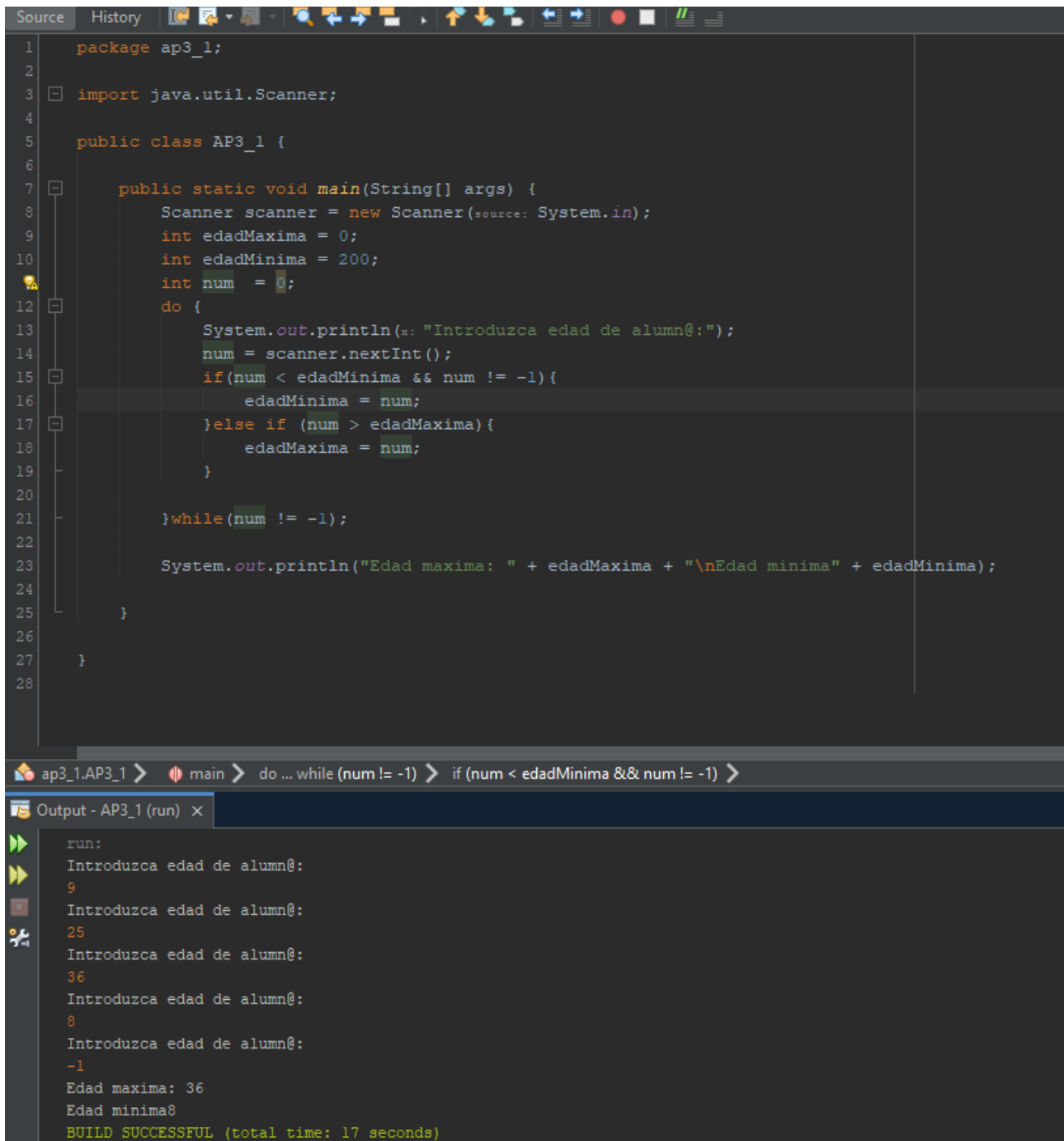


## Actividades propuestas. Realizar las siguientes 3.1, 3.2 y 3.3

### Actividad propuesta 3.1

Diseña una aplicación que muestre la edad máxima y mínima de un grupo de alumnos. El usuario introducirá las edades y terminará escribiendo un -1.



```
1 package ap3_1;
2
3 import java.util.Scanner;
4
5 public class AP3_1 {
6
7     public static void main(String[] args) {
8         Scanner scanner = new Scanner(System.in);
9         int edadMaxima = 0;
10        int edadMinima = 200;
11        int num = 0;
12        do {
13            System.out.println("Introduzca edad de alumn@:");
14            num = scanner.nextInt();
15            if (num < edadMinima && num != -1) {
16                edadMinima = num;
