Universidad Zaragoza

Fundamentos de Informática

Lección 3. Tipos de datos, constantes y variables

Curso 2010-2011

José Ángel Bañares y Pedro Álvarez
24/09/2010. Dpto. Informática e Ingeniería de Sistemas.

Universidad Zaragoza
1542

Índice de contenidos

Universidad Zaragoza

Concepto de tipo de dato

todo algoritmo ⇨

- procesador
- entorno con los objetos a manejar (datos)
- acciones a realizar

⇩

abstracción de la realidad

propiedades de todo dato ⇨

- tiene un valor concreto (estado)
- está representado de un modo concreto
  - ✉ *algorítmicamente (notación sintáctica)*
  - ✉ *internamente (codificación)*
- pueden aplicársele unas acciones concretas
- *ámbito*

un conjunto de propiedades comunes determina un tipo de dato

Universidad Zaragoza
1542

Concepto de tipo de dato

- *para todo dato usado debe estar perfectamente especificado su tipo*
  - ✓ implícitamente (tipos predefinidos) o
  - ✓ explícitamente (declaración del tipo)

ventajas:
- ✓ > legibilidad (y documentación)
- ✓ > facilidad diseño correcto (detección de errores)

En los lenguajes fuertemente tipados se exige la declaración previa de los datos que se van a utilizar en el algoritmo

Universidad Zaragoza

Concepto de tipo de dato

especificación
tipo de dato ⇨

- dominio de valores
- representación habitual (interna y algorítmica)
- conjunto de operadores asociado
- nombre para identificarlo

- Ejemplo, tipo de dato **int**
  - Dominio de valores:[–2.147.483.648 .. ,0,.. 2.147.483.647]
  - Representación
    - Interna 4 bytes, representación en binario en complemento a 2.
      - 0000 0000
    - Externa: **<entero>::=[+|-]<digito>{<digito>}**
  - Conjunto de operadores: +,-,*, /, %

Universidad Zaragoza
1542

Clasificación de tipos de datos

- En función de quién los define
  - Tipos de datos estándar
  - Tipos de datos definidos por el usuario
- En función de si representan una “agrupación de datos o no”
  - Tipos de datos simples/primitivos
  - Tipos de datos estructurados/agrupacione

