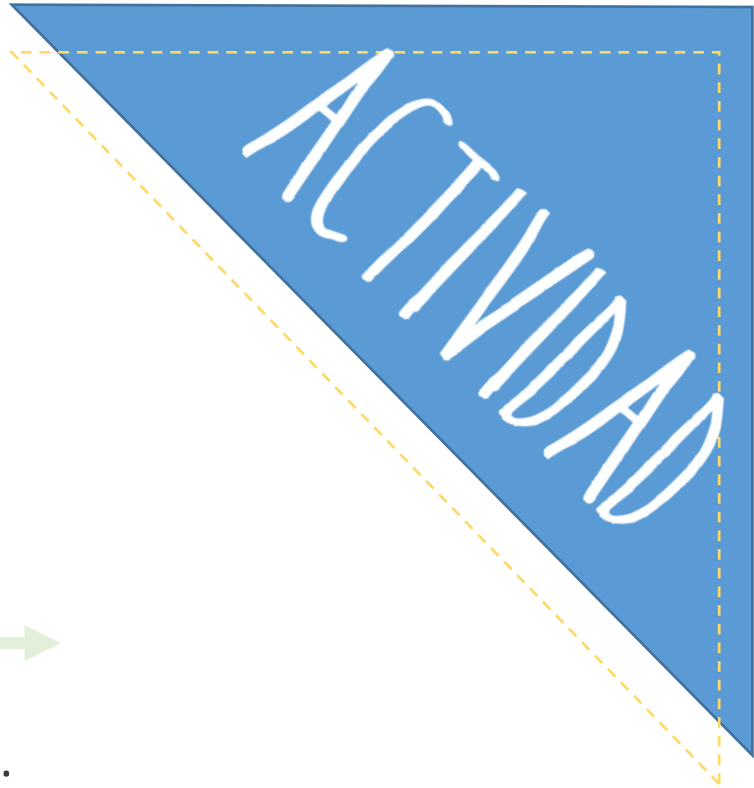

Dosier de Actividades del Tema 2:
Programación de hilos

Desarrollo de Aplicaciones Multiplataforma

Programación de Servicios y Procesos

Linkia FP

Ejercicio 1:

Implementar un programa en Java que realice la siguiente función:

- Crear un matriz de 25 filas y 10 columnas.
- Rellenar la matriz con números aleatorios entre 1 y 100 utilizando 10 hilos. Cada hilo se encargará de rellenar una determinada columna.
- Visualizar el contenido de la matriz.

Código

```
class ThreadFiller extends Thread {
    private final int[][] array;
    private final int minNum;
    private final int maxNum;

    ThreadFiller(String threadName, int[][] array, int minNum, int maxNum) {
        this.array = array;
        this.minNum = minNum;
        this.maxNum = maxNum;
        this.setName(threadName);
        this.start();
    }

    public void run() {
        int position = Integer.parseInt(this.getName());
        StringBuilder threadPhrase = new StringBuilder("Thread " + position + "
filled the positions: ");
        for (int j = 0; j < array.length; j++) {
            array[j][position] = getRandomNumberBetween(minNum, maxNum);

threadPhrase.append("[").append(position).append(",").append(j).append("] ");
        }
        System.out.println(threadPhrase);
    }

    public static void printFullArray(int[][] array, int maxNumLength) {
        System.out.println("\nHere goes the array filled:");

        for (int[] ints : array) {
            for (int j = 0; j < array[0].length; j++) {
                int spacesToAdd = maxNumLength - String.valueOf(ints[j]).length();
                System.out.print("[");
                for (int s = 0; s < spacesToAdd; s++) System.out.print(" ");
                System.out.print(ints[j]+"]");
            }
            System.out.println();
        }
    }

    private static int getRandomNumberBetween(int minNum, int maxNum) {
        return  (int)(Math.random()*(maxNum-minNum+1)+minNum);
    }
}

public class ArrayFiller {
```

```
    public static void main(String[] args) {
        int arrayX = 10;
        int arrayY = 25;
        int minNum = 1;
        int maxNum = 100;

        int[][] array = new int[arrayY][arrayX];
        ThreadFiller[] threads = new ThreadFiller[arrayX];

        for (int i = 0; i < arrayX; i++) {
            threads[i] = new ThreadFiller(String.valueOf(i), array, minNum,
maxNum);
        }

        for (int i = 0; i < arrayX; i++) {
            try {
                threads[i].join();
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
        }

        ThreadFiller.printFullArray(array, String.valueOf(maxNum).length());
    }
}
```