17            } else if (num > edadMaxima) {
18                edadMaxima = num;
19            }
20
21        } while (num != -1);
22
23        System.out.println("Edad maxima: " + edadMaxima + "\nEdad minima" + edadMinima);
24
25    }
26
27 }
28
```

ap3\_1.AP3\_1 > main > do ... while (num != -1) > if (num < edadMinima && num != -1) >

Output - AP3\_1 (run) X

```
run:
Introduzca edad de alumn@:
9
Introduzca edad de alumn@:
25
Introduzca edad de alumn@:
36
Introduzca edad de alumn@:
8
Introduzca edad de alumn@:
-1
Edad maxima: 36
Edad minima8
BUILD SUCCESSFUL (total time: 17 seconds)
```

### Actividad propuesta 3.2

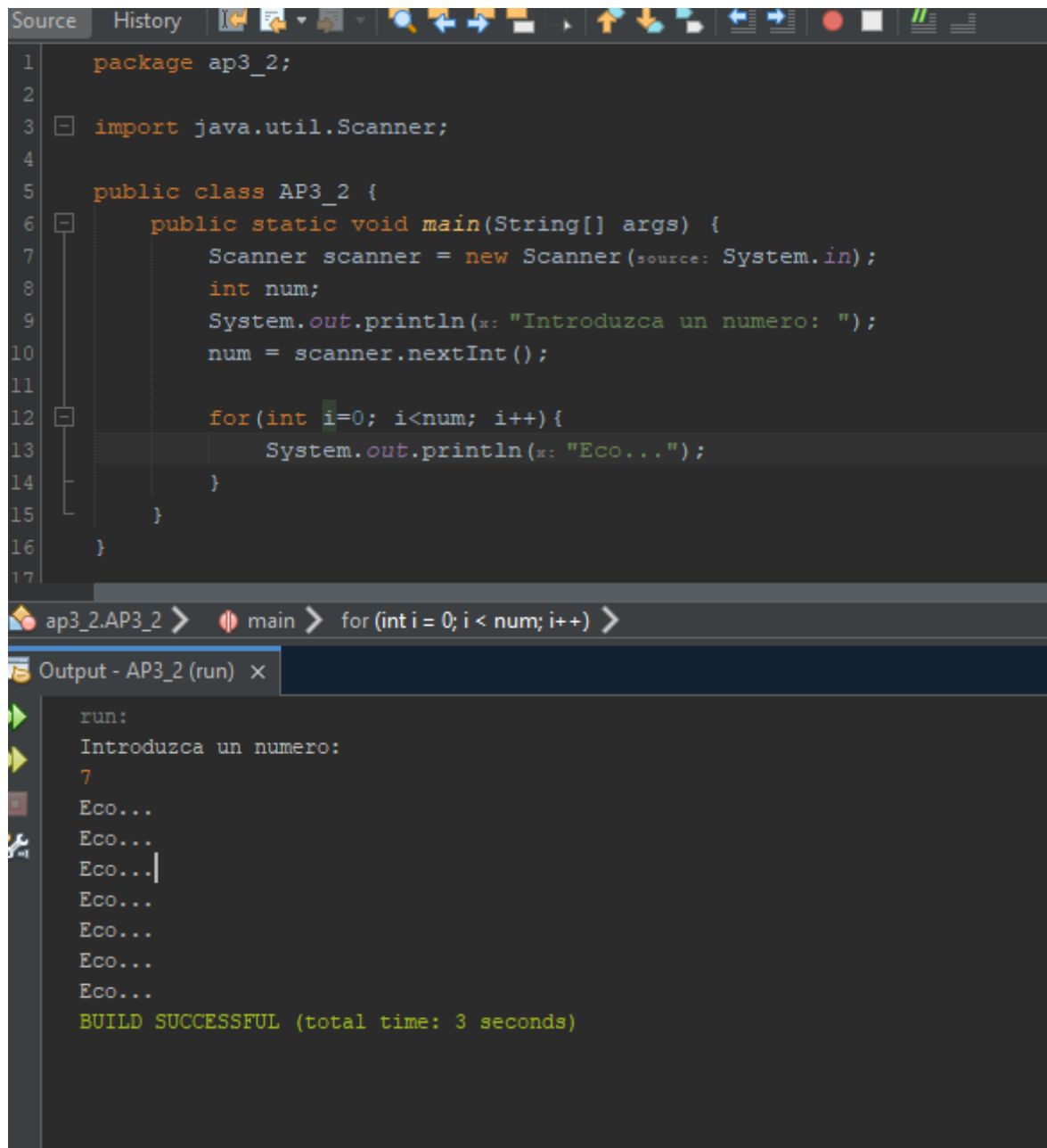
Implementa la aplicación Eco, que pide al usuario un número y muestra en pantalla la salida:

Eco...

Eco...

Eco...

Se muestra «Eco...» tantas veces como indique el número introducido. La salida anterior sería para el número 3.



The screenshot shows an IDE with a dark theme. The top toolbar contains various icons for file operations, running, and debugging. The source code editor displays the following Java code:

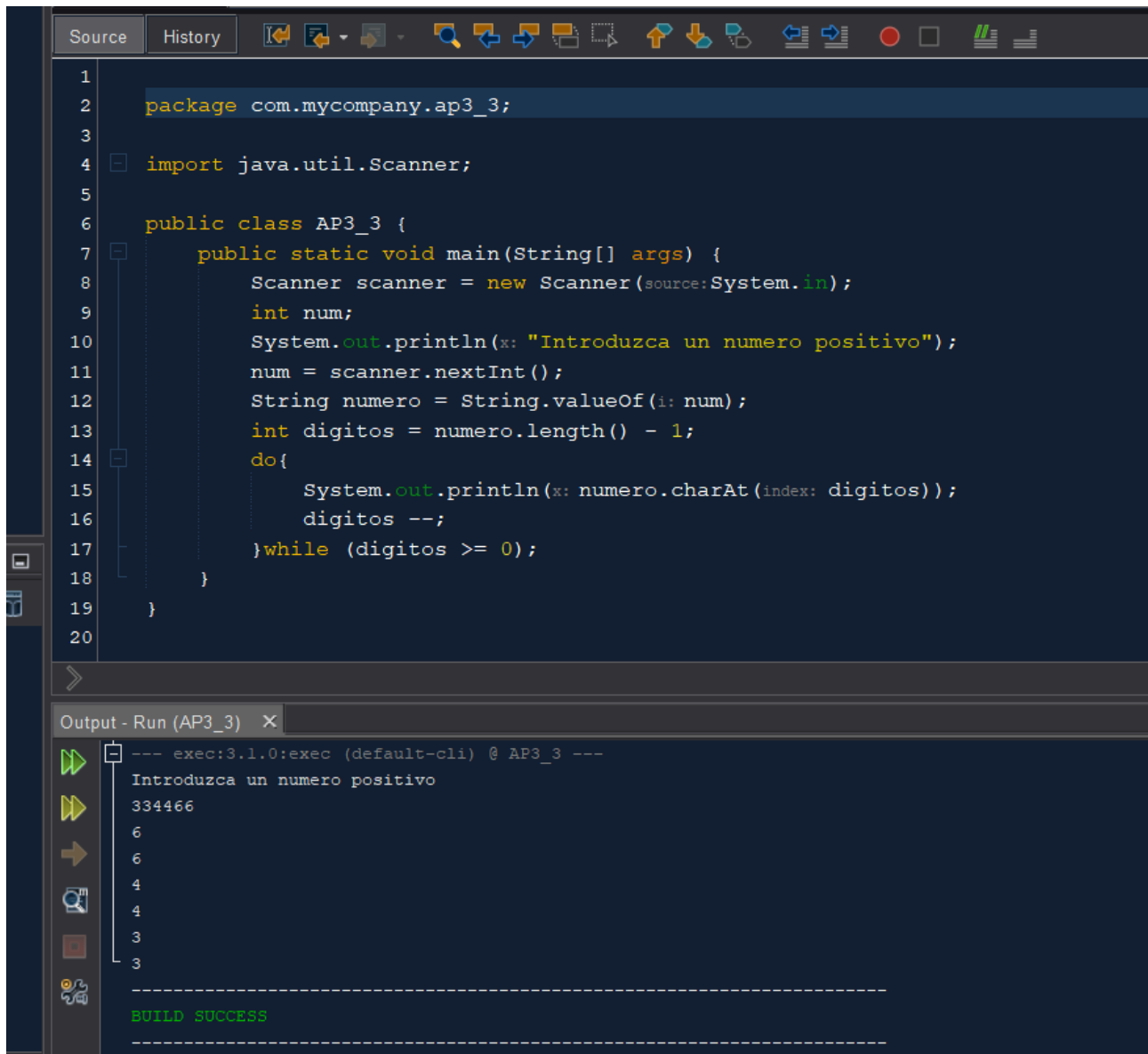
```
1 package ap3_2;
2
3 import java.util.Scanner;
4
5 public class AP3_2 {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         int num;
9         System.out.println("Introduzca un numero: ");
10        num = scanner.nextInt();
11
12        for(int i=0; i<num; i++){
13            System.out.println("Eco...");
14        }
15    }
16 }
17
```

Below the code editor, the breadcrumb navigation shows the path: `ap3_2.AP3_2 > main > for (int i = 0; i < num; i++)`. The output console, titled "Output - AP3\_2 (run)", shows the following text:

```
run:
Introduzca un numero:
7
Eco...
Eco...
Eco...
Eco...
Eco...
Eco...
Eco...
BUILD SUCCESSFUL (total time: 3 seconds)
```

### Actividad propuesta 3.3

Implementa un programa que pida al usuario un número positivo y lo muestre guarismo a guarismo. Por ejemplo, para el número 123, debe mostrar primero el 3, a continuación el 2 y por último el 1.



```
1
2 package com.mycompany.ap3_3;
3
4 import java.util.Scanner;
5
6 public class AP3_3 {
7     public static void main(String[] args) {
8         Scanner scanner = new Scanner(System.in);
9         int num;
10        System.out.println("Introduzca un numero positivo");
11        num = scanner.nextInt();
12        String numero = String.valueOf(num);
13        int digitos = numero.length() - 1;
14        do{
15            System.out.println(numero.charAt(index: digitos));
16            digitos --;
17        }while (digitos >= 0);
18    }
19 }
20
```

Output - Run (AP3\_3) X

```
--- exec:3.1.0:exec (default-cli) @ AP3_3 ---
Introduzca un numero positivo
334466
6
6
4
4
3
3
-----
BUILD SUCCESS
-----
```

