

UNIVERSIDAD TECMILENIO

Mauricio Estrada De la Garza - AL02976904

Igor Sung Min Kim Juliao - AL02829189

30/08/2023

Programación Orientada a Objetos

Reto 2

Reto 2: Implementación de Funciones, Ciclos y Datos primitivos

Objetivo de la actividad:

El alumno desarrollará un programa en el cual se implementa los conceptos:

Requerimientos para la actividad:

Uso Java jdk a través de un IDE, preferentemente Netbeans.

Explorando Números amigos

• •

Conceptos a calificar:

1. Uso de Funciones.
2. Uso de Ciclos
3. El correcto uso y asignación de datos.

Instrucciones para el alumno:

En esta actividad, los alumnos crearán un programa en lenguaje Java para **identificar pares de números amigos dentro de un rango específico**. Dos números enteros positivos, A y B, son amigos si la suma de los divisores propios de A es igual a B, y la suma de los divisores propios de B es igual a A.

Por ejemplo, los números 220 y 284 son amigos porque:

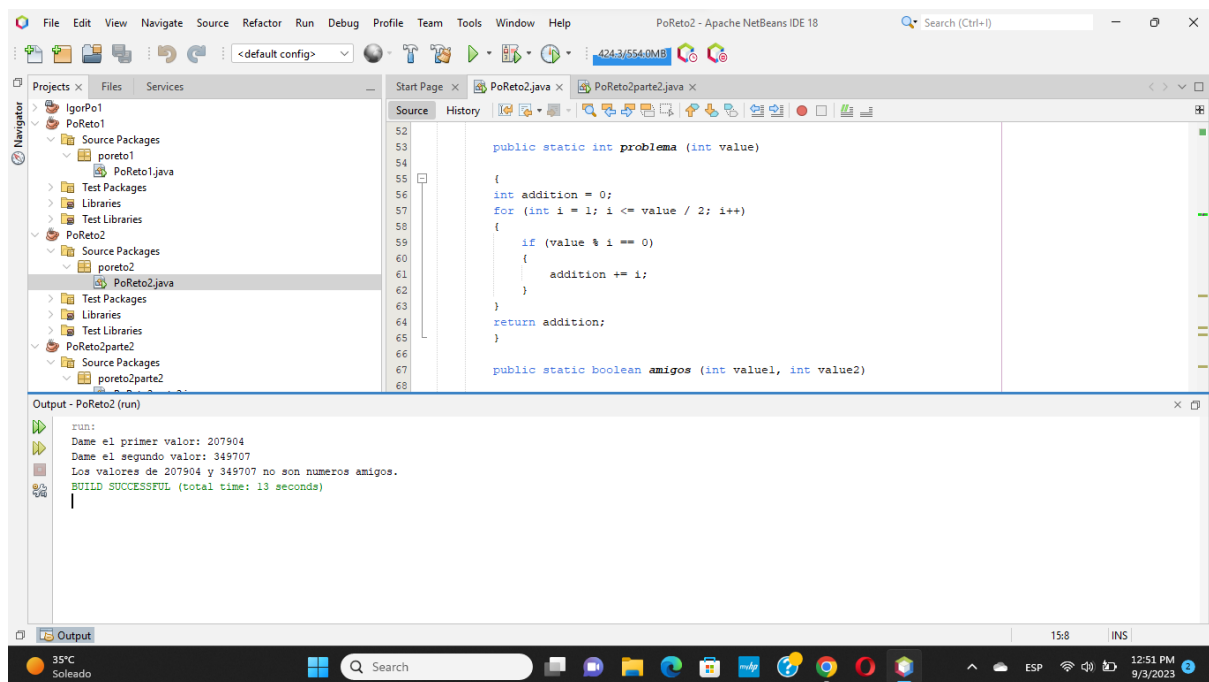
Los divisores propios de 220 son 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 y 110, cuya suma es 284

Los divisores propios de 284 son 1, 2, 4, 71 y 142, cuya suma es 220.