Funcionamiento

(El programa continua hacia la derecha, pero no cabian todas las lineas en la captura)

```
Thread 9 filled the positions: [9,0] [9,1] [9,2] [9,3] [9,4] [9,5] [9,6] [9,7] [9,8] [9,9] [9,10] [9,11] [9,12] [9,13] [
Thread 1 filled the positions: [1,0] [1,1] [1,2] [1,3] [1,4] [1,5] [1,6] [1,7] [1,8] [1,9] [1,10] [1,11] [1,12] [1,13] [
Thread 3 filled the positions: [3,0] [3,1] [3,2] [3,3] [3,4] [3,5] [3,6] [3,7] [3,8] [3,9] [3,10] [3,11] [3,12] [3,13] [
Thread 5 filled the positions: [5,0] [5,1] [5,2] [5,3] [5,4] [5,5] [5,6] [5,7] [5,8] [5,9] [5,10] [5,11] [5,12] [5,13] [
Thread 4 filled the positions: [4,0] [4,1] [4,2] [4,3] [4,4] [4,5] [4,6] [4,7] [4,8] [4,9] [4,10] [4,11] [4,12] [4,13] [
Thread 2 filled the positions: [2,0] [2,1] [2,2] [2,3] [2,4] [2,5] [2,6] [2,7] [2,8] [2,9] [2,10] [2,11] [2,12] [2,13] [
Thread 8 filled the positions: [8,0] [8,1] [8,2] [8,3] [8,4] [8,5] [8,6] [8,7] [8,8] [8,9] [8,10] [8,11] [8,12] [8,13] [
Thread 7 filled the positions: [7,0] [7,1] [7,2] [7,3] [7,4] [7,5] [7,6] [7,7] [7,8] [7,9] [7,10] [7,11] [7,12] [7,13] [
Thread 6 filled the positions: [6,0] [6,1] [6,2] [6,3] [6,4] [6,5] [6,6] [6,7] [6,8] [6,9] [6,10] [6,11] [6,12] [6,13] [
Thread 0 filled the positions: [0,0] [0,1] [0,2] [0,3] [0,4] [0,5] [0,6] [0,7] [0,8] [0,9] [0,10] [0,11] [0,12] [0,13] [

Here goes the array filled:
[ 35][ 33][ 39][  6][ 27][ 67][ 58][ 40][  6][ 90]
[ 42][  1][ 44][ 85][ 53][ 98][ 26][  9][ 89][  7]
[  8][ 81][ 16][ 97][ 11][ 36][ 63][ 61][ 36][100]
[ 72][ 16][ 23][ 20][ 33][ 10][ 39][ 34][ 53][ 65]
[ 65][ 50][ 43][ 61][100][ 42][ 25][ 34][ 96][ 65]
[ 88][ 93][ 62][ 38][ 30][ 97][ 58][ 83][ 22][ 45]
[ 52][ 48][ 31][ 37][ 80][ 43][ 86][ 42][ 90][ 22]
[ 58][ 68][ 84][ 52][ 18][ 38][ 12][ 38][ 20][  8]
[ 85][ 10][ 29][ 85][ 92][ 15][ 45][ 38][ 92][ 18]
[ 39][ 25][ 96][ 68][ 79][ 86][ 74][ 53][ 69][ 39]
[ 22][ 85][ 46][ 83][ 95][ 93][ 75][ 86][ 53][  2]
[ 41][  3][ 78][  6][ 31][ 83][ 54][ 71][ 48][  7]
[ 19][  4][ 69][ 39][ 46][ 90][ 56][ 43][ 34][ 89]
[ 61][100][ 16][ 42][ 57][ 21][ 30][ 50][  6][ 37]
[  2][ 29][ 37][ 93][ 71][  4][ 23][ 40][ 79][100]
[ 10][ 72][ 75][ 62][ 93][ 48][ 92][ 42][ 17][ 59]
[ 67][ 30][ 22][ 97][ 92][ 60][ 42][ 58][ 98][ 20]
[ 55][ 24][ 64][ 79][ 19][ 91][  8][ 32][ 33][ 68]
[ 10][ 71][ 33][  9][ 96][ 82][ 60][ 91][ 32][ 12]
[ 99][ 62][ 19][ 93][ 18][ 94][ 31][  9][ 46][ 15]
[  2][ 99][ 73][ 31][  3][ 52][ 82][ 54][ 39][ 90]
[  5][ 13][ 84][ 93][ 44][  4][ 53][ 32][ 64][ 89]
[ 58][ 12][ 86][ 95][ 97][ 13][ 32][ 98][ 27][ 60]
[ 71][ 11][ 78][ 55][ 79][ 49][ 20][ 35][  2][ 74]
[ 40][ 58][ 69][ 96][ 19][ 14][ 29][ 30][ 58][ 79]
```

Ejercicio 2:

Realizaremos una variación del ejercicio anterior.

