

ACTIVIDADES DE LA UNIDAD DE TRABAJO

4 - PRO

Autor: Derimán Tejera Fumero. 1ºDAW

Fecha: 11/11/2022

Índice

Actividades de comprobación	2
Actividades de aplicación	4

Actividades de comprobación

4.1. Los parámetros en la llamada a una función en Java pueden ser opcionales si:

- a) Todos los parámetros son del mismo tipo.
- b) Todos los parámetros son de distinto tipo.
- c) Nunca pueden ser opcionales.
- d) Siempre que el tipo devuelto no sea void.

4.2. Una variable local (declarada dentro de una función) puede usarse:

- a) En cualquier lugar del código.
- b) Solo dentro de main().
- c) Solo en la función donde se ha declarado.
- d) Ninguna de las opciones anteriores es correcta.

4.3. El tipo devuelto de todas las funciones definidas en nuestro programa tiene que ser siempre:

- a) int.
- b) double.
- c) void.
- d) Ninguna de las opciones anteriores es correcta.

4.4. ¿Qué instrucción permite a una función devolver un valor?

- a) value.
- b) return.
- c) static.
- d) function.

4.5. La forma de distinguir entre dos o más funciones sobrecargadas es:

- a) Mediante su nombre.
- b) Mediante el tipo devuelto.
- c) Mediante el nombre de sus parámetros.

d) Mediante su lista de parámetros: número o tipos.

4.6. ¿Cuál es la definición de una función recursiva?

a) Es aquella que se invoca desde dentro de su propio bloque de instrucciones.

b) Es aquella cuyo nombre permite la sobrecarga y además realiza alguna comprobación mediante if.

c) Es aquella cuyo bloque de instrucciones utiliza alguna sentencia if (lo que llamamos caso base).

d) Es aquella que genera un bucle infinito.

4.7. El paso de parámetros a una función en Java es siempre:

a) Un paso de parámetros por copia.

b) Un paso de parámetros por desplazamiento.

c) Un paso de parámetros recursivo.

d) Un paso de parámetros funcional.

4.8. En el caso de que una función devuelva un valor, ¿cuál es la recomendación con respecto a la instrucción return?

a) Utilizar tantos como hagan falta.

b) Emplear tantos como hagan falta, pero siempre que se encuentren en bloques de instrucciones distintas.

c) Usar solo uno.

d) Utilizar solo uno, que será siempre la primera instrucción de la función.

4.9. ¿Cuáles de las siguientes operaciones se pueden implementar fácilmente mediante funciones recursivas?

a) $a^n = a \times a^{n-1}$

b) $\text{esPar}(n) = \text{esImpar}(n - 1)$ y $\text{esImpar}(n) = \text{esPar}(n - 1)$.

c) $\text{suma}(a, b) = \text{suma}(a + 1, b - 1)$.

d) Todas las respuestas anteriores son correctas.

4.10. En los identificadores de las funciones, al igual que en los de las variables, se recomienda utilizar la siguiente nomenclatura:

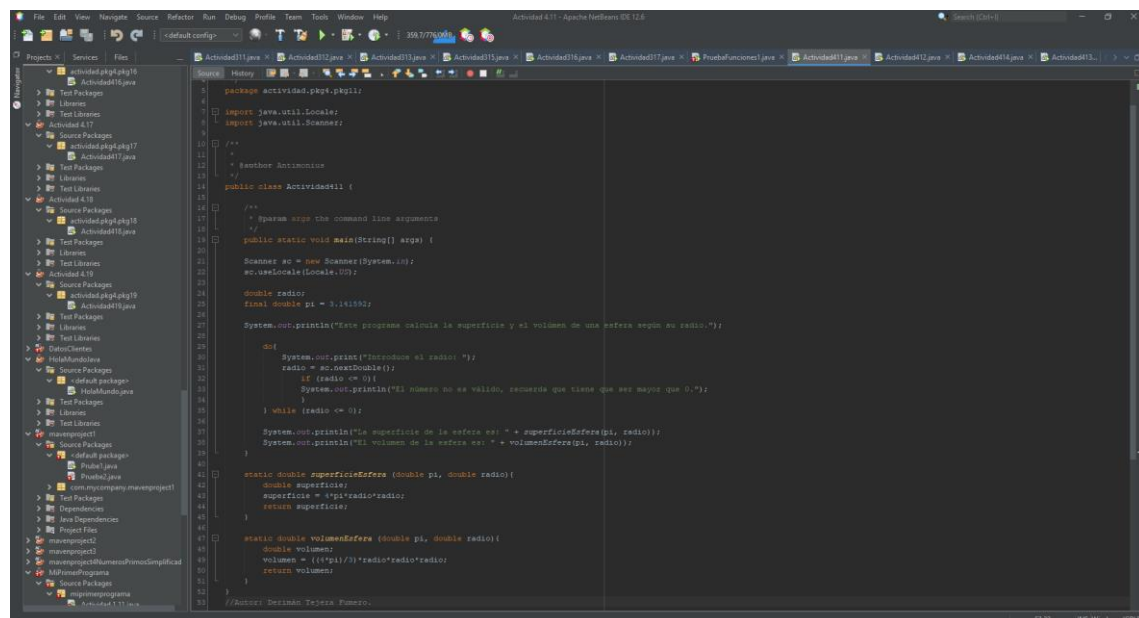
- a) suma_notas_alumnos().
- b) sumanotasalumnos().
- c) SumaNotasAlumnos().

