

CIFP César Manrique.

Programación 1º de Desarrollo de Aplicaciones Web

Profesor: José David Díaz Díaz

Actividades de la Unidad 3: Bucles



Esta obra está licenciada bajo la Licencia Creative Commons Atribución 4.0 Internacional.
Para ver una copia de esta licencia, visite <http://creativecommons.org/licenses/by/4.0/> o
envíe una carta a Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Índice

Actividades.....	1
Desarrollo.....	2
Actividades de comprobación.....	2
3.1. Un bucle do-while se ejecutará, como mínimo:.....	2
3.2. El uso de llaves para encerrar el bloque de instrucciones de un bucle:.....	2
3.3. La instrucción que permite detener completamente las iteraciones de un bucle es:.....	2
3.4. La instrucción que permite detener la iteración actual de un bucle, continuando con la siguiente, si procede, es:.....	2
3.5. De un bucle do-while , cuya condición depende de una serie de variables que en el bloque de instrucciones no se modifican , se puede afirmar :.....	3
3.6 . ¿ Cuántas veces se ejecutará el bloque de instrucciones del bucle más interno en el siguiente fragmento de código ?.....	3
3.7. Analiza el siguiente código y busca qué valores de a y b implican un menor número de iteraciones:.....	3
3.8. En cada iteración, el incremento de un bucle for se ejecuta :.....	4
3.9. Una variable que se declara dentro de su bloque de instrucciones solo se podrá utilizar :.....	4
3.10. En un bucle for, la inicialización, condición e incremento son :.....	4
Actividades de Aplicación.....	5
3.13. Escribe un programa que incremente la hora de un reloj.....	5

3.14. Realiza un programa que nos pida un número n, y nos diga cuántos números hay entre 1 y n que sean primos.....	7
3.16. Solicita al usuario un número n y dibuja un triángulo de base y altura n, de la forma (para n igual a 4) :.....	8
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.....	10
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.....	12
3.20. Escribe un programa que solicite al usuario las distintas cantidades de dinero de las que dispone. Por ejemplo: la cantidad de dinero que tiene en el banco, en una hucha, en un cajón y en los bolsillos. La aplicación mostrará la suma total de dinero de la que dispone el usuario. La manera de especificar que no se desea introducir más cantidades es mediante el cero.....	14

Actividades

Actividades de la Unidad 3: Bucles.

En este documento se detallan las actividades a realizar. Se entregará al profesor en la plataforma digital dos ficheros. Un primer fichero pdf con todas las actividades a realizar, el nombre del fichero será "unidad2 + nombre del alumno.pdf". Añadir en el fichero pdf por cada actividad de programación dos capturas de pantalla, una del código y otra de su ejecución. También en el fichero pdf copiar todas las preguntas y las respuestas correctas de las actividades de comprobación. Además, entregar un segundo fichero comprimido con todos los códigos fuentes de cada actividad de programación realizada.

Todas las actividades resueltas se deberán de analizar y no se entregarán.

A continuación, detallamos las actividades a realizar:

- **Actividades propuestas.** No realizar ninguna.
- **Actividades de comprobación.** Realizarlas todas. Copiar todas las preguntas y sus respuestas correctas.
- **Actividades de aplicación.** Realizar las siguientes **3.13, 3.14, 3.16, 3.17, 3.19 y 3.20.**
- **Actividades de ampliación.** No realizar ninguna.

[Volver al índice](#)

Desarrollo

Actividades de comprobación.

3.1. Un bucle `do-while` se ejecutará, como mínimo:

- b) Una vez

[Volver al índice](#)

3.2. El uso de llaves para encerrar el bloque de instrucciones de un bucle:

- b) Es opcional si el bloque está formado por una única instrucción.

[Volver al índice](#)

3.3. La instrucción que permite detener completamente las iteraciones de un bucle es:

- b) `break`.

[Volver al índice](#)

3.4. La instrucción que permite detener la iteración actual de un bucle, continuando con la siguiente, si procede, es:

- c) `continue`.

[Volver al índice](#)

3.5. De un bucle `do-while` , cuya condición depende de una serie de variables que en el bloque de instrucciones no se modifican , se puede afirmar :

- d) Ninguna de las opciones anteriores es correcta.

[Volver al índice](#)

3.6 . ¿ Cuántas veces se ejecutará el bloque de instrucciones del bucle más interno en el siguiente fragmento de código ?

```
for ( i = 1 ; i <= 10 ; i ++ ) {  
    for ( i = 1 ; i <= 5 ; i ++ ) {  
        System.out.println ( " Hola " );  
    }  
}
```

- c) 50 veces

[Volver al índice](#)

3.7. Analiza el siguiente código y busca qué valores de `a` y `b` implican un menor número de iteraciones:

```
for ( int i = a ; i <= a + b ; i ++ ) {  
    for ( int j = a + b ; j >= 0 ; j -- ) {  
        ...  
    }  
}
```

- c) `a=1` y `b=1`

[Volver al índice](#)

3.8. En cada iteración, el incremento de un bucle `for` se ejecuta :

- c) Después de evaluar la condición.

[Volver al índice](#)

3.9. Una variable que se declara dentro de su bloque de instrucciones solo se podrá utilizar :

- c) Dentro del bloque de instrucciones donde se ha declarado .