- Creamos una matriz de 20 filas y 10 columnas. La rellenamos con números aleatorios entre (0-99) y luego la visualitzamos por pantalla.
- Pedimos un número por el teclado.
- Utilizando 10 hilos (uno para cada columna) buscaremos y visualizaremos en que posición se encuentra este número. Cada hilo solo se encarga de recorrer su columna.

```
[36][09][02][26][87][16][20][60][66][09]
[44][10][79][38][66][47][09][13][41][57]
[58][13][38][39][80][59][27][16][02][47]
[21][72][92][45][28][62][40][88][88][70]
[87][07][97][16][26][87][77][97][38][37]
[11][39][23][64][72][73][34][43][65][44]
[43][53][75][29][98][02][67][43][91][71]
[58][05][63][49][62][43][34][35][16][81]
[27][50][80][09][78][55][27][57][34][29]
[72][54][50][04][03][65][55][02][60][99]
[25][56][27][60][92][00][89][28][17][55]
[57][39][68][86][02][07][85][95][54][61]
[44][78][88][28][19][13][92][65][91][18]
[83][48][46][15][06][93][83][08][70][12]
[69][82][02][51][50][87][59][01][45][11]
[13][07][27][90][93][63][98][20][34][57]
[96][18][75][76][38][69][71][81][82][41]
[03][88][25][13][26][56][49][55][84][56]
[50][52][45][55][99][89][98][48][44][47]
[77][45][09][54][65][02][50][58][42][47]

Escribe el número a buscar (0-99): 92

Hilo:3 Posición: [4][3]
Hilo:5 Posición: [11][5]
Hilo:7 Posición: [13][7]
```

Código

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
class ThreadReader implements Runnable {
    private final int[][] array;
    private final int numberToFind;
    private final int column;
    ThreadReader(int column, int[][] array, int numberToFind) {
        this.array = array;
        this.numberToFind = numberToFind;
        this.column = column;
    }

    public void run() {
        for (int j = 0; j < array.length; j++) {
            if (array[j][column] == numberToFind) {
                System.out.println("Hilo:"+(column+1)+" Posición:
["+(j+1)+"]["+(column+1)+"]");
            }
        }
    }
}

public class ArrayPosition {
    public static void main(String[] args) {
        int arrayX = 10;
        int arrayY = 20;
        int minNum = 0;
        int maxNum = 99;
        int numberToFind;

        int[][] array = new int[arrayY][arrayX];

        arrayFiller(array, minNum, maxNum);

        printFullArray(array, String.valueOf(maxNum).length());

        numberToFind = getUserNumber(minNum, maxNum);

        for (int i = 0; i < arrayX; i++) new Thread( new ThreadReader(i, array,
numberToFind)).start();
    }

    public static void arrayFiller(int[][] array, int minNum, int maxNum) {
        for (int i = 0; i < array.length; i++) {
            for (int j = 0; j < array[0].length; j++) {
                array[i][j] = getRandomNumberBetween(minNum, maxNum);
            }
        }
    }
```

```
    public static void printFullArray(int[][] array, int maxNumLength) {
        System.out.println("Here goes the array filled:");

        for (int[] ints : array) {
            for (int j = 0; j < array[0].length; j++) {
                int spacesToAdd = maxNumLength - String.valueOf(ints[j]).length();
                System.out.print("[");
                for (int s = 0; s < spacesToAdd; s++) System.out.print(" ");
                System.out.print(ints[j]+"]");
            }
            System.out.println();
        }
    }




    public static int getRandomNumberBetween(int min, int max) {
        return  (int)(Math.random()*(max-min+1)+min);
    }

    private static int getUserNumber(int min, int max) {
        BufferedReader userIn = new BufferedReader(new InputStreamReader(System.in));

        while (true) {
            System.out.print("\nEscribe el número a buscar ("+min+"-"+max+"): ");
            try {
                int num = Integer.parseInt(userIn.readLine());
                if (num >= min && num <= max) return num;
                else System.out.print("No es un número comprendido entre "+min+" y "+max+".
Introduce un entero válido.");
            } catch (NumberFormatException e) {
                System.out.print("No es un número válido. Introduce un número entero.");
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        }
    }
}
```

Funcionamiento

```
Here goes the array filled:
[ 9][49][90][ 9][32][71][47][79][73][73]
[13][21][18][67][65][40][19][79][75][38]
[39][43][26][10][ 6][63][66][84][87][60]
[84][41][76][69][78][89][54][44][25][32]
[ 9][53][83][34][76][71][49][ 7][87][97]
[94][73][72][49][32][52][57][27][29][55]
[75][56][49][83][89][83][69][54][68][ 2]
[98][26][51][63][59][83][74][45][39][28]
[71][32][89][29][52][69][73][27][51][62]
[78][57][ 3][95][96][60][37][61][23][60]
[94][ 2][12][75][85][87][90][46][49][66]
[89][82][65][70][79][ 7][58][49][99][67]
[82][78][65][95][19][87][72][37][29][ 6]
[ 9][ 4][ 0][75][42][81][82][76][97][42]
[88][38][88][61][11][65][30][29][76][31]
[83][76][67][15][97][63][82][21][ 5][54]
[65][83][92][33][20][30][81][62][75][32]
[55][83][57][54][55][53][88][78][57][49]
[29][74][15][75][31][40][47][14][ 5][41]
[11][24][30][66][39][67][11][85][50][ 9]