d) sumaNotasAlumnos().

Actividades de aplicación

4.11. Diseña una función que calcule y muestre la superficie y el volumen de una esfera.

$$\text{Superficie} = 4\pi \cdot \text{radio}^2$$
$$\text{Volumen} = \frac{4\pi}{3} \cdot \text{radio}^3$$



```
1 package actividad.pkg4.pkg11;
2
3 import java.util.Locale;
4 import java.util.Scanner;
5
6 /**
7  *
8  * @author Anthonisse
9  */
10 public class Actividad11 {
11
12     /**
13      * @param args the command line arguments
14      */
15     public static void main(String[] args) {
16         Scanner sc = new Scanner(System.in);
17         Locale locale = Locale.ROOT;
18
19         double radio;
20         final double pi = 3.141592;
21
22         System.out.println("Este programa calcula la superficie y el volumen de una esfera segun su radio.");
23
24         do {
25             System.out.print("Introduce el radio: ");
26             radio = sc.nextDouble();
27             if (radio <= 0) {
28                 System.out.println("El numero no es valido, recuerda que tiene que ser mayor que 0.");
29             }
30         } while (radio <= 0);
31
32         System.out.println("La superficie de la esfera es: " + superficieEsfera(pi, radio));
33         System.out.println("El volumen de la esfera es: " + volumenEsfera(pi, radio));
34     }
35
36     static double superficieEsfera (double pi, double radio) {
37         double superficie;
38         superficie = 4*pi*radio*radio;
39         return superficie;
40     }
41
42     static double volumenEsfera (double pi, double radio) {
43         double volumen;
44         volumen = (4*pi/3)*radio*radio*radio;
45         return volumen;
46     }
47 }
48
49 //Autor: Desirée Tejera Fumero.
```

```
Output - Actividad 4.11 (run) ×
run:
Este programa calcula la superficie y el volumen de una esfera según su radio.
Introduce el radio: 50
La superficie de la esfera es: 31415.92
El volumen de la esfera es: 523598.6666666667
BUILD SUCCESSFUL (total time: 2 seconds)
```

4.12. Implementa la función

static double distancia(double x1, double y1, double x2, double y2) que calcula y devuelve la distancia euclídea que separa los puntos (x1. y1) y (x2. y2). La fórmula para calcular esta distancia es:

$$\text{distancia} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

```

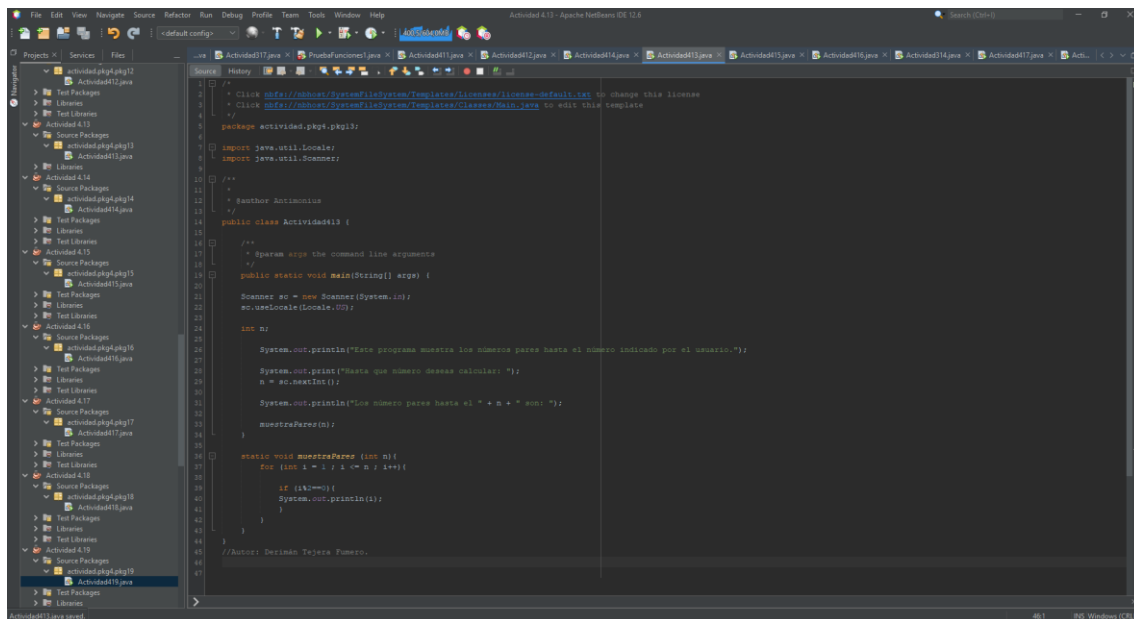
1  // Autor: Derivado Texeira Fumero.
2  // Fecha: 12/03/2023.
3
4  package Actividad412;
5
6  import java.util.Scanner;
7
8  public class Actividad412 {
9
10     /**
11      * Programa que calcula la distancia euclídea que separan los puntos x1, y1 y x2, y2.
12      */
13     public static void main(String[] args) {
14
15         Scanner sc = new Scanner(System.in);
16         sc.useLocale(Locale.ES);
17
18         double x1;
19         double x2;
20         double y1;
21         double y2;
22
23         System.out.println("Este programa calcula la distancia euclídea que separan los puntos x1, y1 y x2, y2.");
24
25         System.out.print("Introduce el valor de x1: ");
26         x1 = sc.nextDouble();
27
28         System.out.print("Introduce el valor de y1: ");
29         y1 = sc.nextDouble();
30
31         System.out.print("Introduce el valor de x2: ");
32         x2 = sc.nextDouble();
33
34         System.out.print("Introduce el valor de y2: ");
35         y2 = sc.nextDouble();
36
37         System.out.println("El valor de la distancia es: " + distancia(x1, y1, x2, y2));
38     }
39
40     static double distancia(double x1, double y1, double x2, double y2) {
41
42         double distancia;
43
44         distancia = ((x2-x1) * (x2-x1) + (y2-y1) * (y2-y1));
45         distancia = Math.sqrt(distancia);
46
47         return distancia;
48     }
49 }

```

```
Output - Actividad 4.12 (run)

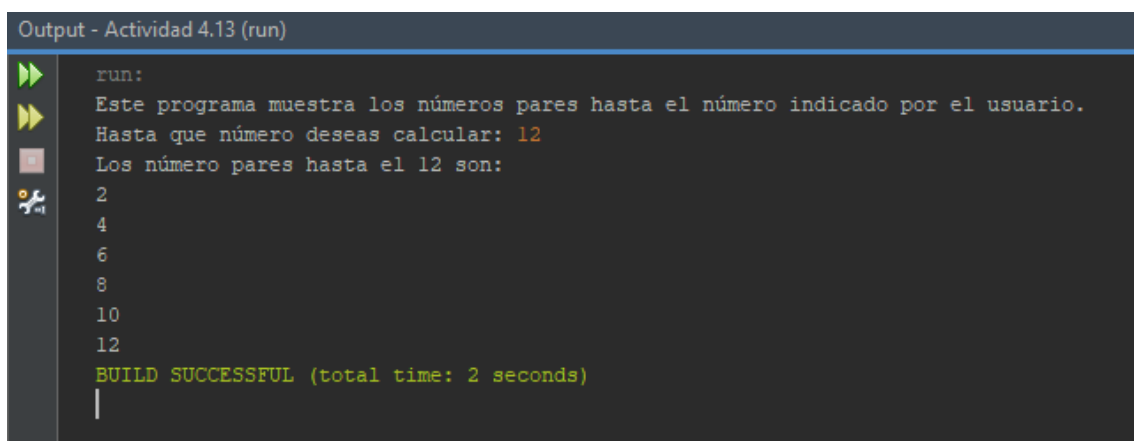
run:
Este programa calcula la distancia euclídea que separan los puntos x1, y1 y x2, y2..
Introduce el valor de x1: 5
Introduce el valor de y1: 10
Introduce el valor de x2: 85
Introduce el valor de y2: 90
El valor de la distancia es: 113.13708498984761
BUILD SUCCESSFUL (total time: 4 seconds)
```