## Actividades de aplicación. Realizar las siguientes 3.13, 3.14, 3.15, 3.17 y 3.19.

**3.13.** Escribe un programa que incremente la hora de un reloj. Se pedirán por teclado la hora, minutos y segundos, así como cuántos segundos se desea incrementar la hora introducida. La aplicación mostrará la nueva hora. Por ejemplo, si las 13:59:51 se incrementan en 10 segundos, resultan las 14:00:01.

```
1 package com.mycompany.aa3_13;
2
3 import java.util.Scanner;
4
5 public class Aa3_13 {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         int hora;
9         int min;
10        int seg;
11        int incrementaSeg;
12        do{
13            System.out.println(x: "Introduzca hora: ");
14            hora = scanner.nextInt();
15            System.out.println(x: "Introduzca minuto: ");
16            min = scanner.nextInt();
17            System.out.println(x: "Introduzca segundo: ");
18            seg = scanner.nextInt();
19        }while(!esCorrecto(hora, min, seg)); //Compruebo que sean datos válidos
20        System.out.println("Hora introducida: " + hora + ':' + min + ':' + seg);
21        do{
22            System.out.println(x: "¿Cuántos segundos incrementamos?");
23            incrementaSeg = scanner.nextInt();
24        }while(!(incrementaSeg >= 0));
25
26        while( incrementaSeg > 0){
27            if(seg < 59){
28                seg ++;
29                incrementaSeg--;
30            }else if(min < 59){
31                seg = 0;
32                min++;
33                incrementaSeg--;
34            }else if(hora < 23){
35                min = 0;
36                hora++;
37                incrementaSeg--;
38            }else{
39                seg = 0;
40                min = 0;
41                hora = 0;
42                incrementaSeg--;
43                System.out.println(x: "Hemos pasado al día siguiente!!");
44            }
45        }
46        System.out.println("La hora final es:" + hora + ':' + min + ':' + seg);
47    }
48
49
50    private static boolean esCorrecto(int hora, int min, int seg){
51        boolean esCorrecto = hora >= 0 && hora<24 && min >= 0 &&
52            min < 60 && seg >= 0 && seg < 60;
53        if (!esCorrecto){
54            System.out.println("La hora introducida no es correcta, "
55                + "introducir nuevos datos.");
56        }
57        return esCorrecto;
58    }
59 }
```

```
--- exec:3.1.0:exec (default-cli) @ Aa3_13 ---
Introduzca hora:
50
Introduzca minuto:
50
Introduzca segundo:
50
La hora introducida no es correcta, introducir nuevos datos.
Introduzca hora:
20
Introduzca minuto:
3
Introduzca segundo:
45
Hora introducida: 20:3:45
¿Cuantos segundos incrementamos?
20
La hora final es:20:4:5
-----
BUILD SUCCESS
```

```
--- exec:3.1.0:exec (default-cli) @ Aa3_13 ---
Introduzca hora:
10
Introduzca minuto:
10
Introduzca segundo:
50
Hora introducida: 10:10:50
¿Cuantos segundos incrementamos?
20
La hora final es:10:11:10
-----
BUILD SUCCESS
-----
Total time: 18.436 s
```

```
Introduzca hora:
23
Introduzca minuto:
59
Introduzca segundo:
20
Hora introducida: 23:59:20
¿Cuantos segundos incrementamos?
60
Hemos pasado al día siguiente!!
La hora final es:0:0:20
-----
BUILD SUCCESS
```

**3.14.** Realiza un programa que nos pida un número  $n$ , y nos diga cuántos números hay entre 1 y  $n$  que sean primos. Un número primo es aquel que solo es divisible por 1 y por él mismo. Veamos un ejemplo para  $n = 8$ :

Comprobamos todos los números del 1 al 8

{	1 → primo
	2 → primo
	3 → primo
	4 → no primo
	5 → primo
	6 → no primo
	7 → primo
	8 → no primo

Resultan un total de 5 números primos.

```

4
5 public class Aa3_14 {
6
7     public static void main(String[] args) {
8         Scanner scanner = new Scanner(System.in);
9         System.out.print(s: "Ingrese un número: ");
10        int numeroIngresado = scanner.nextInt();
11        int cantidadPrimos = contarPrimos(n: numeroIngresado);
12        System.out.println("Cantidad de números primos entre 1 y " + numeroIngresado + ": " + cantidadPrimos);
13        scanner.close();
14    }
15
16    private static int contarPrimos(int n) {
17        int contadorPrimos = 1;
18
19        for (int i = 1; i <= n; i++) {
20            if (esPrimo(num: i)) {
21                contadorPrimos++;
22            }
23        }
24        return contadorPrimos;
25    }
26
27    static boolean esPrimo(int num) {
28        if (num <= 1) {return false;}
29
30        for (int i = 2; i <= Math.sqrt(a: num); i++) {
31            if (num % i == 0) {
32                return false;
33            }
34        }
35        return true;
36    }
37 }

```

Compiling 1 source file with javac [debug target

```

--- exec:3.1.0:exec (default-cli) @ Aa3_14 ---
Ingrese un número: 8
Cantidad de números primos entre 1 y 8: 5
-----
BUILD SUCCESS
-----

```

```

--- exec:3.1.0:exec (default-cli) @ Aa3_14 ---
Ingrese un número: 15
Cantidad de números primos entre 1 y 15: 7
-----
BUILD SUCCESS
-----

```

- 3.15. Diseña una aplicación que dibuje el triángulo de Pascal, para  $n$  filas. Numerando las filas y elementos desde 0, la fórmula para obtener el  $m$ -ésimo elemento de la  $n$ -ésima fila es:

$$E(n, m) = \frac{n!}{m!(n-m)!}$$

Donde  $n!$  es el factorial de  $n$ .