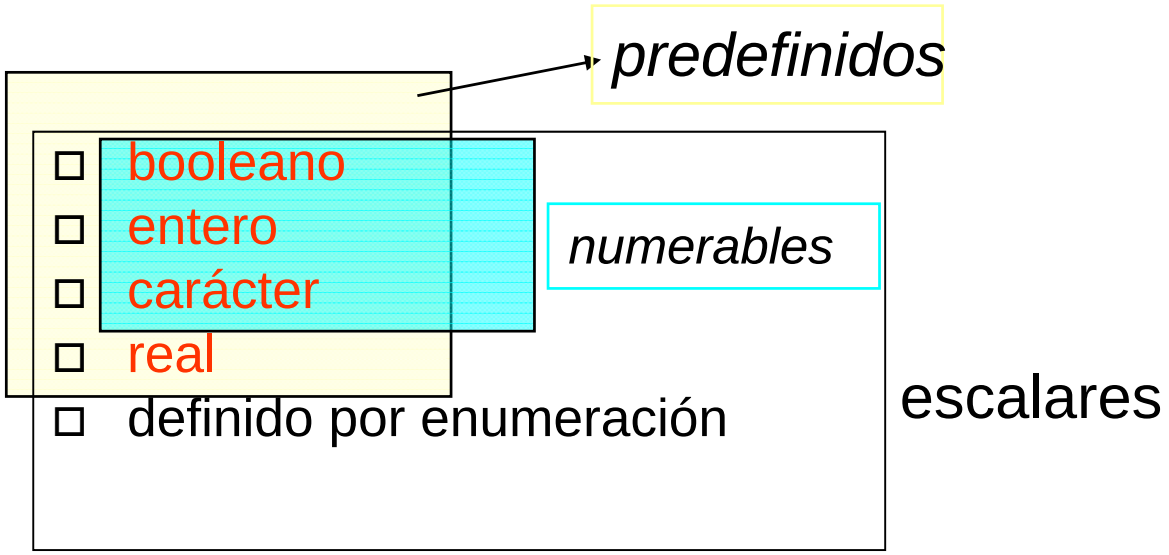
Universidad Zaragoza
1542

Tipos de datos primitivos

- tipos de datos primitivos
  - Datos de tipo numérico
    - Números enteros: Byte, int ,short,long
    - Números en coma flotante: Double, float
  - Datos de tipo carácter: char
  - Datos de tipo lógico: boolean

Universidad Zaragoza

Clasificación de los tipos de datos



Universidad Zaragoza

Tipos de datos primitivos

Tipo	Tamaño	Rango	Notas
int	4 bytes	–2.147.483.648 a 2.147.483.647	
short	2 bytes	–32.768 a 32.767	
long	8 bytes	–9.223.372.036.854.775.808 a 9.223.372.036.,854.775.807	Valores acaban en L (p.e. 1L).
byte	1 byte	–128 a 127	El rango no es 0 ... 255.
float	4 bytes	Aproximadamente ±3.40282347E+38F (6–7 dígitos decimales significativos)	Valores acaban en F (p.e. 0.5F)
double	8 bytes	Aproximadamente ±1.79769313486231570E+308 (15 dígitos decimales significativos)	
char	2 bytes	\u0000 to \uFFFF	
boolean		true o false	

Universidad Zaragoza

Tipo entero (Literales/Representación externa)

Literal

Especificación de un valor concreto de un tipo de dato

- Los literales enteros pueden expresarse
  - En decimal (base 10); 255
  - En octal (base 8): 0377 ($3.8^2 + 3.8^1 + 7 = 255$)
  - En hexadecimal (base 16)=0xFF ($15.16^1 + 15 = 255$)
  - Los literales enteros son de tipo int por defecto (entre -2^{31} y $2^{31} -1$)
- Un literal entero es de tipo long si va acompañado del sufijo l o L (preferiblemente L)

  1234567890L

Universidad Zaragoza

Representación interna enteros

- Enteros en complemento a 2
  - Conversión rápida

    negar todos los dígitos y después sumar un 1 al resultado,

    100001 ---> 011110 --> 011111
  - ¿Para que sirve?
  - Facilita las operaciones matemáticas con números binario:

La resta de dos números en binario se obtiene sumando al minuendo el complemento a dos del sustraendo.

```
0 1 1 →  3
1 1 1 → -1
---------
0 1 0 →  2
```

Complemento a dos	Decimal
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6
1001	-7
1000	-8

Universidad Zaragoza
1542

Operaciones enteros

Operador	Operación
+	Suma
-	Resta
*	Producto
/	División
%	Resto

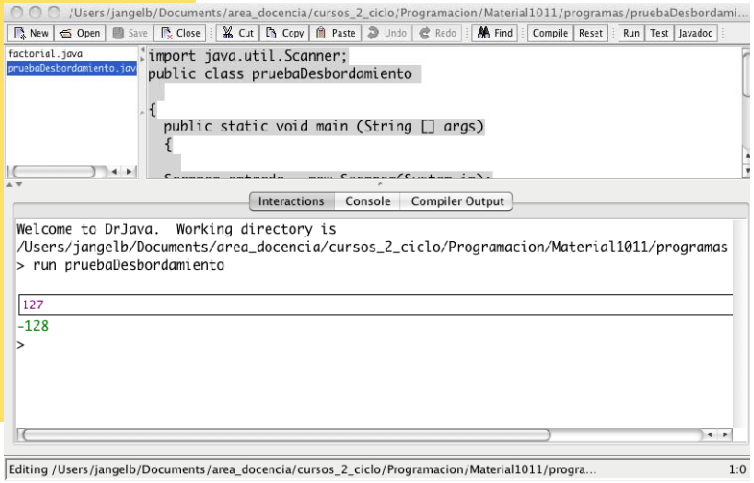
Si los operandos son enteros el resultado es entero