4.13. Crea la función `muestraPares (int n)` que muestre por consola los primeros `n` números pares.



The screenshot shows an IDE window titled 'Actividad 4.13 - Apache NetBeans IDE 12.6'. The 'Source' tab is active, displaying the following Java code:

```
1  /**
2   * Click https://github.com/psalm/psalm/blob/master/LICENSE to change this license
3   * Click https://github.com/psalm/psalm/blob/master/CONTRIBUTING.md to edit this template
4   */
5  package actividad.pkg4.pkg13;
6
7  import java.util.Locale;
8  import java.util.Scanner;
9
10 /**
11  *
12  * Author Antimoonius
13  */
14 public class Actividad4.13 {
15
16     /**
17      * Specem exp the command line arguments
18      */
19     public static void main(String[] args) {
20
21         Scanner sc = new Scanner(System.in);
22         sc.useLocale(Locale.ES);
23
24         int n;
25
26         System.out.println("Este programa muestra los números pares hasta el número indicado por el usuario.");
27
28         System.out.print("Hasta que número deseas calcular: ");
29         n = sc.nextInt();
30
31         System.out.println("Los números pares hasta el " + n + " son:");
32
33         muestraPares(n);
34     }
35
36     static void muestraPares (int n){
37         for (int i = 1; i <= n; i++){
38             if (i%2==0){
39                 System.out.println(i);
40             }
41         }
42     }
43 }
44
45 //Autor: Darinán Tejera Páez.
```



The screenshot shows the 'Output - Actividad 4.13 (run)' window. The output text is as follows:

```
run:
Este programa muestra los números pares hasta el número indicado por el usuario.
Hasta que número deseas calcular: 12
Los números pares hasta el 12 son:
2
4
6
8
10
12
BUILD SUCCESSFUL (total time: 2 seconds)
```

4.14. Escribe una función a la que se pase como parámetros de entrada una cantidad de días, horas y minutos. La función calculará y devolverá el número de segundos que existen en los datos de entrada.

```

24  int minutos;
25
26  System.out.println("Este programa calcula en segundos los días, horas y minutos introducidos por el usuario.");
27
28  do {
29      System.out.print("Introduce los días: ");
30      dias = sc.nextInt();
31      if (dias <= 0) {
32          System.out.println("El número no es válido, recuerda que tiene que ser mayor que 0.");
33      }
34      while (dias <= 0) {
35      }
36
37      do {
38          System.out.print("Introduce las horas: ");
39          horas = sc.nextInt();
40          if (horas <= 0) {
41              System.out.println("El número no es válido, recuerda que tiene que ser mayor que 0.");
42          }
43          while (horas <= 0) {
44          }
45
46      } while (horas <= 0);
47
48      do {
49          System.out.print("Introduce los minutos: ");
50          minutos = sc.nextInt();
51          if (minutos <= 0) {
52              System.out.println("El número no es válido, recuerda que tiene que ser mayor que 0.");
53          }
54          while (minutos <= 0) {
55          }
56
57      } while (minutos <= 0);
58
59      System.out.println("El total es: " + totalSegundos(dias, horas, minutos) + " segundos.");
60
61  } while (true);
62
63  static int totalSegundos (int dias, int horas, int minutos) {
64
65      int Total;
66      int TotalDias;
67      int TotalHoras;
68      int TotalMinutos;
69
70      TotalDias = dias*86400;
71      TotalHoras = horas*3600;
72      TotalMinutos = minutos*60;
73
74      Total = TotalDias + TotalHoras + TotalMinutos;
75      return Total;
76  }
77
78  //Autor: Decindo Tejada Fuentes.

```

```

run:
Este programa calcula en segundos los días, horas y minutos introducidos por el usuario.
Introduce los días: 5
Introduce las horas: 10
Introduce los minutos: 20
El total es: 469200 segundos.
BUILD SUCCESSFUL (total time: 6 seconds)

```

4.15. Disena una función a la que se le pasan las horas y minutos de dos instantes de tiempo, con el siguiente prototipo:

static int diferenciaMin (int hora1, int minuto1, int hora2, int minuto2) La función devolverá la cantidad de minutos que existen de diferencia entre los dos instantes utilizados.

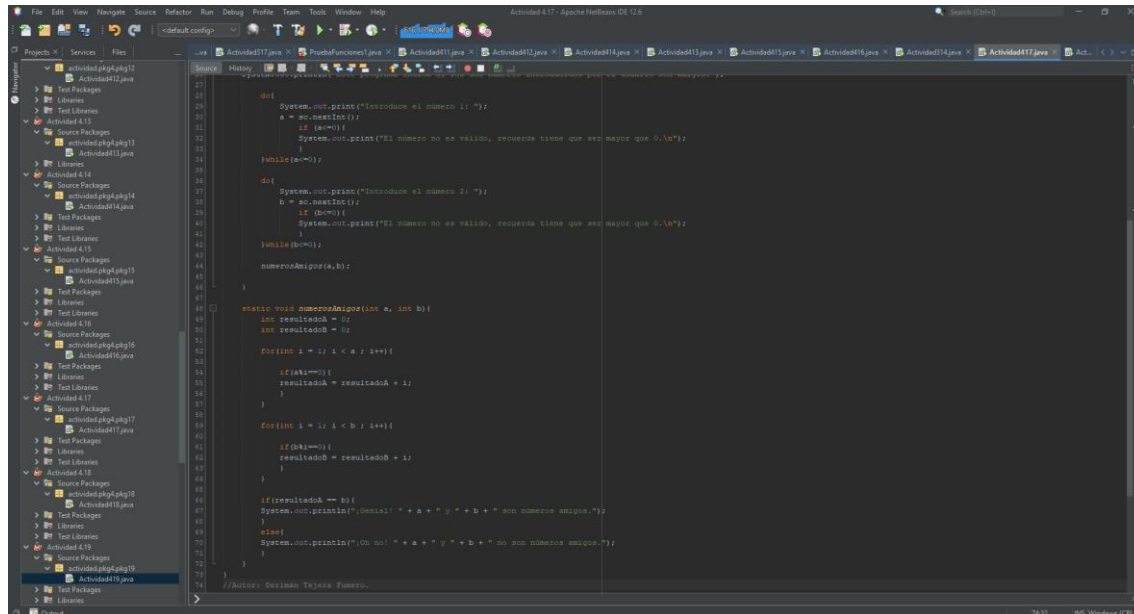
```

46  hora2 = sc.nextInt();
47  if (hora2 <= 0) {
48      System.out.println("El número no es válido, recuerda que tiene que ser mayor que 0.");
49  }
50  while (hora2 <= 0) {
51  }
52
53  do {
54      System.out.print("Tiempo 2. Minutos: ");
55      minuto2 = sc.nextInt();
56      if (minuto2 <= 0) {
57          System.out.println("El número no es válido, recuerda que tiene que ser mayor que 0.");
58      }
59      while (minuto2 <= 0) {
60      }
61
62      System.out.println("El total es: " + diferenciaMin(hora1, minuto1, hora2, minuto2) + " minutos.");
63
64  } while (true);
65
66  static int diferenciaMin (int hora1, int minuto1, int hora2, int minuto2) {
67
68      int Total, TotalHora = 0, TotalMinuto = 0;
69
70      if (hora2 > hora1 && minuto2 > minuto1) {
71          TotalHora = (hora2-hora1)*60;
72          TotalMinuto = (minuto2-minuto1);
73      }
74      else if (hora2 > hora1 && minuto2 < minuto1) {
75          TotalHora = (hora2-hora1)*60;
76          TotalMinuto = (minuto2-minuto1);
77      }
78      else if (hora2 < hora1 && minuto2 > minuto1) {
79          TotalHora = (hora1-hora2)*60;
80          TotalMinuto = (minuto2-minuto1);
81      }
82      else if (hora2 < hora1 && minuto2 < minuto1) {
83          TotalHora = (hora1-hora2)*60;
84          TotalMinuto = (minuto1-minuto2);
85      }
86
87      Total = TotalHora + TotalMinuto;
88      return Total;
89  }
90
91  //Autor: Decindo Tejada Fuentes.

```

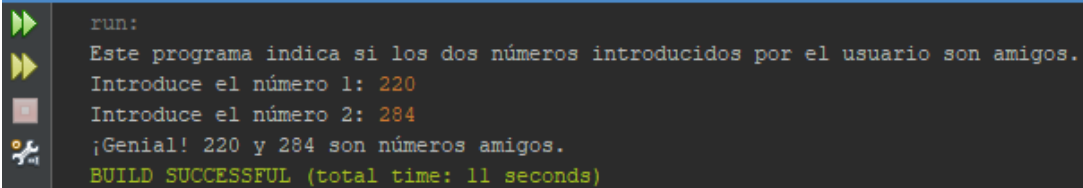

4.17. Escribe una función que decida si dos números enteros positivos son amigos. Dos números a y b son amigos si la suma de los divisores propios (distintos de él mismo) de a es igual a b . Y viceversa.

