/2 + (x-1)^3/3 -  ...
        final int ITERACIONES= 20;
                                    // Número establecido de iteraciones
        double  logx= 0.0;
        double x1= x-1;
        for (int  i= 1; i <= ITERACIONES; i++) {
            if (i % 2 == 0)      // i par
                logx  -= Math.pow(x1, i)/i;


           else
               logx  += Math.pow(x1, i)/i;  }
        System.out.println("Ln  x= " +  logx);  }  }
```

Ejemplo 2 de iteración: Ln x

```
import  javax.swing.*;                // Misma serie que el ejemplo 1
public class Iteration2 {
    public static void main(String[]  args) {
        String entrada=JOptionPane.showInputDialog("Intr. x (0-2)");
        double x= Double.parseDouble(entrada);
        final double TOLERANCIA= 0.00001;  // Tol establece el n° de términos
        double  logx= 0.0;
        double x1= x-1;
        int  i= 1;
        double term= 0.0;                  // Defínala fuera de do {}
        do {
            term= Math.pow(x1, i)/i;
            if (i % 2 == 0)                // i par
                logx  -= term;
            else
                logx  += term;
            i++;
        } while (Math.abs(term) > TOLERANCIA);
        System.out.println("Ln  x= " +  logx);
        System.out.println("Encontrado en " + i + " iteraciones"); }  }
```