Un ejemplo de triángulo de Pascal con 5 filas ( $n = 4$ ) es:

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
```

```
1 package com.mycompany.aa3_15;
2
3 import java.util.Scanner;
4
5 public class Aa3_15 {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         System.out.print(s: "Ingresa el número de filas para el Triángulo de Pascal: ");
9         int numRows = scanner.nextInt();
10
11         for (int i = 0; i < numRows+1; i++) {
12             // Calcular y mostrar los números en la fila
13             int coeficiente = 1;
14             for (int j = 0; j <= i; j++) {
15                 System.out.print(coeficiente + " ");
16                 coeficiente = coeficiente * (i - j) / (j + 1);
17             }
18             // Cambiar de línea para la siguiente fila
19             System.out.println();
20         }
21     }
22 }
23
```

Output - Run (Aa3\_15) ×

```
--- exec:3.1.0:exec (default-cli) @ Aa3_15 ---
Ingresa el número de filas para el Triángulo de Pascal: 7
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
```

BUILD SUCCESS

**3.17.** 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.

```
1 package com.mycompany.aa3_17;
2
3 import java.util.Scanner;
4
5 public class Aa3_17 {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         System.out.print(s: "Ingresa el primer número: ");
9         int numero1 = scanner.nextInt();
10        System.out.print(s: "Ingresa el segundo número: ");
11        int numero2 = scanner.nextInt();
12        // Calcular y mostrar el MCD
13        int mcd = calcularMCD(a: numero1, b: numero2);
14        System.out.println("El MaximoComunDivisor de " + numero1 + " y "
15            + numero2 + " es: " + mcd);
16    }
17
18    static int calcularMCD(int a, int b) {
19        while (b != 0) {
20            int temp = b;
21            b = a % b;
22            a = temp;
23        }
24        return a;
25    }
26 }
```

Output - Run (Aa3\_17) ×

```
--- exec:3.1.0:exec (default-cli) @ Aa3_17 ---
Ingresa el primer número: 50
Ingresa el segundo número: 30
El MaximoComunDivisor de 50 y 30 es: 10
```

-----  
BUILD SUCCESS  
-----



**3.19.** Calcula la raíz cuadrada de un número natural mediante aproximaciones. En el caso de que no sea exacta, muestra el resto. Por ejemplo, para calcular la raíz cuadrada de 23, probamos  $1^2 = 1$ ,  $2^2 = 4$ ,  $3^2 = 9$ ,  $4^2 = 16$ ,  $5^2 = 25$  (nos pasamos), resultando 4 la raíz cuadrada de 23 con un resto  $(23 - 16)$  de 7.

```
1  package com.mycompany.aa3_19;
2
3  import java.util.Scanner;
4
5  public class Aa3_19 {
6      public static void main(String[] args) {
7          Scanner scanner = new Scanner(System.in);
8          int resto = 0;
9          int raiz = 0;
10         int numero;
11         boolean continuar = true;
12         System.out.print(s: "Ingresa un número para calcular su raíz cuadrada: ");
13         numero = scanner.nextInt();
14         scanner.close();
15
16         if (numero < 0) {
17             System.out.println("No se puede calcular la raíz cuadrada de "
18                 + "un número negativo.");
19         }
20
21         for (int i = 1; continuar; i++) {
22             int result = i * i;
23             if (result < numero) {
24                 raiz = i;
25                 resto = numero - result;
26             } else {
27                 continuar = false;
28             }
29         }
30         System.out.println("La raíz cuadrada aproximada de " + numero
31             + " es: " + raiz + " el resto es " + resto);
32     }
33 }
```

```
--- exec:3.1.0:exec (default-cli) @ Aa3_19 ---
Ingresa un número para calcular su raíz cuadrada: 23
La raíz cuadrada aproximada de 23 es: 4 el resto es 7
-----
BUILD SUCCESS
-----
```