Para probar se pueden usar los números 220 y 284. que son amigos.



```
27
28
29     System.out.println("Introduce el número 1: ");
30     a = sc.nextInt();
31     if (a <= 0) {
32         System.out.println("El número no es válido, recuerda tiene que ser mayor que 0.");
33     }
34
35     while (a <= 0) {
36
37     }
38
39     System.out.println("Introduce el número 2: ");
40     b = sc.nextInt();
41     if (b <= 0) {
42         System.out.println("El número no es válido, recuerda tiene que ser mayor que 0.");
43     }
44     while (b <= 0) {
45
46     }
47
48     numeroDivisor(a, b);
49
50 }
51
52 static void numeroDivisor(int a, int b) {
53     int resultadoA = 0;
54     int resultadoB = 0;
55
56     for (int i = 1; i < a; i++) {
57
58         if (a % i == 0) {
59             resultadoA = resultadoA + i;
60         }
61
62     }
63
64     for (int i = 1; i < b; i++) {
65
66         if (b % i == 0) {
67             resultadoB = resultadoB + i;
68         }
69     }
70
71     if (resultadoA == b) {
72         System.out.println("¡Genial! " + a + " y " + b + " son números amigos.");
73     } else {
74         System.out.println("Oh no! " + a + " y " + b + " no son números amigos.");
75     }
76 }
77
78 //Autor: Verónica Tejeda Romero.
```

Output - Actividad 4.17 (run)



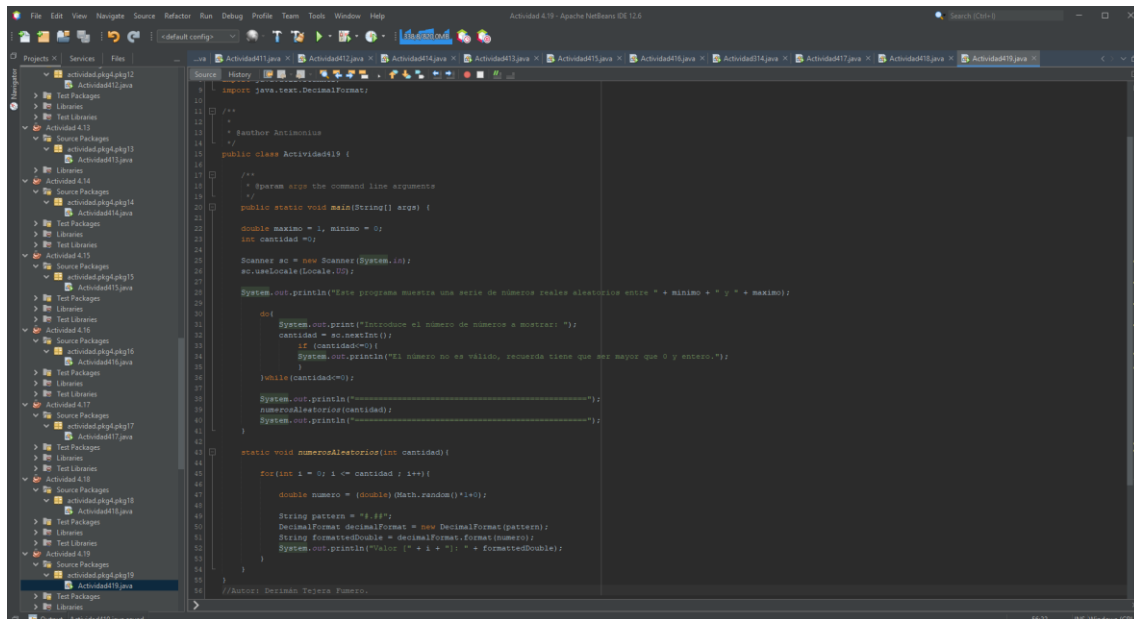
```
run:
Este programa indica si los dos números introducidos por el usuario son amigos.
Introduce el número 1: 220
Introduce el número 2: 284
¡Genial! 220 y 284 son números amigos.
BUILD SUCCESSFUL (total time: 11 seconds)
```