1. Definición de funciones: Definir al menos dos funciones:

2. Función para calcular la suma de los divisores propios de un número

3. Función para determinar si dos números son amigos.



```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
PoReTo2 - Apache NetBeans IDE 18
Search (Ctrl+I)
424.3/554.0MB

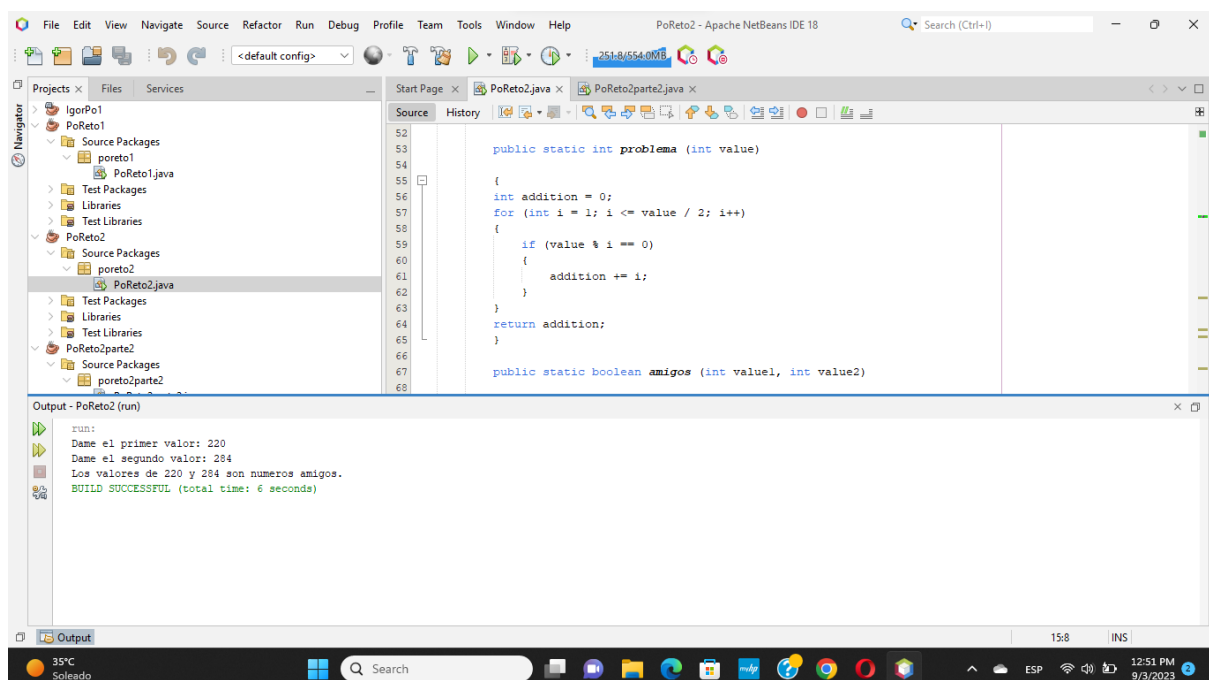
Projects: Files Services
IgorPo1
PoReTo1
Source Packages
poreto1
PoReTo1.java
Test Packages
Libraries
Test Libraries
PoReTo2
Source Packages
poreto2
PoReTo2.java
Test Packages
Libraries
Test Libraries
PoReTo2parte2
Source Packages
poreto2parte2

Source History
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68

public static int problema (int value)
{
    int addition = 0;
    for (int i = 1; i <= value / 2; i++)
    {
        if (value % i == 0)
        {
            addition += i;
        }
    }
    return addition;
}

public static boolean amigos (int value1, int value2)

Output - PoReTo2 (run)
run:
Dame el primer valor: 207904
Dame el segundo valor: 349707
Los valores de 207904 y 349707 no son numeros amigos.
BUILD SUCCESSFUL (total time: 13 seconds)
```



```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
PoReTo2 - Apache NetBeans IDE 18
Search (Ctrl+I)
261.8/554.0MB

Projects: Files Services
IgorPo1
PoReTo1
Source Packages
poreto1
PoReTo1.java
Test Packages
Libraries
Test Libraries
PoReTo2
Source Packages
poreto2
PoReTo2.java
Test Packages
Libraries
Test Libraries
PoReTo2parte2
Source Packages
poreto2parte2

Source History
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68

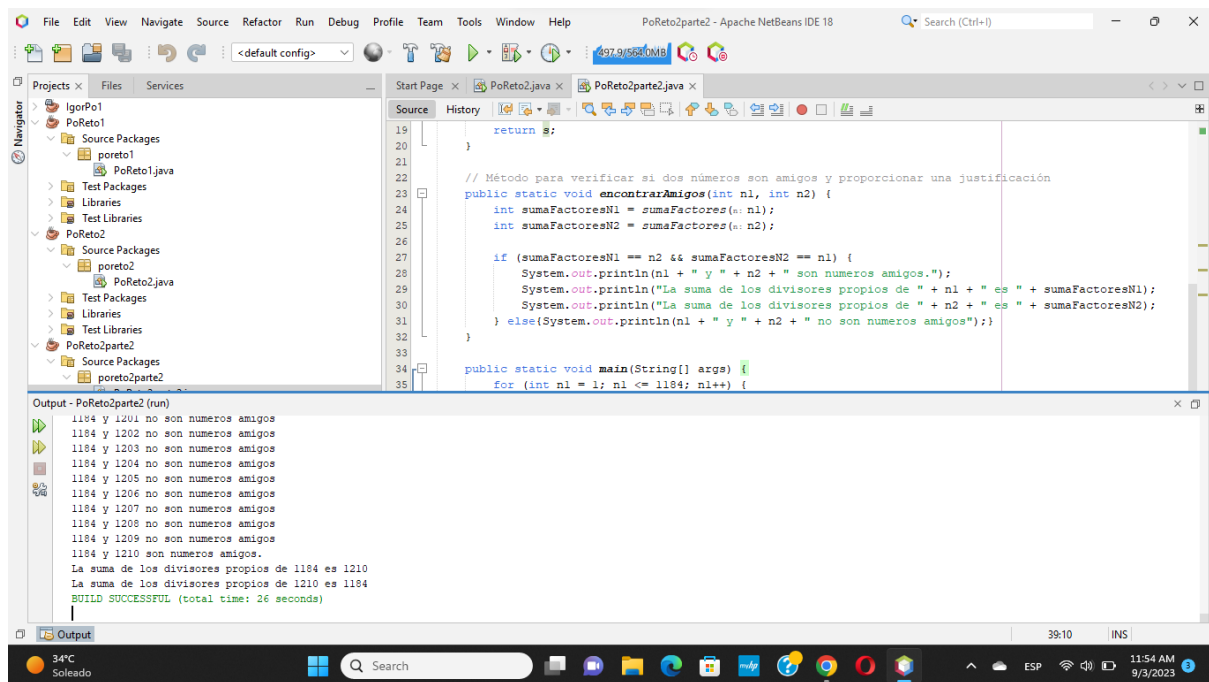
public static int problema (int value)
{
    int addition = 0;
    for (int i = 1; i <= value / 2; i++)
    {
        if (value % i == 0)
        {
            addition += i;
        }
    }
    return addition;
}

public static boolean amigos (int value1, int value2)

Output - PoReTo2 (run)
run:
Dame el primer valor: 220
Dame el segundo valor: 284
Los valores de 220 y 284 son numeros amigos.
BUILD SUCCESSFUL (total time: 6 seconds)
```

3. Ciclo para explorar números: Implementar un ciclo para iterar a través de un rango de números (por ejemplo, del 1 al 1000) y dentro del ciclo, utilizar las funciones definidas anteriormente para verificar si un número tiene un número amigo dentro del mismo rango.

4. Resultados: Después de completar el ciclo, mostrar en pantalla si son números amigos o no, con la justificación.



The screenshot shows the Apache NetBeans IDE with a project named 'PoReto2parte2'. The 'Source' tab displays the following Java code:

```
19     }
20     return s;
21
22     // Método para verificar si dos números son amigos y proporcionar una justificación
23     public static void encontrarAmigos(int n1, int n2) {
24         int sumaFactoresN1 = sumaFactores(n1);
25         int sumaFactoresN2 = sumaFactores(n2);
26
27         if (sumaFactoresN1 == n2 && sumaFactoresN2 == n1) {
28             System.out.println(n1 + " y " + n2 + " son numeros amigos.");
29             System.out.println("La suma de los divisores propios de " + n1 + " es " + sumaFactoresN1);
30             System.out.println("La suma de los divisores propios de " + n2 + " es " + sumaFactoresN2);
31         } else {System.out.println(n1 + " y " + n2 + " no son numeros amigos");}
32     }
33
34     public static void main(String[] args) {
35         for (int n1 = 1; n1 <= 1184; n1++) {
```

