Usar tipos de datos primitivos en código Java

• Especificar literales para los tipos primitivos y para las cadenas

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

La palabra clave nueva no se usa cuando se inicializa

una variable de un tipo primitivo

Por el contrario, se deberá asignar un valor literal a cada variable en el momento de la inicialización, un literal puede ser cualquier número, texto u otra información que representa un valor

• Ejemplos de declaración de una variable y asignación del valor literal:

boolean result = true;

char capitalC = 'C';

byte b = 100;

short s = 10000;

int i = 100000;

Los valores de d1 y d2 son iguales

La inicialización de d2 muestra cómo se puede usar la notación científica para establecer el valor, al total y ss\_num se les asigna el mismo valor

La inicialización de ss num muestra que se pueden

usar los subrayados para separar a los números para

legibilidad

long total=999999999;

long ss\_num = 999\_99\_9999;

double d1 = 123.4;

double d2 = 1.234e2;

float f1 = 123.4f;

• Demostrar cómo inicializar variables

public static void main (String [] args) {

int i;

i = 0;

Sistema.fuera.println"yo soy " + I);

}

• Describir las reglas de alcance de un método

−No use una palabra clave o una palabra reservada de Java

−No use un espacio en el nombre de la variable

−Use una combinación de letras o una combinación de letras y números

−No puede comenzar por un número

−Los únicos símbolos permitidos son el subrayado ( \_ ) y el signo pesos ($)

• Identificar la necesidad de conversión de tipo en una expresión

El uso de literales binarios para representar valores de comparaciones y cálculos es sustancialmente más rápido que el uso de valores del tipo de datos real, los procesadores de alto rendimiento modernos generalmente realizan cálculos en enteros tan rápido como con los literales binarios, por eso, ¿por qué usar literales?, se recomienda usar literales para energía y rendimiento general, porque utilizan menos recursos.

• Aplicar la conversión en código Java

byte aByte = (byte)0b00100001;

short aShort = (short)0b1010\_0001\_0100\_0101;

int anInt1=0b1010\_0001\_0100\_0101\_1010\_0001\_0100\_0101;

int anInt2=0b101;

int anInt3=0B101;

• Usar operadores aritméticos

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

• Usar el operador de asignación

int x = 4;

int y = 5;

int z = 10;

int total = 12;

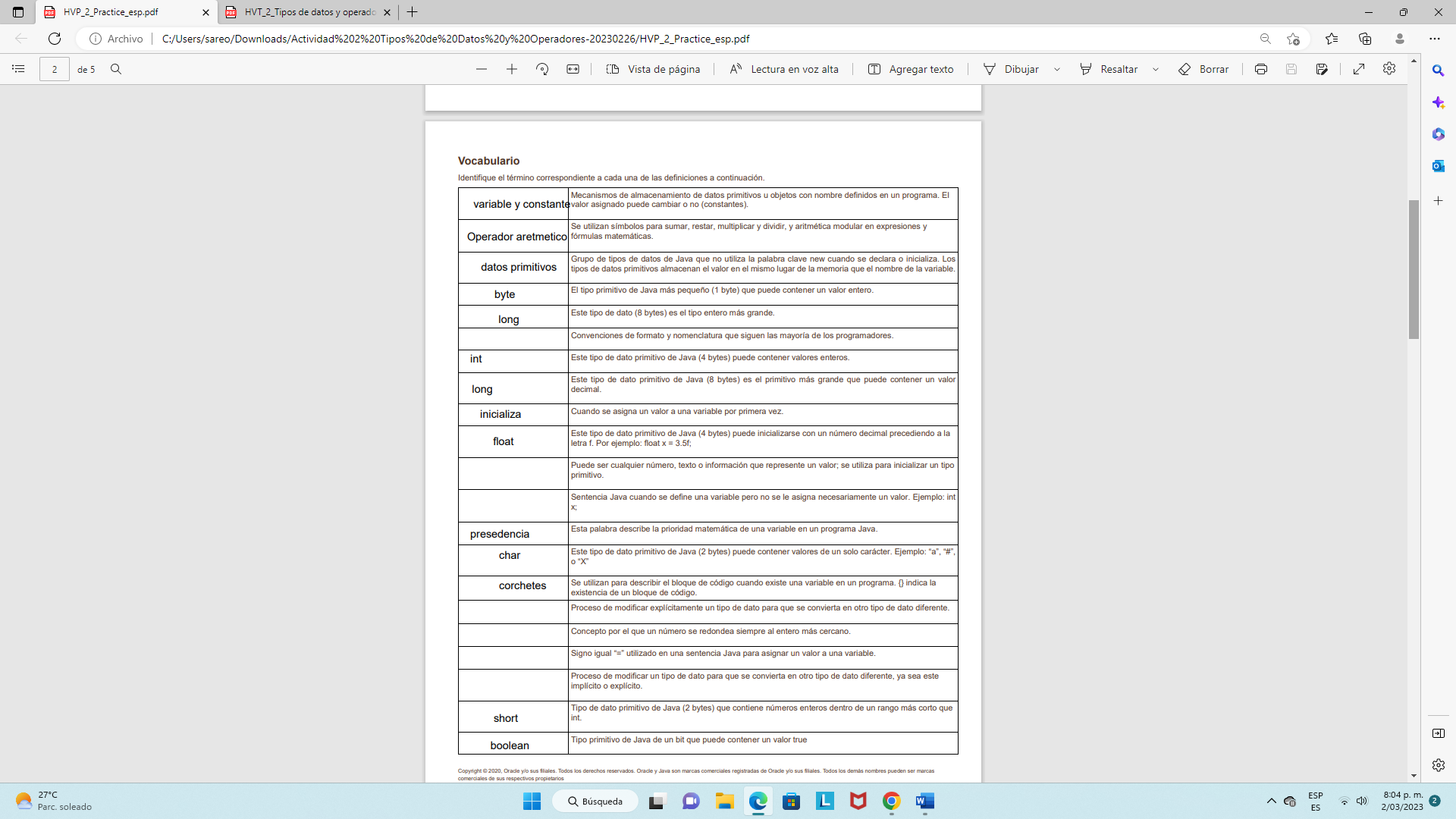
• Usar un método de la clase Math

Math.pow(x, y):