Universidad Zaragoza
1542

Desbordamiento

- Si sobrepasamos el valor máximo que se puede representar con un tipo entero, **no nos avisa**: en la ejecución obtendremos un valor incorrecto:

```
import java.util.Scanner;
public class pruebaDesbordamiento

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

  Scanner entrada = new Scanner(System.in);

  byte numero= entrada.nextByte();

  numero++;
  System.out.println(numero);
  }
}
```

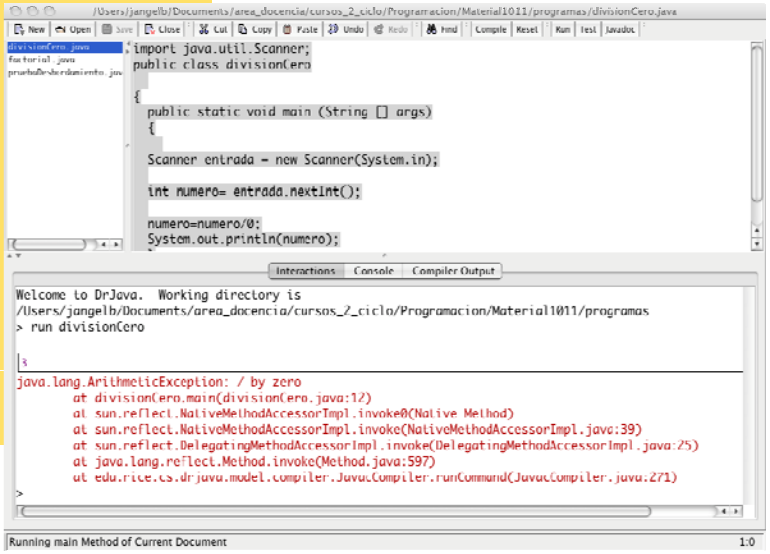


Universidad Zaragoza
1542

División por cero

- División por cero produce error en tiempo de ejecución (Se aborta el programa)!!!

```
import java.util.Scanner;
public class divisionCero

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

  Scanner entrada = new Scanner(System.in);

  int numero= entrada.nextInt();

  numero=numero/0;
  System.out.println(numero);
  }
```



Universidad Zaragoza
1542

Literales Reales

- Representación con punto decimal

  123.45 0.0 .001

- En notación científica

  123e45 123E+45 1E-6

- Por defecto los literales reales son de tipo double

- Para representar literales de tipo float, utilizaremos el sufijo f o F

  123.45F 0.0F .001f

Universidad Zaragoza

Representación interna reales

- **IEEE 754, estándar de la IEEE para aritmética de coma flotante**
  - **Precisión simple 32 bits valor= signo*exponente**

signo	Exponente	mantisa
1 bit	8 bits	23 bits

  - **Precisión simple 64 bits valor= signo*exponente**

signo	Exponente	mantisa
1 bit	11 bits	52 bits

No se pueden representar todos los reales dentro del dominio de valores
Operaciones sujetas a error de redondeo

Universidad Zaragoza
1542

Operaciones reales

- Las operaciones aritméticas en coma flotante no generan excepciones aunque sean ilegales:
  - Cuando el resultado está fuera de rango devuelve +Infinity, -Infinity
    - 1.0/0.0 → + Infinity
    - -1.0/0.0 → - Infinity
  - Cuando es indeterminado devuelve NaN (Not a Number)
    - 0.0/0.0 → NaN

Universidad Zaragoza

Operaciones con reales

Si algún operando es float o double, el resultado es real

```
> System.out.println(1.0/2);
0.5
> System.out.println(1/2);
0
> System.out.println(7/3.0f);
2.3333333
> System.out.println(7%3);
1
> System.out.println(4.3%2.1);
0.09999999999999964
>
```