The 'Output' tab shows the following results:

```
1184 y 1201 no son numeros amigos
1184 y 1202 no son numeros amigos
1184 y 1203 no son numeros amigos
1184 y 1204 no son numeros amigos
1184 y 1205 no son numeros amigos
1184 y 1206 no son numeros amigos
1184 y 1207 no son numeros amigos
1184 y 1208 no son numeros amigos
1184 y 1209 no son numeros amigos
1184 y 1210 son numeros amigos.
La suma de los divisores propios de 1184 es 1210
La suma de los divisores propios de 1210 es 1184
BUILD SUCCESSFUL (total time: 26 seconds)
```

5. Reflexión: Discutir sobre cómo implementaron las funciones, cómo utilizaron los ciclos y cómo manejaron los datos en su programa.

Reflexion Parte 1 (Determinar si las variables son números amigos): Lo que hice para determinar si dos variables son números amigos, fue crear 2 funciones, la primera es del tipo Integer y la segunda del tipo booleano, la primera función se llama "problema", en esta función declare como parámetro una variable int llamada "value", luego declare otra variable del mismo tipo llamada "addition" que es igual a 0, después hice un ciclo For donde "i" es igual a 1 y mientras fuera menor o igual a la mitad del parámetro, el ciclo continúa. Dentro del ciclo también declare una condicional donde si el residuo del parámetro fuera igual a cero, entonces entonces "i" es un divisor propio del parámetro, entonces se sumaría a la variable "addition", y al final declare que me regresa como dato de salida el parámetro que había declarado en la función. En la función del tipo Boolean, declare como parámetros de entrada las variables que use en el Main como datos de entrada, y dentro de la función mientras el algoritmo fuera verdadero, o sea que ambos números fuesen números amigos, me regresaría como valor "True". ya en el main del código declare los datos de entrada y después puse un condicional "IF" y "ELSE", en el algoritmo de

"IF" puse prácticamente que mientras las instrucciones fuesen "True" entonces que se imprimiera como dato de salida que las variables si son números amigos, de lo contrario en el "ELSE" imprimiera que las variables no son números amigos

Reflexión Parte 2 (Ciclo para encontrar numeros amigos): Primero usé un ciclo for que iba a tomar los valores de n, y para cada valor de n lo iba a sacar el valor residual de si n es dividida entre i mientras i fure incremental (i++, osea que se le sume 1), y el residual de n sobre i es igual a 0, todo valor de i donde el residual fuera 0 iba a ser parte de la sumatoria que iba a dar el valor de una variable llamada s, esta función se llamaría sumaFactores. así para obtener la sumatoria de los factores, o divisores propios de n, luego cree una nueva función donde asigne 2 variables que tomaron los valores de las sumatorias de los divisores de n1 y n2 respectivamente usando la función sumaFactores, las cuales se llamarían sumaFactoresN1 y sumaFactoresN2, dentro de esta función tomaremos los valores de n1, n2 y los comparemos con los valores de sumaFactoresN1 y sumaFactoresN2, utilizando un if, dentro del cual compararia sumaFactoresN1 con n2 y sumaFactoresN2 con n1, si sumaFactoresN2 es igual a n1 y sumaFactoresN1 es igual a n2 nos regresara un mensaje que diría que son números amigos y que la razón de esto es que la suma de los divisores del número 1 es igual al número 2 y que la suma de los divisores del número 2 es igual al número 1, y un else que diría que no son números amigos, para finalizar en el main coloque un ciclo for anidado con valores incrementales de n1 y n2 donde repetiría la función encontrar amigos con n1 y n2 donde n1 y n2 son incrementales hasta que n1 llegue a 1184 y n2 llegue hasta 1210, la razón porque elegí estos dos numero para terminar cada ciclo for es porque que ría que los ultimos valores que me mostrara la funcion fueran numeros amigossya que no hay ningun par de numeros que fueran amigos entre 1000 y 2000 aparte de 1184 y 1210 y no queria que mi programa fuera demasiado grande, despues de todo no fue solicitado el rango del ciclo nunca en el rango. y si quisiera podria quitar el else de la funcion encontrar amigos para que el programa me muestre únicamente los números amigos.