[Volver al índice](#)

3.10. En un bucle `for`, la inicialización, condición e incremento son :

- b) Todos opcionales.

(Es obligatorio el uso del punto y coma, el resultado será un bucle infinito)

[Volver al índice](#)

Actividades de Aplicación.

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 main;
2
3 import java.util.Scanner;
4
5 /**
6  * @author juancfm
7  */
8 public class Main {
9
10     /* Escribe un programa que incremente la hora de un reloj. Se pedirán por
11     teclado la hora, minutos y segundos, así como cuántos segundos se desea
12     incrementar la hora introducida. La aplicación mostrará la nueva hora.
13     Por ejemplo, si las 13:59:51 se incrementan en 10 segundos,
14     resultan las 14:00:01.*/
15
16     public static void main(String[] args) {
17         int hora, minutos, segundos, aux = 0, incremento = 0;
18         Scanner sc = new Scanner(System.in);
19
20         System.out.println("Vamos a calcular el incremento de unos "
21             + "segundos");
22         System.out.print("Ingrese la hora(0-23): ");
23         hora = sc.nextInt();
24         System.out.print("Ingrese los minutos(0-60): ");
25         minutos = sc.nextInt();
26         System.out.print("Ingrese los segundos(0-60): ");
27         segundos = sc.nextInt();
28         System.out.print("Ingrese el incremento en segundos: ");
29         incremento = sc.nextInt();
30         System.out.println("La hora introducida es:");
31         System.out.println(hora + ":" + minutos + ":" + segundos);
32         System.out.println("El incremento indicado fue de:");
33         System.out.println(incremento + " segundos");
34
35         incremento += segundos;
36         segundos = 0;
37
38         for (int i = 0; i < incremento; i++) {
39             segundos++;
40             if (segundos > 59) {
41                 segundos = 0;
42                 aux++;
43             }
44             incremento = minutos + aux;
45             minutos = aux = 0;
46             for (int i = 0; i < incremento; i++) {
47                 minutos++;
48                 if (minutos > 59) {
49                     minutos = 0;
50                     aux++;
51                 }
52             }
53             incremento = hora + aux;
54             hora = aux = 0;
55             for (int i = 0; i < incremento; i++) {
56                 hora++;
57                 if (hora > 23) {
58                     hora = 0;
59                     aux++;
60                 }
61             }
62
63             System.out.println("El resultado es:");
64             System.out.println(hora + ":" + minutos + ":" + segundos);
65             if (aux > 0) System.out.println("de " + aux +
66                 " día(s) después.");
67             sc.close();
68         }
69     }
70 }
71
72 }
```

```
run:
Vamos a calcular el incremento de unos segundos
Ingrese la hora(0-23): 13
Ingrese los minutos(0-60): 59
Ingrese los segundos(0-60): 59
Ingrese el incremento en segundos: 10
La hora introducida es:
13:59:59
El incremento indicado fue de:
10 segundos
El resultado es:
14:0:9
BUILD SUCCESSFUL (total time: 26 seconds)
|
```

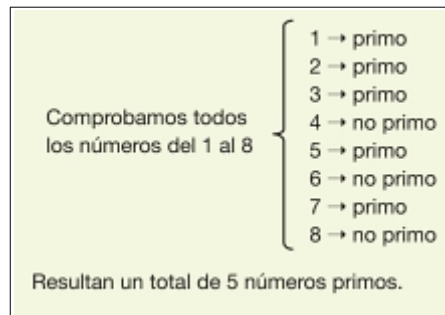
Actividades de la Unidad 3: Bucles

```
run:
Vamos a calcular el incremento de unos segundos
Ingrese la hora(0-23): 23
Ingrese los minutos(0-60): 59
Ingrese los segundos:(0-60): 59
Ingrese el incremento en segundos: 92000
La hora introducida es:
23:59:59
El incremento indicado fue de:
92000 segundos
El resultado es:
1:33:19
de 2 día(s) después.
BUILD SUCCESSFUL (total time: 45 seconds)
|
```

[Volver al índice](#)

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 :



```
package main;

import java.util.Scanner;

/**
 * @author juancfm
 */
public class Main {

    /**
     * Realiza un programa que nos pida un número n, y nos diga cuántos
     * números hay entre 1 y n que sean primos.
     */

    public static void main(String[] args) {
        int i, j, n, contador = 0;
        boolean primo;
        Scanner sc = new Scanner(System.in);
        System.out.println("Introduzca un número y le diré cuántos");
        System.out.print("números primos hay entre ese número y el 1: ");
        n = sc.nextInt();
        System.out.println("\n");
```

```
        for (i = 2; i <= n; i++) {
            primo = true;
            for (j = i - 1; j > 1; j--) {
                if (i % j == 0) {
                    primo = false;
                    break;
                }
            }
            if (primo) {
                System.out.println(i + " es primo");
                contador++;
            }
        }

        System.out.println("\nSe contabilizaron: " + contador +
            " números primos.");
        System.out.println("fin");
    }
}
```

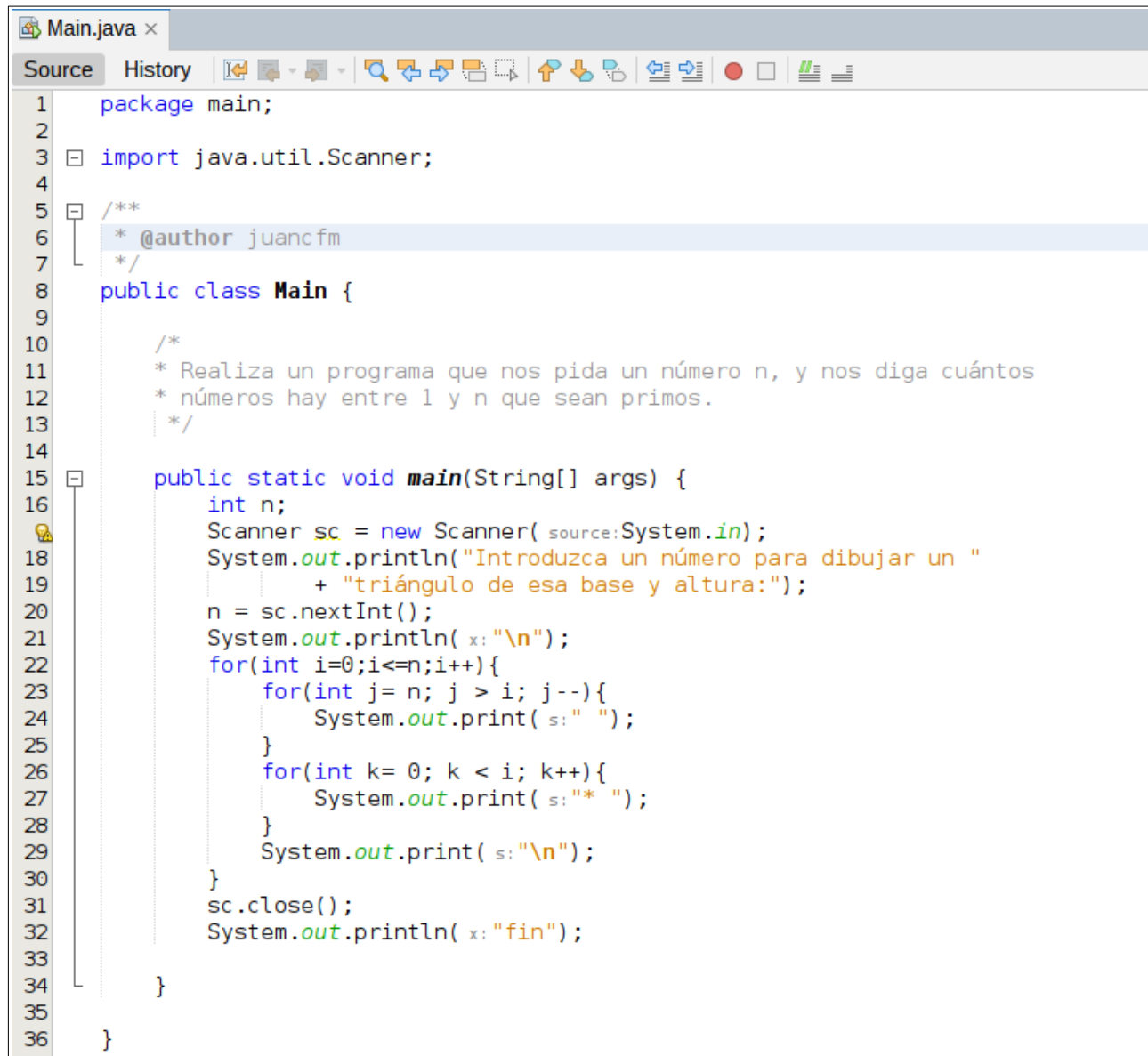
```
Output x
Debugger Console x Main (run) x
run:
Introduzca un número y le diré cuántos
números primos hay entre ese número y el 1: 10

