Introducción a Java

Definiciones

Un **objeto** es una estructura que encapsula el estado y comportamiento de un conjunto de datos

El **estado** de un objeto está definido por los valores guardados en los **atributos** del objeto

Un **atributo** es un espacio de memoria en donde se guarda información

Definiciones

Un **objeto** es una estructura que encapsula el estado y comportamiento de un conjunto de datos

El **comportamiento** está definido por los **métodos**

Un **método** es una secuencia de instrucciones guardadas en la memoria

Definiciones

Una **clase** es un modelo o plantilla para crear objetos

A un objeto de una clase C se le llama **instancia de C**

A la entidad que guarda la dirección en la memoria donde se guarda un objeto, se le llama **referencia**

Clases en Java

Sintaxis para declarar una clase:

```
class Ejemplo{

}
```

Java Code Conventions [JCC]

Las convenciones de código son recomendaciones de cómo debe escribirse un archivo fuente

Mejoran la legibiliad del código para facilitar el mantenimiento del software

Cubren la organización del archivo, indentación, comentarios, declaraciones, convenciones de nombrado, practicasde programación... entre muchas otras cosas

Clases en Java

[JCC] Los nombres de clase comienzan con mayúsula.

Si el nombre se forma con mas de una palabra, se escriben sin espacios y cada palabra comenzando con mayúscula

```
class Ejemplo{
 ...
}
```

```
class HolaMundo{
 ...
}
```

```
class UnNombreDeClaseMuyLargo{
 ...
}
```

Clases en Java

[JCC] La delcaración de una clase se divide en el siguiente orden

```
class Ejemplo{

    \\ Declaracion de atributos


    \\ Declaracion de métodos

}
```

Clases en Java

Los archivos de **código fuente** contienen una o más clases

El archivo se debe llamar igual que la clase principal con la extensión **.java**

Ejemplo:

```
class Ejemplo{

}
```

El archivo de código fuente debe llamarse:

Ejemplo.java

Compilación

El archivo fuente está escrito en un **lenguaje de alto nivel** que puede ser leido y entendido facilmente por humanos

Una computadora funciona en lenguaje **binario**, por lo que debemos **compilar (**traducir) nuestro archivo fuente a lenguaje binario para que sea interpretado por esta

Compilación en java

Para compilar el archivo fuente, se usa el comando **javac**

```
$ javac Ejemplo.java
```

El compilador escribirá un nuevo archivo en **bitecode** con el mismo nombre pero con extensión **.class**

El **bitecode** es un lenguaje binario interpretado por la máquina virtual de Java (JVM)

Variables

Una **variable** es un identificador de una localidad de memoria usada para guardar y procesar datos

En java se tiene que delcarar **tipo de dato** de una variable antes de usarla

Tipo de dato

Le indica a una computadora como interpretar la información guardada en la memoria, los valores que puede tomar y las operaciones que puede realizar

En java una referencia a un objeto es un tipo definido por la clase del objeto

Además de referencias a objetos, cuenta con tipos de datos predefinidos, a estos se les llama **tipos de datos primitivos**

	Tamaño (bits)	Mínimo	Máximo
byte	8	-128	127
short	16	-32,768	32,767
int	32	-2^{31}	$2^{31}-1$
long	64	-2^{63}	$2^{63}-1$
float	32	-	-
double	64	-	-
boolean	-	-	-
char	16	0	65,535

Variables

Declaración

```
T variable;
```

Declaración múltiple

```
T variable, variable2, variable3;
```

Variables

Declaración

```
T variable;
```

Declaración múltiple

```
T variable, variable2, variable3;
```

[JCC] ¡No se recomienda!

Variables

[JCC] Comienza con minúscula, si se requiere una segunda palabra, esta comenzará con mayúscula. Puede contener números.

i j k ultimoAcceso automovil

c e d comprasUltimoAño

m n estado1 estado2

Variables de un objeto (atributos)

Variables de instancia Una localidad distinta para cada instancia

```
T variable;
```

Variables de clase La misma localidad de memoria para todas las instancias

```
static T variable;
```

Otras variables

Variable local

```
T variable;
```

Parametro

```
metodo(variable, variable2, variable3)
```

Tipos de variable

[JCC] La delcaración de atributos debe ser en el siguiente orden

```
class Ejemplo{

    \\ Declaracion de variables de clase
    \\ Declaracion de instancia

    \\ Declaracion de métodos

}
```

Literales

Son valores constantes que no se declaran previamente

Como su nombre lo indica, son la representación **literal** de un valor

Sintaxis para literales

Literales de enteros de los tipos primitivos byte, short e int

Decimal	-234	234
Hexadecimal	0xa3f4	
Binaria	0b1010	

Sintaxis para literales

Literales de enteros del tipo primitivo long

Decimal	-234L	234L
Hexadecimal	0xa3f4L	
Binaria	0b1010L	

Sintaxis para literales

Para separar digitos y facilitar la lectura del código, se usa un guion bajo '_'

Decimal	-234_234L
Hexadecimal	0xa3f4_212
Binaria	0b1010_0101

Sintaxis para literales

Literales de punto flotante del tipo primitivo double

-23423.4 0.577215

Notación científica

1.123e5

Sintaxis para literales

Literales de punto flotante de tipo primitivo float

-23423.4f 0.577215f

Notación científica

1.123e5f

Sintaxis para literales

Literales del tipo de dato primitivo char

'a' '\n' '\u0012'

Sintaxis para literales

Secuencias de escape en Java

\b	backspace	\r	carriage return
\t	tab	\"	double quote
\n	line feed	\'	single quote
\f	form feed	\\	backslash

Sintaxis para literales

Literales instancias de la calse String

”Esto es una cadena”

”Esto es una cadena\n”

””

Sintaxis para literales

La literal null es un valor de referencia que apunta a nada, internamente es una referencia a la dirección de memoria 0

Se utiliza para indicar que una variable no hace referencia a ningun objeto

Operador asignación

Cambia el valor de una variable por uno nuevo

```
variable = 1234;

variable2 = variable;
```

Declaración y asignación

Declaración y asignación

```
int variable = 1234;
```

Declaración y asignación múltiple

```
int variable = 1234, variable2 = 1234;
```

Declaración y asignación

Declaración y asignación

```
int variable = 1234;
```

Declaración y asignación múltiple

```
int variable = 1234, variable2 = 1234;
```

[JCC] ¡No se recomienda!

Declaración y asignación

[JCC] Se considera una buena práctica inicializar las variables inmediatamente después de declararla a menos que se requiera algún cálculo para su valor inicial.

[JCC] Se recomienda incluir un comentario al final de la una declaración que ayude a entender su uso.

```
int max = 1234; //Máximo valor posible
int size = 1234;  //Tamaño del arreglo
```

Constantes

Declaración

```
final int CONST;
```

Declaración y asignación

```
final int CONST = 1234;
```

Constantes

[JCC] Se escriben con mayúsculas, si se requiere una segunda palabra se usa _.

CONST FECHA_NACIMIENTO

G ERROR_DATOS

PI IVA