Universidad Zaragoza
1542

Expresiones aritméticas

- Combinación de operadores y literales para formar expresiones complejas

(4+ 5* x) / 32 + (5 * (z – 7) + (a + b + c))/42

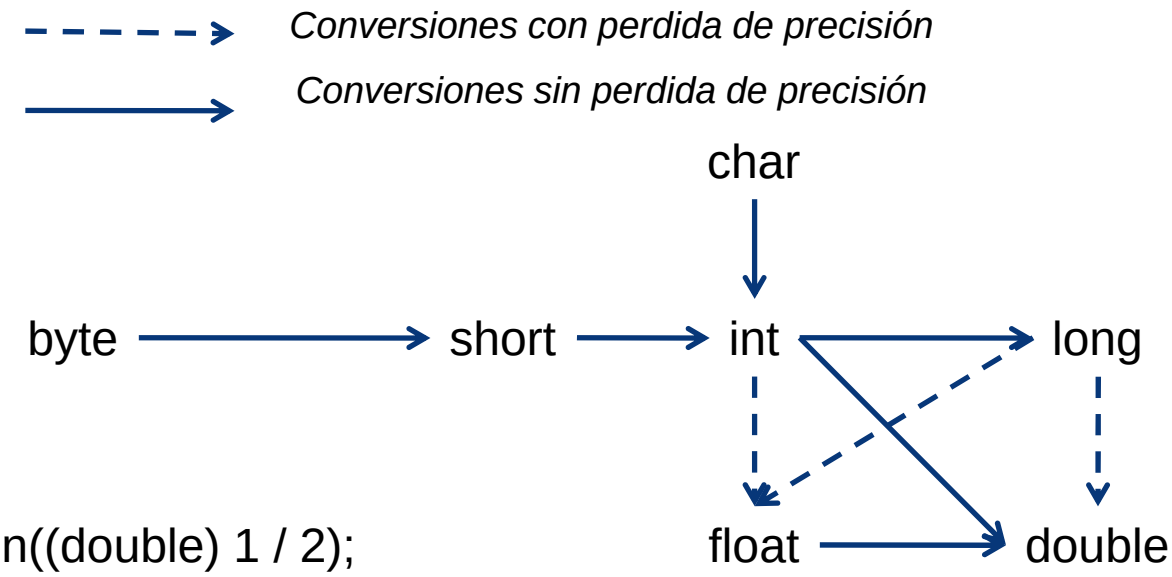
Universidad Zaragoza

Orden de precedencia en los operadores:

1. ++ --
2. * / %
3. + -
4. < > <= >=
5. == !=
6. &
7. |
8. = *= /= %= += -=

Universidad Zaragoza

Conversión entre tipos (Casting)




```
> System.out.println((double) 1 / 2);
0.5
> System.out.println(1 / (double) 2);
0.5
> System.out.println((double) (1/2));
0.0
```

Universidad Zaragoza
1542

caracteres

- Codificación ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Universidad Zaragoza
1542

caracteres

- Codificación UNICODE ISO/IEC10646 (16 bits 65536 símbolos)

Zona	*Códigos*		*Símbolos codificados*	*Nº de caracteres*
A	0000	0000 00FF	Latin-1	256
			otros alfabetos	7.936
		2000	Símbolos generales y caracteres fonéticos chinos, japoneses y coreanos	8.192
I	4000		Ideogramas	24.576
O	A000		Pendiente de asignación	16.384
R	E000 FFFF		Caracteres locales y propios de los usuarios. Compatibilidad con otros códigos	8.192

Universidad Zaragoza
1542

Literales de tipo char

- Entre comillas

  'a' 'b' '1' '*'

  Códigos unicode en hexadecimal

  '\u000a' avance de línea

  '\u000d' retorno de carro

  Secuencias de escape

Secuencia escape	descripción
\t	Tabulador
\n	Avance de línea
\r	Retorno de carro
\b	retroceso
\'	Comillas simples
\"	Comillas dobles
\\	Barra invertida

Universidad Zaragoza
1542

Booleanos

- Literales

  true **false**

- Operadores relacionales (Comparan números o caracteres) y devuelven un valor booleano

operador	significado
==	igual
!=	distinto
>	menos
>	mayor
<=	menor o igual
>=	mayor o igual

Universidad Zaragoza
1542

Expresiones boolenas

Operador	Significado
!	NOT
&&	AND
ll	OR
^	XOR

AND	T	F
T	T	F
F	F	F

- NOT cambia el valor
- And devuelve true si los dos operandos son true.
- Or decuelve true si alguno es true.
- XOR devuelve true si los dos operandos son diferentes.