2 es primo
3 es primo
5 es primo
7 es primo

Se contabilizaron: 4 números primos.
fin
BUILD SUCCESSFUL (total time: 3 seconds)
```

[Volver al índice](#)

3.16. Solicita al usuario un número n y dibuja un triángulo de base y altura n, de la forma (para n igual a 4) :

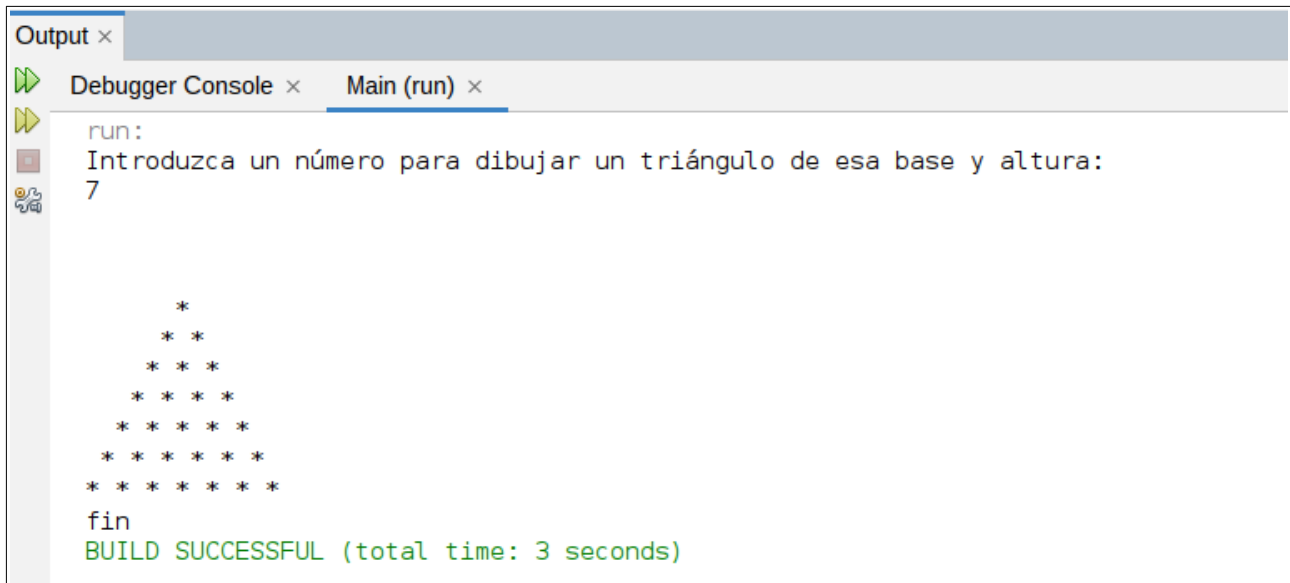


```

1  package main;
2
3  import java.util.Scanner;
4
5  /**
6   * @author juancfm
7   */
8  public class Main {
9
10     /*
11     * Realiza un programa que nos pida un número n, y nos diga cuántos
12     * números hay entre 1 y n que sean primos.
13     */
14
15     public static void main(String[] args) {
16         int n;
17         Scanner sc = new Scanner( source: System.in);
18         System.out.println("Introduzca un número para dibujar un "
19             + "triángulo de esa base y altura:");
20         n = sc.nextInt();
21         System.out.println( x: "\n");
22         for(int i=0;i<=n;i++){
23             for(int j= n; j > i; j--){
24                 System.out.print( s: " ");
25             }
26             for(int k= 0; k < i; k++){
27                 System.out.print( s: "* ");
28             }
29             System.out.print( s: "\n");
30         }
31         sc.close();
32         System.out.println( x: "fin");
33     }
34 }
35
36

```

Actividades de la Unidad 3: Bucles



The screenshot shows a debugger window with two tabs: "Debugger Console" and "Main (run)". The "Main (run)" tab is active, displaying the following output:

```
run:
Introduzca un número para dibujar un triángulo de esa base y altura:
7

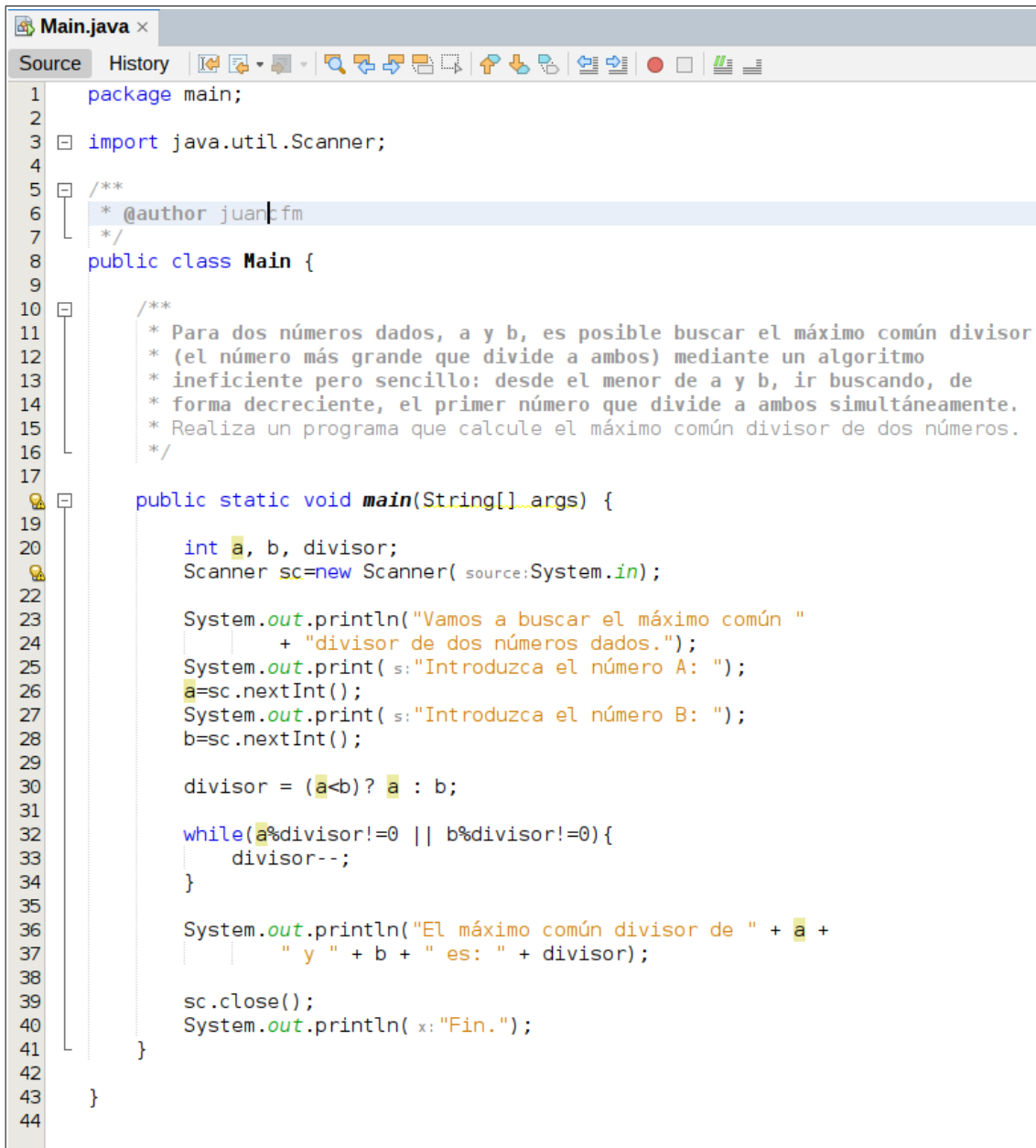
  *
 * *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
fin
BUILD SUCCESSFUL (total time: 3 seconds)
```

The output shows a prompt for a number, the user input '7', and a triangle of asterisks with 7 rows. The first row has 1 asterisk, the second has 2, and so on, up to 7 asterisks in the seventh row. The output ends with "fin" and a green status message "BUILD SUCCESSFUL (total time: 3 seconds)".

[Volver al índice](#)

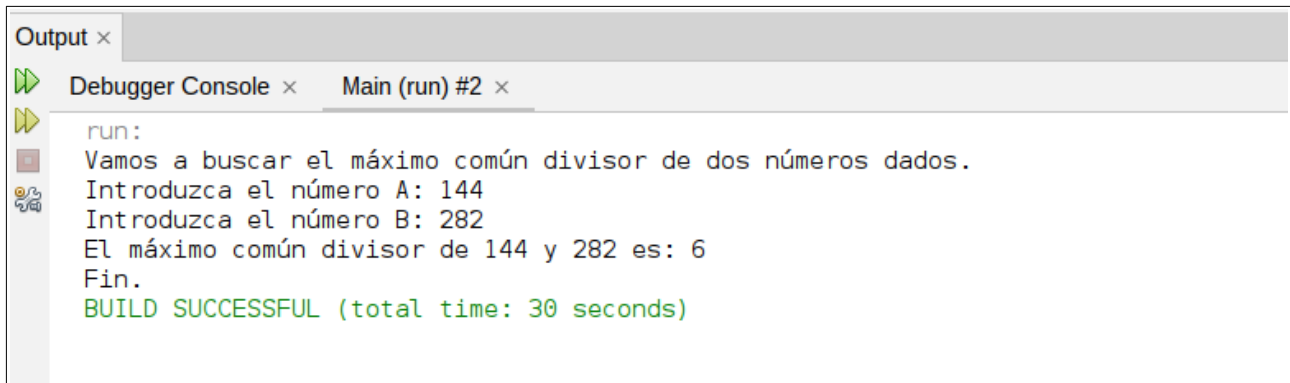
3.17. Para dos números dados, a y b, es posible buscar el máximo común divisor (el número más grande que divide a ambos) mediante un algoritmo ineficiente pero sencillo: desde el menor de a y b, ir buscando, de forma decreciente, el primer número que divide a ambos simultáneamente.

Realiza un programa que calcule el máximo común divisor de dos números.



```
1 package main;
2
3 import java.util.Scanner;
4
5 /**
6  * @author juan:fm
7  */
8 public class Main {
9
10     /**
11      * Para dos números dados, a y b, es posible buscar el máximo común divisor
12      * (el número más grande que divide a ambos) mediante un algoritmo
13      * ineficiente pero sencillo: desde el menor de a y b, ir buscando, de
14      * forma decreciente, el primer número que divide a ambos simultáneamente.
15      * Realiza un programa que calcule el máximo común divisor de dos números.
16      */
17
18     public static void main(String[] args) {
19
20         int a, b, divisor;
21         Scanner sc=new Scanner( source: System.in);
22
23         System.out.println("Vamos a buscar el máximo común "
24             + "divisor de dos números dados.");
25         System.out.print( s:"Introduzca el número A: ");
26         a=sc.nextInt();
27         System.out.print( s:"Introduzca el número B: ");
28         b=sc.nextInt();
29
30         divisor = (a<b)? a : b;
31
32         while(a%divisor!=0 || b%divisor!=0){
33             divisor--;
34         }
35
36         System.out.println("El máximo común divisor de " + a +
37             " y " + b + " es: " + divisor);
38
39         sc.close();
40         System.out.println( x:"Fin.");
41     }
42
43 }
44
```

Actividades de la Unidad 3: Bucles

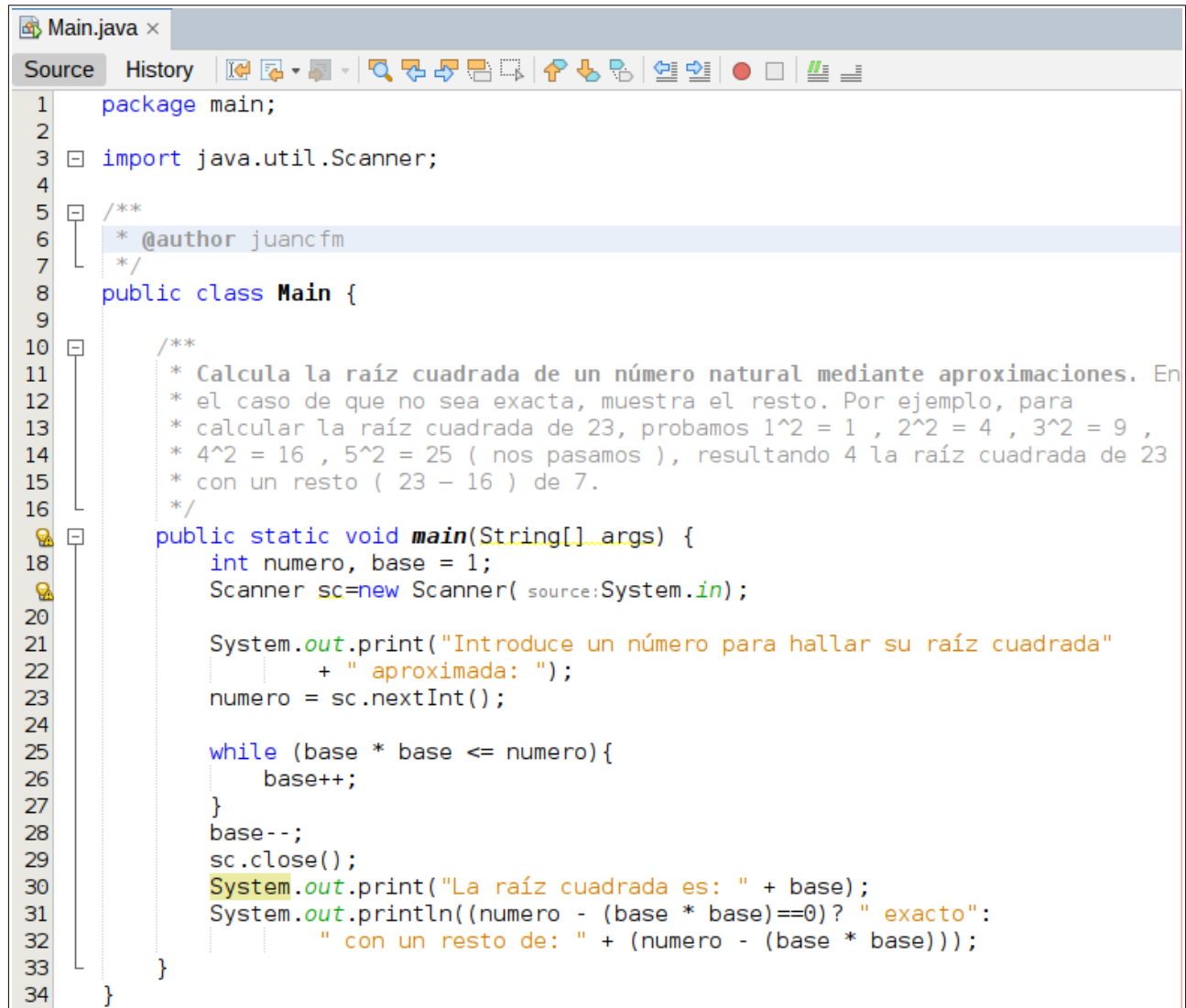


The screenshot shows a debugger console window with the following content:

```
Output x
Debugger Console x  Main (run) #2 x
run:
Vamos a buscar el máximo común divisor de dos números dados.
Introduzca el número A: 144
Introduzca el número B: 282
El máximo común divisor de 144 y 282 es: 6
Fin.
BUILD SUCCESSFUL (total time: 30 seconds)
```

[Volver al índice](#)

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.

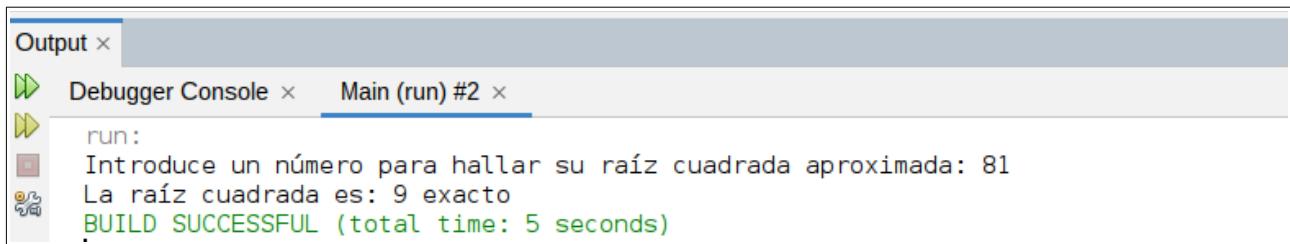


```

1  package main;
2
3  import java.util.Scanner;
4
5  /**
6   * @author juancfm
7   */
8  public class Main {
9
10     /**
11      * Calcula la raíz cuadrada de un número natural mediante aproximaciones. En
12      * el caso de que no sea exacta, muestra el resto. Por ejemplo, para
13      * calcular la raíz cuadrada de 23, probamos  $1^2 = 1$  ,  $2^2 = 4$  ,  $3^2 = 9$  ,
14      *  $4^2 = 16$  ,  $5^2 = 25$  ( nos pasamos ), resultando 4 la raíz cuadrada de 23
15      * con un resto (  $23 - 16$  ) de 7.
16      */
17     public static void main(String[] args) {
18         int numero, base = 1;
19         Scanner sc=new Scanner( source:System.in);
20
21         System.out.print("Introduce un número para hallar su raíz cuadrada"
22             + " aproximada: ");
23         numero = sc.nextInt();
24
25         while (base * base <= numero){
26             base++;
27         }
28         base--;
29         sc.close();
30         System.out.print("La raíz cuadrada es: " + base);
31         System.out.println((numero - (base * base)==0)? " exacto":
32             " con un resto de: " + (numero - (base * base)));
33     }
34 }

```

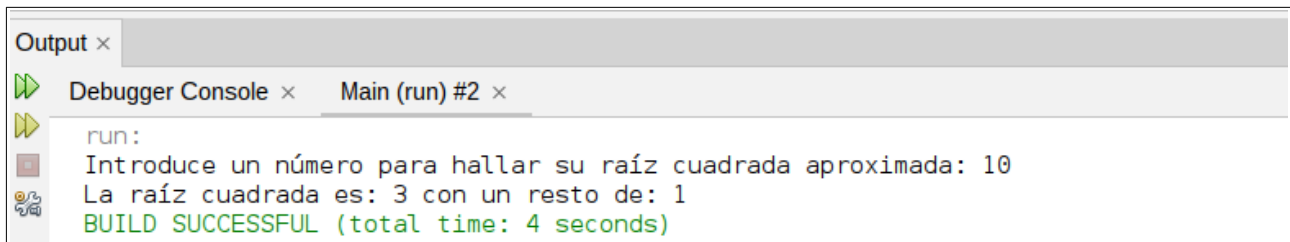

Actividades de la Unidad 3: Bucles



Output x

Debugger Console x Main (run) #2 x

run:
Introduce un número para hallar su raíz cuadrada aproximada: 81
La raíz cuadrada es: 9 exacto
BUILD SUCCESSFUL (total time: 5 seconds)



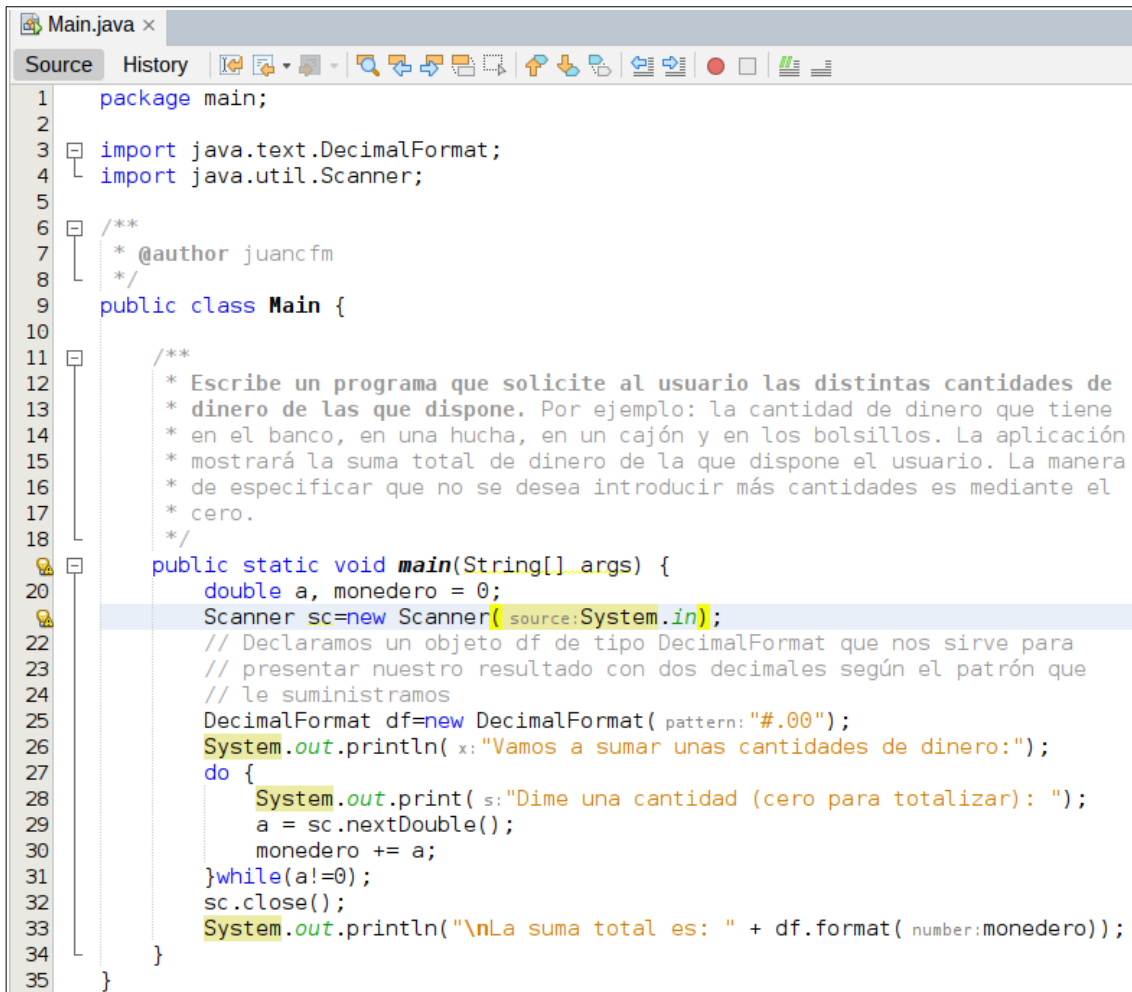
Output x

Debugger Console x Main (run) #2 x

run:
Introduce un número para hallar su raíz cuadrada aproximada: 10
La raíz cuadrada es: 3 con un resto de: 1
BUILD SUCCESSFUL (total time: 4 seconds)

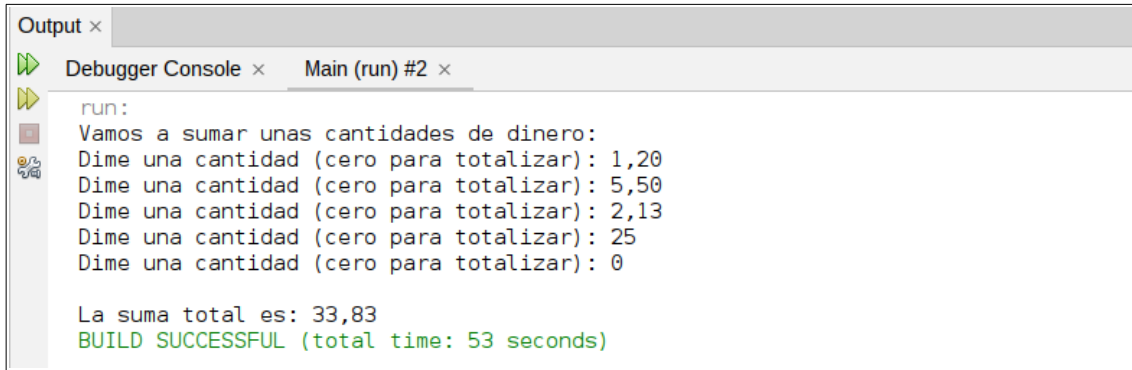
[Volver al índice](#)

3.20. Escribe un programa que solicite al usuario las distintas cantidades de dinero de las que dispone. Por ejemplo: la cantidad de dinero que tiene en el banco, en una hucha, en un cajón y en los bolsillos. La aplicación mostrará la suma total de dinero de la que dispone el usuario. La manera de especificar que no se desea introducir más cantidades es mediante el cero.



