Segundo Ejercicio Práctico

https://ocw.uc3m.es/pluginfile.php/2379/mod_page/content/12/practica_java_basics.pdf

Este ejercicio práctico consta de tres partes:

- La primera parte es una introducción al entorno de programación con Eclipse. No se entregará ningún material para esta parte.
- La segunda parte tiene como objetivo introducir la programación en Java. Esta parte incluye una serie de ejercicios básicos que los estudiantes deberán desarrollar, así como un conjunto de preguntas asociadas a cada ejercicio. No se entregará el código de estos ejercicios.
- **La tercera parte incluye ejercicios más complejos**. El código de estos ejercicios deberá copiarse a un archivo en formato **PDF** y enviarse antes del **08 de mayo de 2026** al correo electrónico del profesor (`wcruzy@unp.edu.pe`).

3. Tercera parte: Ejercicios avanzados

Ejercicio 1. División

Crea un programa que lea dos números enteros del teclado y los almacene en dos variables enteras, `valor_a` y `valor_b`. Si `valor_a` es divisible por `valor_b`, el programa debe imprimir "*[a] es divisible por [b]*" (donde [a] y [b] deben ser los valores reales de las variables `valor_a` y `valor_b`). Si `valor_a` no es divisible por `valor_b`, el programa debe imprimir "*[a] no es divisible por [b], el resto es [r]*" (donde *[r]* es el valor real del resto de la división a/b).

Ejercicio 2. Piedra, papel o tijera

Crea un programa para calcular quién gana una partida de piedra, papel o tijera. El programa debe leer dos cadenas de texto del teclado, cada una representando la selección de un jugador (`selección_1`, `selección_2`). El programa debe imprimir en pantalla qué jugador es el ganador y los gestos de los jugadores. Por ejemplo, si `selection_1` es "*Piedra*" y `selection_2` es "*Papel*", el programa debe imprimir "*Gana el jugador 2. Papel derrota a Piedra*". Si ambos jugadores hacen el mismo gesto, el programa debe imprimir "*¡Empate!*".

En Eclipse, selecciona la opción **Mostrar números de línea** (**Preferencias** ->**Editores de texto** –>**Mostrar números de línea**). Ejecuta el programa paso a paso con el depurador, usando "*Papel*" y "*Piedra*" como entradas, y anota la secuencia de números de línea resultante. Incluye esta secuencia como comentario al final del programa.

Ejercicio 3. Quiniela de fútbol

Crea un programa que declare una matriz de caracteres bidimensional de 15 filas x 3 columnas. Usa bucles anidados para asignar un '`1`' a los elementos de la primera columna, una '`X`' a los de la segunda y un '`2`' a los de la última. Utiliza bucles anidados para imprimir el contenido del array en pantalla. Cambia el valor de cualquier celda del array (establece el valor '`?`') e imprímelo de nuevo.

Ejercicio 4. Exponenciación

Crea un programa que imprima las potencias de un número entero leído desde el teclado. Por ejemplo, si el valor leído es 3, el programa debe imprimir la siguiente salida.

NOTA: utiliza el método `Math.pow(base, exponente)` para realizar el cálculo.

```
3^0 = 1
3^1 = 3
3^2 = 9
3^3 = 27
...
3^10 = 59049
```

Ejercicio 5. Día

Crea un programa para calcular el período del día correspondiente a una hora leída desde el teclado. El programa debe comprobar si la hora recibida está entre 0 y 23, e imprimir (mediante una instrucción `switch`):

mañana cuando la hora esté entre 6 y 12

tarde cuando la hora esté entre 13 y 21

noche cuando la hora esté entre 22 y 2

madrugada cuando la hora esté entre 3 y 5

Ejercicio 6. Damas

Crea un programa que defina un arreglo para representar un tablero de damas de 10x10 con la posición inicial de todas las piezas. Imprime el tablero en la pantalla. Implementa el primer movimiento de una de las piezas blancas situada en una de las posiciones más avanzadas. La pieza a mover debe leerse desde el teclado (un valor entero en 1, ..., 5). La posición de destino también debe leerse desde el teclado (un valor entero en 1, 2). donde 1 representa la posición a la izquierda y 2 la posición a la derecha. Imprime el tablero resultante del movimiento. Si la combinación de la pieza y la posición no es válida, imprime "`¡Movimiento no válido!`".

El tablero debe representarse de la siguiente manera:

- Casillas vacías: símbolo '`@`' para casillas claras vacías; símbolo '`#`' para casillas oscuras vacías.
- Casillas con piezas: '`d`' para casillas con una pieza blanca; '`D`' para casillas con una pieza negra.

Fuente: http://en.wikipedia.org/wiki/Draughts

Ejercicio 7. Factorización

Crea un programa con un método `main` que, dado un valor entero como parámetro, imprima el resultado de su factorización prima.

Ejercicio 8. Segundo anterior

Crea un programa con un método `main` que, dado un valor de tiempo, imprima el tiempo correspondiente al segundo anterior. Los valores de tiempo deben pasarse como argumentos al método `main` (el primero corresponde a la hora, el segundo a los minutos y el tercero a los segundos).

Ejercicio 9. Inclusión de conjuntos

Crea un programa con un método `main` que reciba varios valores numéricos como argumentos. Los valores deben almacenarse en dos arreglos llamados `arreglo1` y `arreglo2`; la primera mitad en `arreglo1` y la otra mitad en `arreglo2`. Si el número de valores es impar, se descarta el último. El programa debe comprobar si todos los valores de `arreglo2` están incluidos en `arreglo1`. Si se cumple esta condición, el programa debe imprimir "`el conjunto 2 está incluido en el conjunto 1`"; de lo contrario, debe imprimir "`el conjunto 2 no está incluido en el conjunto 1`".

NOTA: Se permiten valores repetidos en ambos arreglos.

Ejercicio 10. Blackjack

Crea un programa para jugar una versión simplificada del blackjack para un jugador.

El juego consta de varias iteraciones en las que se extraen cartas de la baraja gestionada por el ordenador. Cada carta tiene un valor numérico; el valor de la carta extraída se suma a la puntuación del jugador. Por lo tanto, en cada iteración, el programa debe generar una carta aleatoria. Las cartas serán objetos de la clase 'Card', que deben tener tres atributos: valor (`int`), nombre (`string`) y palo (`String`): tréboles ♣, diamantes ♦, corazones ♡ y picas ♠) (ver NOTA).

El valor de una carta se calcula de la siguiente manera: ases 1 punto, figuras 10 puntos, las demás cartas tienen su valor numérico.

Tras actualizar la puntuación del jugador, el programa debe imprimir el nombre de la carta, su valor y la puntuación del jugador. El jugador elige si desea continuar o no. El jugador gana si: (a) obtiene 21 puntos; (b) obtiene menos de 21 puntos, pero su puntuación está más cerca de 21 que la del ordenador (los puntos del ordenador se calculan como un número aleatorio en 1, ..., 21). El jugador pierde si: (a) obtiene más de 21 puntos; (b) obtiene menos de 21 puntos, pero su puntuación no está más cerca de 21 que la del ordenador.

NOTA: No es necesario comprobar si una carta se ha generado previamente; es decir, se permiten cartas repetidas.