4.18. Crea una función que muestre por consola una serie de números aleatorios enteros. Los parámetros de la función serán: la cantidad de números aleatorios que se mostrarán y los valores mínimos y máximos que estos pueden tomar.

```
11 int cantidad, maximo, minimo;
12
13 Scanner sc = new Scanner(System.in);
14 sc.useLocale(Locale.ES);
15
16 System.out.println("Este programa muestra una serie de números aleatorios entre los valores indicados por el usuario.");
17
18 do {
19     System.out.print("Introduce el número de números a mostrar: ");
20     cantidad = sc.nextInt();
21     if (cantidad <= 0) {
22         System.out.println("El número no es válido, recuerda tiene que ser mayor que 0.");
23     }
24 } while (cantidad <= 0);
25
26 do {
27     System.out.print("Introduce el valor mínimo: ");
28     minimo = sc.nextInt();
29     if (minimo > 0) {
30         System.out.println("El número no es válido, recuerda tiene que ser mayor que 0.");
31     }
32 } while (minimo < 0);
33
34 do {
35     System.out.print("Introduce el valor máximo: ");
36     maximo = sc.nextInt();
37     if (maximo < minimo) {
38         System.out.println("El número no es válido, recuerda tiene que ser mayor que el valor mínimo (" + minimo + ").");
39     }
40 } while (maximo < minimo);
41
42 System.out.println("=====");
43 numerosAleatorios(cantidad, minimo, maximo);
44 System.out.println("=====");
45
46 }
47
48 public void numerosAleatorios(int cantidad, int minimo, int maximo) {
49
50     for (int i = 0; i <= cantidad; i++) {
51         int numero = (int) (Math.random() * (maximo - minimo));
52         System.out.println("Valor (" + i + "): " + numero);
53     }
54 }
55
56 // Autor: Darío Tejada Fumero.
```

```
run:
Este programa muestra una serie de números aleatorios entre los valores indicados por el usuario.
Introduce el número de números a mostrar: 15
Introduce el valor mínimo: 0
Introduce el valor máximo: 200
=====
Valor [0]: 90
Valor [1]: 81
Valor [2]: 131
Valor [3]: 43
Valor [4]: 186
Valor [5]: 129
Valor [6]: 74
Valor [7]: 6
Valor [8]: 173
Valor [9]: 129
Valor [10]: 25
Valor [11]: 123
Valor [12]: 132
Valor [13]: 145
Valor [14]: 124
Valor [15]: 107
=====
BUILD SUCCESSFUL (total time: 8 seconds)
```

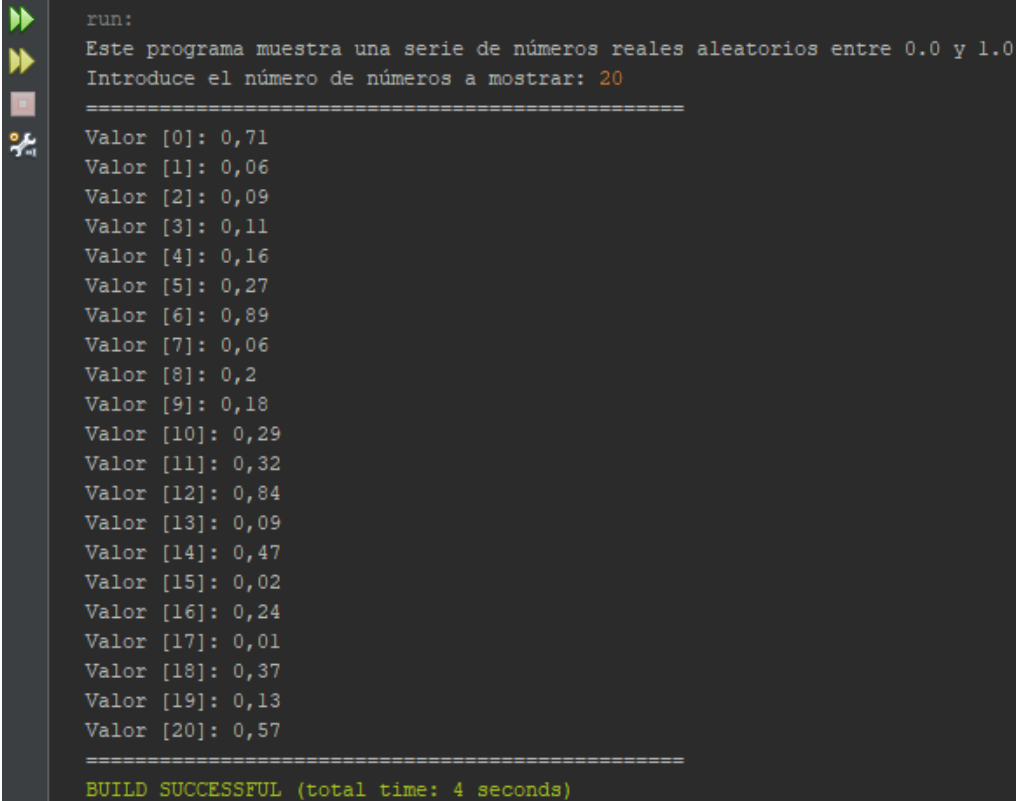
4.19. Sobrecarga la función realizada en la Actividad de aplicación 4.18 para que el único parámetro sea la cantidad de números aleatorios que se muestra por consola. Los números aleatorios serán reales y estarán comprendidos entre 0 y 1.



The screenshot shows an IDE window titled 'Actividad 4.19 - Apache NetBeans IDE 12.6'. The left sidebar displays a project tree with multiple 'Actividad' folders. The main editor area shows the source code for 'Actividad419.java'. The code is a Java program that generates random numbers. It includes a package declaration, imports, a class declaration, and a main method. The main method prompts the user to enter the number of numbers to display, validates the input, and then prints a series of random numbers formatted to two decimal places.

```
1  package actividades;
2
3  import java.text.DecimalFormat;
4
5  /**
6   * Author Antoninus
7   */
8  public class Actividad419 {
9
10     /**
11      * Ignora args the command line arguments
12      */
13     public static void main(String[] args) {
14
15         double maximo = 1, minimo = 0;
16         int cantidad = 0;
17
18         Scanner sc = new Scanner(System.in);
19         sc.useLocale(Locale.ES);
20
21         System.out.println("Este programa muestra una serie de números reales aleatorios entre " + minimo + " y " + maximo);
22
23         do {
24             System.out.print("Introduce el número de números a mostrar: ");
25             cantidad = sc.nextInt();
26             if (cantidad <= 0) {
27                 System.out.println("El número no es válido, recuerda tiene que ser mayor que 0 y entero.");
28             }
29             while (cantidad <= 0) {
30
31             }
32             System.out.println("=====");
33             numerosAleatorios(cantidad);
34             System.out.println("=====");
35         }
36
37         static void numerosAleatorios(int cantidad) {
38
39             for (int i = 0; i <= cantidad; i++) {
40
41                 double numero = (double) (Math.random() * 10);
42
43                 String pattern = "0.##";
44                 DecimalFormat decimalFormat = new DecimalFormat(pattern);
45                 String formattedDouble = decimalFormat.format(numero);
46                 System.out.println("Valor (" + i + "): " + formattedDouble);
47             }
48         }
49     }
50 }
51
52 // Autor: Derwin Tejeda Funes.
```

Output - Actividad 4.19 (run)



The screenshot shows the output of the program. It starts with a 'run:' prompt, followed by the program's introductory message. The user is prompted to enter the number of numbers to display, and the input '20' is shown. The program then prints a series of 20 random numbers, each labeled 'Valor [index]:'. The numbers are: 0,71; 0,06; 0,09; 0,11; 0,16; 0,27; 0,89; 0,06; 0,2; 0,18; 0,29; 0,32; 0,84; 0,09; 0,47; 0,02; 0,24; 0,01; 0,37; 0,13; 0,57. The output ends with a 'BUILD SUCCESSFUL (total time: 4 seconds)' message.

```
run:
Este programa muestra una serie de números reales aleatorios entre 0.0 y 1.0
Introduce el número de números a mostrar: 20
=====
Valor [0]: 0,71
Valor [1]: 0,06
Valor [2]: 0,09
Valor [3]: 0,11
Valor [4]: 0,16
Valor [5]: 0,27
Valor [6]: 0,89
Valor [7]: 0,06
Valor [8]: 0,2
Valor [9]: 0,18
Valor [10]: 0,29
Valor [11]: 0,32
Valor [12]: 0,84
Valor [13]: 0,09
Valor [14]: 0,47
Valor [15]: 0,02
Valor [16]: 0,24
Valor [17]: 0,01
Valor [18]: 0,37
Valor [19]: 0,13
Valor [20]: 0,57
=====
BUILD SUCCESSFUL (total time: 4 seconds)
```