Recuperación Temas 1, 2, y 3

Para dos números dados, a y b, es posible buscar el **máximo común divisor** (el número más grande que divide a ambos) mediante un algoritmo ineficiente pero sencillo: desde el menor de a y b, ir buscando, de forma decreciente, el primer número que divide a ambos simultáneamente. Realiza un programa que calcule el máximo común divisor de dos números.

```
public static void main(String[] args) {
	Scanner scanner = new Scanner(System.in);

	// Pedimos al usuario que ingrese los dos números
	System.out.print("Ingresa el primer número: ");
	int a = scanner.nextInt();

	System.out.print("Ingresa el segundo número: ");
	int b = scanner.nextInt();

	// Calculamos el MCD
	int mcd = calcularMCD(a, b);

	// Mostramos el resultado
	System.out.println("El máximo común divisor de " + a + " y " + b + "
es: " + mcd);
}

public static int calcularMCD(int a, int b) {
	// Encontramos el menor de los dos números
	int menor = a < b ? a : b;

	// Buscamos el primer número que divide a ambos de forma decreciente
	for (int i = menor; i > 0; i--) {
		if (a % i == 0 && b % i == 0) {
			return i; // El primer divisor común es el MCD
		}
	}

	return 1; // Si no se encuentra ningún divisor mayor, el MCD es 1
}
```

Pedir por consola un número n y dibujar un triángulo rectángulo de n elementos de lado, utilizando para ello asteriscos (*).

Por ejemplo, para n= 4:

```
****
***
**
*
```

```
public static void main(String[] args) {
	Scanner scanner = new Scanner(System.in);

	// Pedimos al usuario que ingrese el número n
	System.out.print("Ingresa el número de elementos del triángulo: ");
```

```
        int n = scanner.nextInt();

        // Dibujamos el triángulo
        dibujarTriangulo(n);

}

public static void dibujarTriangulo(int n) {
        // Bucle para imprimir las líneas del triángulo
        for (int i = n; i > 0; i--) {
                // Imprimir 'i' asteriscos en cada línea
                for (int j = 0; j < i; j++) {
                        System.out.print("*");
                }
                // Salto de línea después de cada línea de asteriscos
                System.out.println();
        }
}
```

Escribe un programa que cambie un dígito dentro de un número dando la posición y el valor nuevo. Las posiciones se cuentan de izquierda a derecha empezando por el 1. Suponemos que el usuario introduce correctamente los datos.

Ejemplo:

Por favor, introduzca un número entero positivo: 406783
Introduzca la posición dentro del número: 3
Introduzca el nuevo dígito: 1
El número resultante es 401783

```
// Crear un objeto Scanner para leer datos del usuario
Scanner scanner = new Scanner(System.in);

// Solicitar al usuario que introduzca un número entero positivo
System.out.print("Por favor, introduzca un número entero positivo: ");
int numero = scanner.nextInt();

// Solicitar la posición dentro del número (empezando desde 1)
System.out.print("Introduzca la posición dentro del número: ");
int posicion = scanner.nextInt();

// Solicitar el nuevo dígito a insertar en la posición indicada
System.out.print("Introduzca el nuevo dígito: ");
int nuevoDigito = scanner.nextInt();

// Convertir el número a cadena para trabajar con sus dígitos
String numeroStr = Integer.toString(numero);

// Validar que la posición es válida
if (posicion < 1 || posicion > numeroStr.length()) {
    System.out.println("La posición está fuera de rango.");
} else {
    // Construir el número modificado usando substring
    // Obtener la parte antes de la posición (excluyendo el dígito a cambiar)
    String parteAntes = numeroStr.substring(0, posicion - 1);
    // Obtener el nuevo dígito como String
    String nuevoDigitoStr = Integer.toString(nuevoDigito);
```

```
    // Obtener la parte después de la posición (excluyendo el dígito a cambiar)
    String parteDespues = numeroStr.substring(posicion);

    // Concatenar las partes para obtener el número modificado
    String numeroModificadoStr = parteAntes + nuevoDigitoStr + parteDespues;

    // Convertir el número modificado de vuelta a un entero
    int numeroModificado = Integer.parseInt(numeroModificadoStr);

    // Mostrar el número resultante
    System.out.println("El número resultante es " + numeroModificado);
}
```

Pedir los coeficientes de una ecuación de segundo grado y mostrar sus soluciones reales. Si no existen, habrá que indicarlo. Hay que tener en cuenta que las soluciones de una ecuación de segundo grado, $ax^2 + bx + c = 0$, son:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```
// Crear un objeto Scanner para leer datos del usuario
Scanner scanner = new Scanner(System.in);

// Solicitar los coeficientes a, b y c
System.out.print("Introduzca el coeficiente a: ");
float a = scanner.nextFloat();

System.out.print("Introduzca el coeficiente b: ");
float b = scanner.nextFloat();

System.out.print("Introduzca el coeficiente c: ");
float c = scanner.nextFloat();

// Verificar si a es 0, ya que no sería una ecuación cuadrática
if (a == 0) {
    System.out.println("No es una ecuación de segundo grado, ya que el
coeficiente a es 0.");
} else {
    // Calcular el discriminante
    float discriminante = b * b - 4 * a * c;

    // Verificar las soluciones según el discriminante
    if (discriminante > 0) {
        // Dos soluciones reales y distintas
        float x1 = (float) ((-b + Math.sqrt(discriminante)) / (2 * a));
        float x2 = (float) ((-b - Math.sqrt(discriminante)) / (2 * a));
        System.out.println("Las soluciones reales son: x1 = " + x1 + " y x2 =
" + x2);
    } else if (discriminante == 0) {
        // Una solución real doble
        float x = -b / (2 * a);
        System.out.println("La solución real es: x = " + x);
    } else {
        // No hay soluciones reales
        System.out.println("La ecuación no tiene soluciones reales.");
```

```
    }
}
```

Elabora el juego “Mejora tu cálculo mental” donde el programa mostrará de forma aleatoria una operación matemática compleja compuesta por sumas, restas y multiplicaciones y siempre con tres operadores (3-9*4+2), el usuario introducirá el resultado y el programa indicará si es correcto. Si es correcto mostrará otra operación matemática y en caso contrario seguirá solicitando la misma.

Ayuda: Para generar el número aleatorio puedes utilizar `int numAleatorio = Min + (int)(Math.random() * ((Max - Min) + 1))` donde Min y Max delimitan el rango de generación del número aleatorio.

```
public static void main(String[] args) {
    // Crear un objeto Scanner para leer datos del usuario
    Scanner scanner = new Scanner(System.in);

    // Variables para el rango de los números aleatorios
    int Min = 1;
    int Max = 10; // El rango de números aleatorios es de 1 a 10

    // Variable para controlar si la respuesta es correcta
    boolean respuestaCorrecta = false;

    // Bucle que se ejecuta mientras la respuesta sea incorrecta
    while (!respuestaCorrecta) {
        // Generar tres números aleatorios y dos operadores aleatorios
        int num1 = Min + (int)(Math.random() * ((Max - Min) + 1));
        int num2 = Min + (int)(Math.random() * ((Max - Min) + 1));
        int num3 = Min + (int)(Math.random() * ((Max - Min) + 1));

        // Generar tres operadores aleatorios
        char operador1 = generarOperador();
        char operador2 = generarOperador();

        // Crear la operación matemática
        String operacion = num1 + " " + operador1 + " " + num2 + " " +
operador2 + " " + num3;
        System.out.println("Resuelve la operación: " + operacion);

        // Evaluar el resultado de la operación
        int resultadoEsperado = evaluarOperacion(num1, num2, num3, operador1,
operador2);

        // Pedir al usuario el resultado
        int resultadoUsuario = scanner.nextInt();

        // Validar la respuesta
        if (resultadoUsuario == resultadoEsperado) {
            System.out.println("¡Correcto! Generando nueva operación...");
            respuestaCorrecta = true; // La respuesta fue correcta, salimos
del bucle
        } else {
            System.out.println("Incorrecto. Intenta de nuevo.");
            // El bucle continuará porque la respuesta no fue correcta
        }
    }
```

```
    // Cerrar el scanner
    scanner.close();
}

// Método para generar un operador aleatorio entre +, - y *
public static char generarOperador() {
    int opcion = (int)(Math.random() * 3); // Genera un número aleatorio
entre 0 y 2
    switch (opcion) {
        case 0: return '+';
        case 1: return '-';
        case 2: return '*';
        default: return '+'; // Por si acaso
    }
}

// Método para evaluar el resultado de una operación
public static int evaluarOperacion(int num1, int num2, int num3, char
operador1, char operador2) {
    int resultadoIntermedio;

    // Realizar la primera operación
    if (operador1 == '+') {
        resultadoIntermedio = num1 + num2;
    } else if (operador1 == '-') {
        resultadoIntermedio = num1 - num2;
    } else { // operador1 == '*'
        resultadoIntermedio = num1 * num2;
    }

    // Realizar la segunda operación con el tercer número
    if (operador2 == '+') {
        return resultadoIntermedio + num3;
    } else if (operador2 == '-') {
        return resultadoIntermedio - num3;
    } else { // operador2 == '*'
        return resultadoIntermedio * num3;
    }
}
```