OR	T	F
T	T	T
F	T	F

Universidad Zaragoza
1542

Tipos estructurados

Dependiendo del acceso a las partes
- definido mediante agregación (clase)
- definido mediante indexación (tabla o vector)
- definido como secuencia (fichero secuencial)

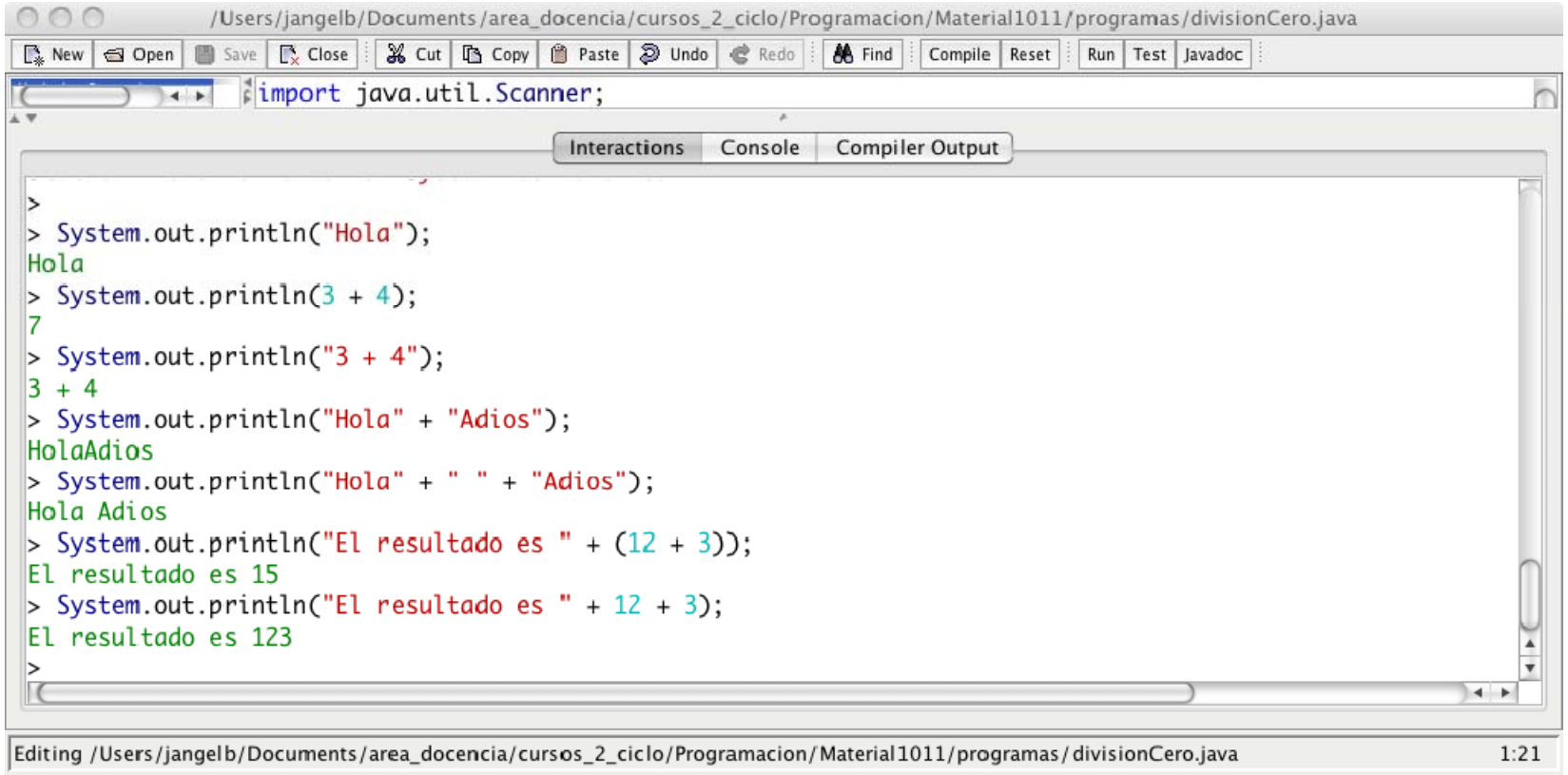
En Java las clases permiten definir nuevos tipos de datos

declaración obligatoria de todos los DATOS y TIPOS de DATOS

Universidad Zaragoza
1542

String

- Secuencias de caracteres



Universidad Zaragoza

Datos constantes

- su valor no puede ser modificado por el algoritmo

atributos ⇩ invariantes

- tipo de dato → implícito (sintaxis)
- valor → explícito
- ámbito
- nombre → > legibilidad, modificabilidad y reusabilidad

```
> final double PI = 3.1416;
> PI=2;
Static Error: Variable 'PI' cannot be modified
>
```

Universidad Zaragoza

Datos variables

- su valor puede ser modificado por el algoritmo

atributos invariantes
- tipo de dato → explícito (declaración obligatoria)
- nombre
- ámbito

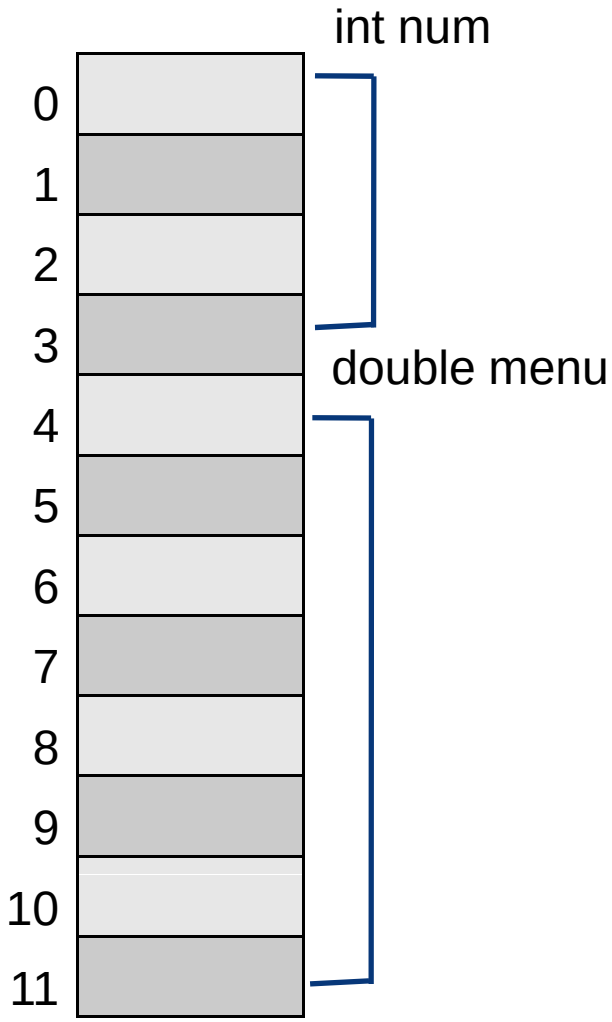
atributo variable
- valor ⟹ operador asignación =

- el valor de una variable sólo declarada es indeterminado

Universidad Zaragoza

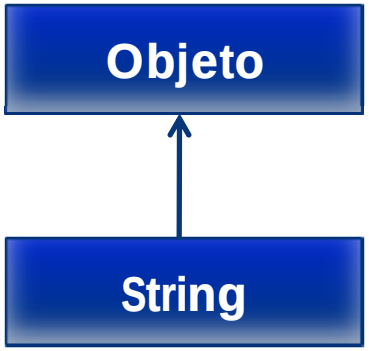
Variables en memoria (datos simples)

```
import java.util.Scanner;
public class factura

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

  Scanner entrada = new Scanner(System.in);

  System.out.rintln("numero de comensales:");
  int num= entrada.nextInt();
  System.out.println("precio menus"):
  int menu= entrada.nextDouble();
  double propina = num*menu*0.2;
  System.out.println("Total:"+(num* menu)+ propina);
}
}
```