• Acceder a un método de la clase Math de la API de Java

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente con confianza media



Truncamiento

Operador de asignacion

Conversión de tipos

Comprensión de tipos y conversiones

double

double

IDC

4. Escriba true o false en los espacios en blanco del programa a continuación para mostrar el valor de la variable booleana

true\_false mientras se ejecuta el programa.

int i=5;

int j=6;

boolean true\_false;

true\_false=(j<5); True

true\_false=(j>3); True

true\_false=(j<i); False

true\_false=(i<5); False

true\_false=(j<=5); True

true\_false=(6<6); False

true\_false=(i!=j); False

true\_false=(i==j || i<50); True

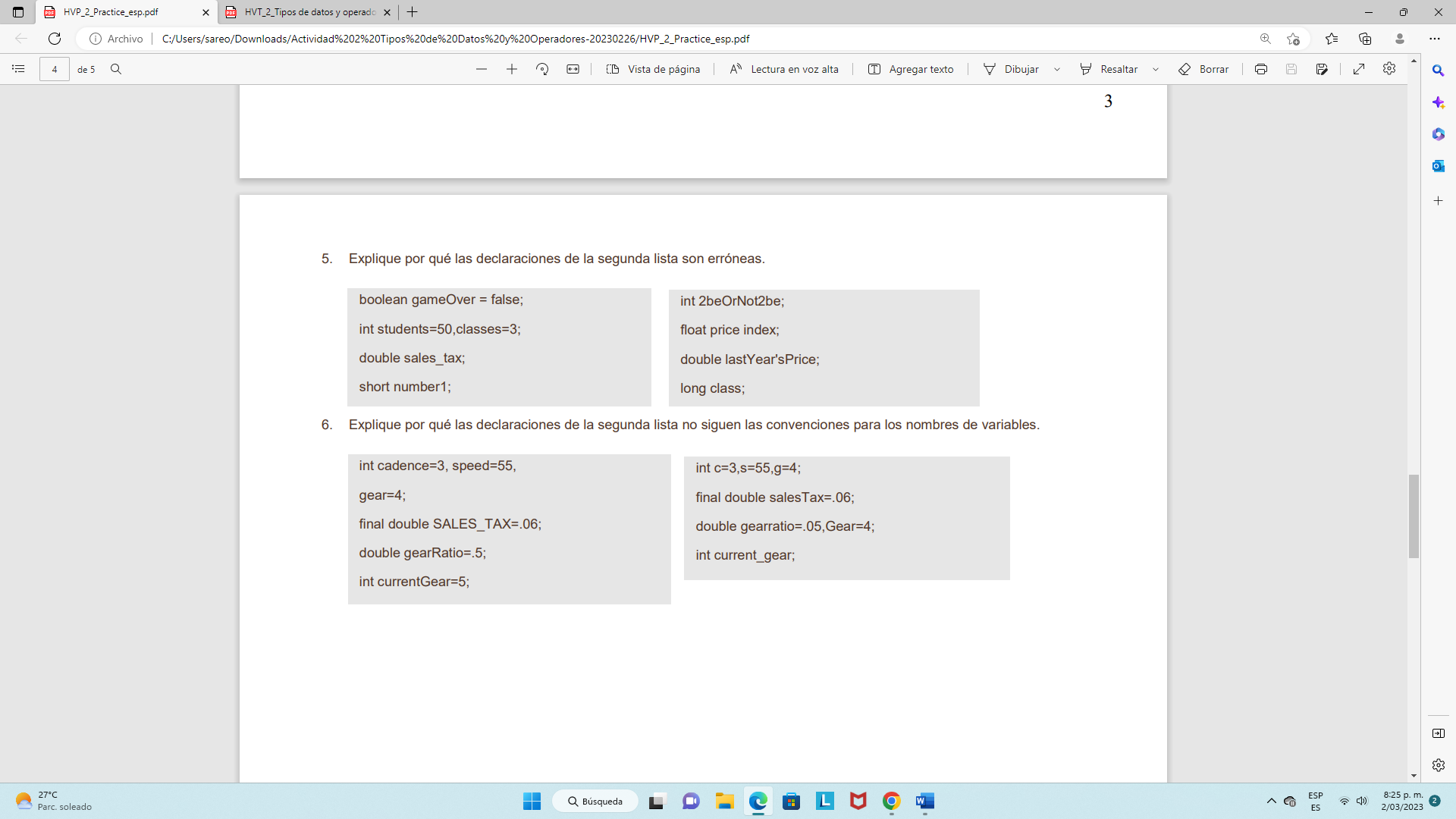
true\_false=(i==j && i<50); True

true\_false=(i>j || true\_false && j>=4); True

true\_false=(!(i<2 && j==5)); False

true\_false=!true\_false; False

5. Explique por qué las declaraciones de la segunda lista son erróneas.



6. Explique por qué las declaraciones de la segunda lista no siguen las convenciones para los nombres de variables.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente