Preguntas de análisis

A. Según el siguiente código, indique qué se imprime por consola y explique el porqué de cada línea donde se imprime.

> Cuando no existe el método con el mismo tipo de dato, se asocia al método que tenga el parámetro del tipo inmediatamente superior

- al llamar a la función con un char la función que se llama es la que recibe un int e imprime el valor asci del char
- al llamar a la función con un short la función que se llama es la que recibe un int e imprime el valor del short (casting implícito)
- al llamar a la función con un byte la función que se llama es la que recibe un int e imprime el valor del byte (casting implícito)
- al llamar a la función con un long la función que se llama es la que recibe un double e imprime el valor del long (casting implícito)
- al llamar a la función con un integer la función que se llama es la que recibe un int e imprime el valor del integer
- al llamar a la función con un double la función que se llama es la que recibe un double e imprime el valor del double
- al llamar a la función con un float la función que se llama es la que recibe un double e imprime el valor del float (casting implícito)

B. Realice los siguientes cambios, teniendo siempre como referencia el código inicial. Explique cómo y por qué cambia lo que se imprime por pantalla.

para entender el por qué el impacto de cada cambio vamos a usar esta tabla

Tipos de datos

En java existen algunos tipos de datos, llamados primitivos o básicos que almacenan valores de un valor especifico

Tipo	Representación / Valor	Tamaño (bits)	Valor mínimo	Valor máximo	Valor por defecto
boolean	true o false	1	N.A	N.A	false
char	Carácter Unicode	16	\u0000	\uffff	\u0000
byte	Entero	8	-128	127	0
short	Entero	16	-32768	32767	0
int	Entero	32	-2147483648	2147483647	0
long	Entero	64	-9223372036854775808	9223372036854775807	0
float	Coma flotante, precisión simple	32	-3.402823e38	3.402823e38	0.0

FACULTAD DE MINAS
Sede Medellín

UNIVERSIDAD NACIONAL DE COLOMBIA

PROGRAMACIÒN ORIENTADA A OBJETOS

double	Coma flotante, precisión doble	64	-1.79769313486232e308	1.79769313486232e308	0.0

• Active la función que recibe un short. • Active la función que recibe un float. • Comente la función que recibe un double y active la que recibe un float. • Comente todas las funciones, excepto la que recibe un double.

El único caso en el que tendríamos errores de compilacion es en el caso de que se comente la función que recibe un double. En este caso como la función es llamada coun un double deberíamos usar Narrowing Casting (cating manual/explicito)

De acuerdo con JLS-5.1.2. Widening Primitive Conversion:

5.1.3. Narrowing Primitive Conversion

22 specific conversions on primitive types are called the *narrowing primitive conversions*:

- `short` to `byte` or `char`
- `char` to `byte` or `short`
- `int` to `byte`, `short`, or `char`
- `long` to `byte`, `short`, `char`, or `int`
- `float` to `byte`, `short`, `char`, `int`, or `long`
- `double` to `byte`, `short`, `char`, `int`, `long`, or `float`

A narrowing primitive conversion may lose information about the overall magnitude of a numeric value and may also lose precision and range.

A narrowing primitive conversion from `double` to `float` is governed by the IEEE 754 rounding rules (§4.2.4). This conversion can lose precision, but also lose range, resulting in a `float` zero from a nonzero `double` and a `float` infinity from a finite `double`. A `double` NaN is converted to a `float` NaN and a `double` infinity is converted to the same-signed `float` infinity.

A narrowing conversion of a signed integer to an integral type T simply discards all but the *n* lowest order bits, where *n* is the number of bits used to represent type T. In addition to a possible loss of information about the magnitude of the numeric value, this may cause the sign of the resulting value to differ from the sign of the input value.

A narrowing conversion of a `char` to an integral type T likewise simply discards all but the *n* lowest order bits, where *n* is the number of bits used to represent type T. In addition to a possible loss of information about the magnitude of the numeric value, this may cause the resulting value to be a negative number, even though chars represent 16-bit unsigned integer values.

A narrowing conversion of a floating-point number to an integral type T takes two steps:

1. In the first step, the floating-point number is converted either to a `long`, if T is `long`, or to an `int`, if T is `byte`, `short`, `char`, or `int`, as follows:
   - If the floating-point number is NaN (§4.2.3), the result of the first step of the conversion is an `int` or `long` `0`.
   - Otherwise, if the floating-point number is not an infinity, the floating-point value is rounded to an integer value V, rounding toward zero using IEEE 754 round-toward-zero mode (§4.2.3). Then there are two cases:
     a. If T is `long`, and this integer value can be represented as a `long`, then the result of the first step is the `long` value V.
     b. Otherwise, if this integer value can be represented as an `int`, then the result of the first step is the `int` value V.
   - Otherwise, one of the following two cases must be true:
     a. The value must be too small (a negative value of large magnitude or negative infinity), and the result of the first step is the smallest representable value of type `int` or `long`.
     b. The value must be too large (a positive value of large magnitude or positive infinity), and the result of the first step is the largest representable value of type `int` or `long`.

W3Schools nos dice que