Universidad Zaragoza

Variables en memoria (objetos)

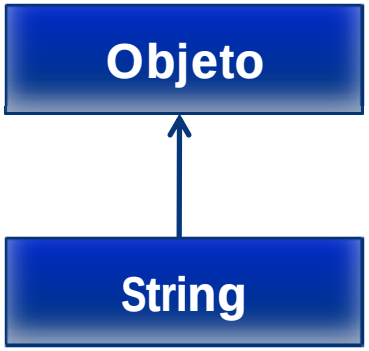


- Las variables objeto reservan memoria para un puntero

```
> String test;
> System.out.println(test)
null
> test = "Hola";
> System.out.println(test);
Hola
> test = new String("Adios");
> System.out.println(test);
Adios
>
```

Universidad Zaragoza

Variables en memoria (objetos)

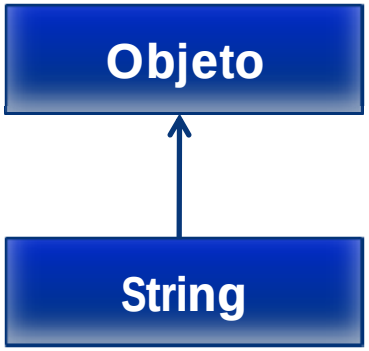


- Las variables objeto reservan memoria para un puntero

```
> String test;
> System.out.println(test)
null
> test = "Hola";
> System.out.println(test);
Hola
> test = new String("Adios");
> System.out.println(test);
Adios
>
```

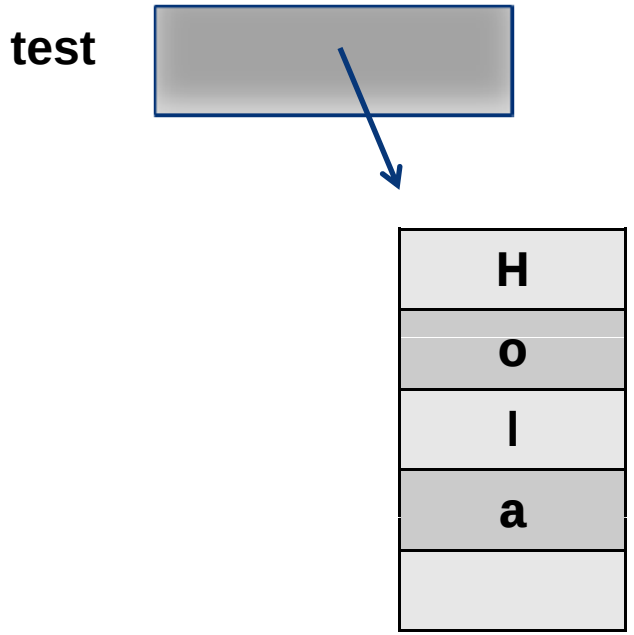
Universidad Zaragoza

Variables en memoria (objetos)



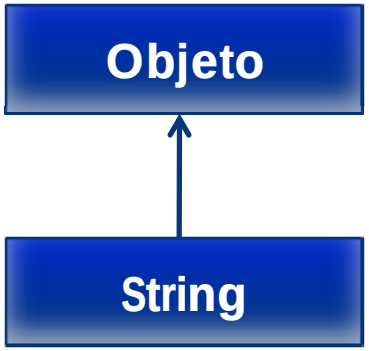
- Las variables objeto reservan memoria para un puntero

```
> String test;
> System.out.println(test)
null
> test = "Hola";
> System.out.println(test);
Hola
> test = new String("Adios");
> System.out.println(test);
Adios
>
```