```
1 package main;
2
3 import java.text.DecimalFormat;
4 import java.util.Scanner;
5
6 /**
7  * @author juancfm
8  */
9 public class Main {
10
11     /**
12      * Escribe un programa que solicite al usuario las distintas cantidades de
13      * dinero de las que dispone. Por ejemplo: la cantidad de dinero que tiene
14      * en el banco, en una hucha, en un cajón y en los bolsillos. La aplicación
15      * mostrará la suma total de dinero de la que dispone el usuario. La manera
16      * de especificar que no se desea introducir más cantidades es mediante el
17      * cero.
18      */
19     public static void main(String[] args) {
20         double a, monedero = 0;
21         Scanner sc=new Scanner(System.in);
22         // Declaramos un objeto df de tipo DecimalFormat que nos sirve para
23         // presentar nuestro resultado con dos decimales según el patrón que
24         // le suministramos
25         DecimalFormat df=new DecimalFormat( pattern:"#.00");
26         System.out.println( x:"Vamos a sumar unas cantidades de dinero:");
27         do {
28             System.out.print( s:"Dime una cantidad (cero para totalizar): ");
29             a = sc.nextDouble();
30             monedero += a;
31         }while(a!=0);
32         sc.close();
33         System.out.println("\nLa suma total es: " + df.format( number:monedero));
34     }
35 }
```

Actividades de la Unidad 3: Bucles



The screenshot shows a debugger console window with the following content:

```
Output x
Debugger Console x  Main (run) #2 x
run:
Vamos a sumar unas cantidades de dinero:
Dime una cantidad (cero para totalizar): 1,20
Dime una cantidad (cero para totalizar): 5,50
Dime una cantidad (cero para totalizar): 2,13
Dime una cantidad (cero para totalizar): 25
Dime una cantidad (cero para totalizar): 0

La suma total es: 33,83
BUILD SUCCESSFUL (total time: 53 seconds)
```

[Volver al índice](#)