Escribe el número a buscar (0-99): 9
Hilo:0 Posición: [1][1]
Hilo:0 Posición: [5][1]
Hilo:0 Posición: [14][1]
Hilo:9 Posición: [20][10]
Hilo:3 Posición: [1][4]
```

Ejercicio 3:

Construir un proyecto Java implementando una variación del productor consumidor.

- Crearemos 20 hilos que generen números aleatorios positivos y negativos. También podéis asignar los números manualmente.
- Estos hilos se encargarán de incrementar o decrementar una variable llamada total. Con un valor inicial de 0.
- El programa deberá tener en cuenta que esta variable nunca puede tener un valor negativo. Si el hilo que decrementa la variable es superior al valor debe dormirse hasta que no produzca un resultado negativo.

Ejemplo de solución:

```
Valor Inicial 2
Valor 5 Total ->: 7
Valor -5 Total ->: 2
Quiero restar 7 hay 2 -->Duermo
Valor 5 Total ->: 7
Despierto--> Valor -7 Total ->: 0
Quiero restar 4 hay 0 -->Duermo
Quiero restar 2 hay 0 -->Duermo
Valor 5 Total ->: 5
Despierto--> Valor -4 Total ->: 1
hilo 1: valor 3 -> total 3
hilo 2: valor -4 -> duermo
hilo 3: valor 5 -> total 8
despierto hilo 2 -> total 4
hilo 4: valor -1 -> total 3
…
```

Código

```
class Account {
    private int total;
    Account() {
        this.total = 0;
        System.out.println("Valor Inicial "+total);
    }
    private int getTotal() {return total;}
    private void setTotal(int total) {this.total = total;}
    public synchronized void updateTotal(String name, int value) {
        String chain = name+": valor "+value;
        if (getTotal() + value < 0) {
            chain += " hay " + getTotal() + " -->Duermo";
            System.out.println(chain);
            try {
                wait();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
            updateTotal(name, value);
        }
        else {
            setTotal(getTotal() + value);
            System.out.println(chain+" Total ->: "+getTotal());
        }
        notifyAll();
    }
}

class ThreadAccountUpdater implements Runnable {
    private final int id;
    private int value;
    private final Account account;

    ThreadAccountUpdater(int id, int value, Account account) {
        this.id = id;
        this.value = value;
        this.account = account;
    }

    public void run() {
        account.updateTotal("hilo "+id, value);
    }
}

public class ProductConsumer {
    public static void main(String[] args) {
```

```
        Account account = new Account();
        int threadQTY = 20;
        int minNum = -5;
        int maxNum = 5;

        for (int i = 0; i < threadQTY; i++) new Thread( new
ThreadAccountUpdater(i+1, getRandomNumberBetween(minNum, maxNum),account)
).start();
    }

    public static int getRandomNumberBetween(int min, int max) {
        int range = max - min + 1;
        int randomNumber = (int) (Math.random() * range) + min;
        return randomNumber;
    }

}
```

Funcionamiento

(El generador de numeros aleatorios genera entre –5 y 5, para que se diera el caso de que la suma del valor negativo fuera menor que 0)

```
Valor Inicial 0
hilo 1: valor 1 Total ->: 1
hilo 20: valor 4 Total ->: 5
hilo 19: valor -1 Total ->: 4
hilo 18: valor -1 Total ->: 3
hilo 17: valor -2 Total ->: 1
hilo 16: valor -3 hay 1 -->Duermo
hilo 15: valor 3 Total ->: 4
hilo 14: valor 0 Total ->: 4
hilo 13: valor 2 Total ->: 6
hilo 12: valor 5 Total ->: 11
hilo 11: valor -3 Total ->: 8
hilo 10: valor -3 Total ->: 5
hilo 9: valor 3 Total ->: 8
hilo 8: valor -3 Total ->: 5
hilo 7: valor 5 Total ->: 10
hilo 6: valor 5 Total ->: 15
hilo 5: valor 4 Total ->: 19
hilo 2: valor -2 Total ->: 17
hilo 4: valor 5 Total ->: 22
hilo 3: valor 4 Total ->: 26
hilo 16: valor -3 Total ->: 23
```

Formato de entrega.

Un archivo comprimido (en formato zip o rar) con el siguiente contenido:

- Archivos .java y jar de los diferentes ejercicios.
- Un documento en formato pdf con las capturas del código y una prueba de funcionamiento de cada actividad.