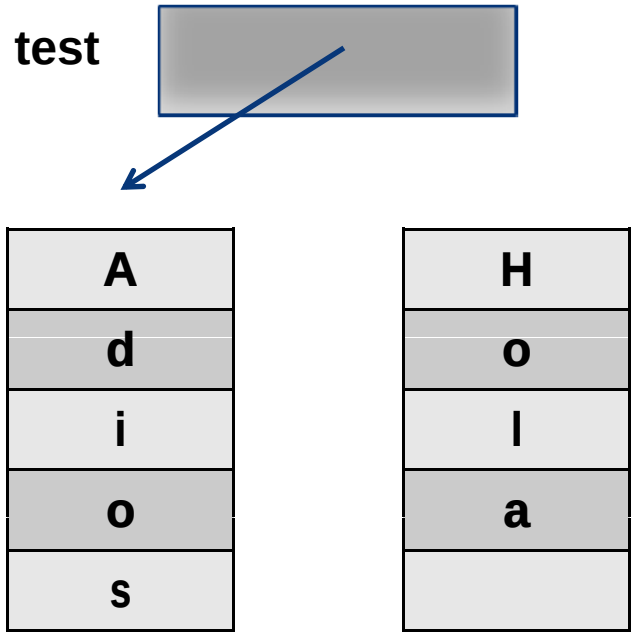
Universidad Zaragoza

Variables en memoria (objetos)



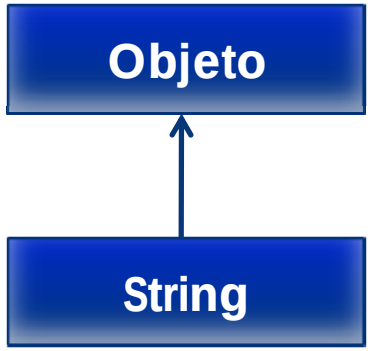
- Las variables objeto reservan memoria para un puntero

```
> String test;
> System.out.println(test)
null
> test = "Hola";
> System.out.println(test);
Hola
> test = new String("Adios");
> System.out.println(test);
Adios
>
```



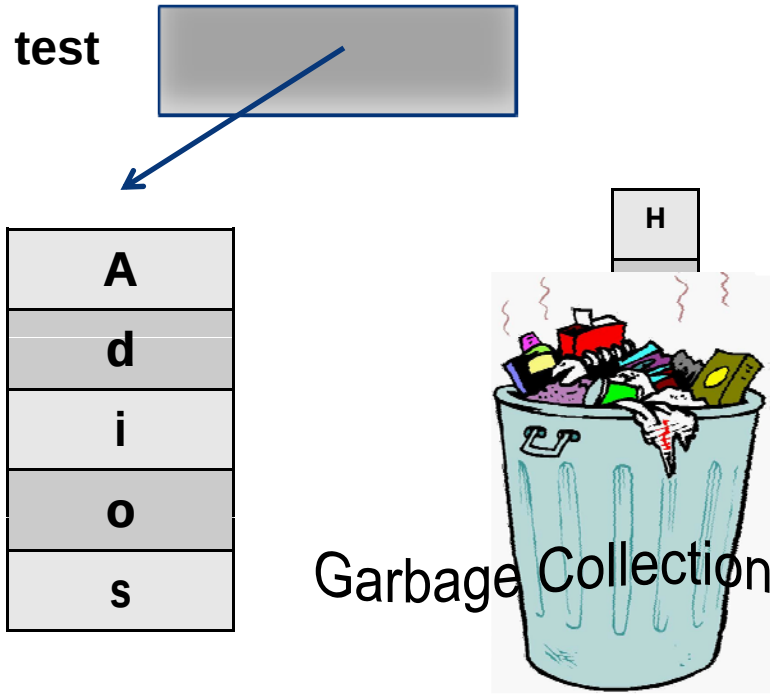
Universidad Zaragoza

Variables en memoria (objetos)



- Las variables objeto reservan memoria para un puntero

```
> String test;
> System.out.println(test)
null
> test = "Hola";
> System.out.println(test);
Hola
> test = new String("Adios");
> System.out.println(test);
Adios
>
```



Universidad Zaragoza
1542