Programación

Métodos

¿Qué son los métodos? En palabras simples, un método es una función la cual está relacionada directamente con un objeto, y cumple una única función. Además, se recomienda que sea un código corto. Un objeto siempre llamará al método mediante la declaración:

objeto.Metodo( );

Entre las características de un método se encuentran:

que puede o no tener parámetros;
que puede o no retornar algo;
que puede o no necesitar de argumentos.

Un método siempre se definirá como “public/private “valor de retorno” “nombreDelMetodo” “parámetros”.

Un método que fuera público, que no tuviera ni un valor de retorno ni un parámetro se escribiría:

```
public void nombreMetodo ( ) {

    //cuerpo del método

}
```

Un método público, que retorna un valor de tipo float, pero no tiene parámetros, se escribiría:

```
public float nombreMetodo ( ) {

    //cuerpo del método
}
```

Un método privado, que retorna un valor de tipo int y que tiene parámetros: “num1, num2” de tipo int, se escribiría:

```
private int nombreMetodo (int num1, int num2) {

    //cuerpo del método
}
```

Para demostrar que ha aprendido, cree los métodos: sumar, restar, multiplicar y dividir en intelliJ, encargados de sumar 2 números, los cuales deben ir en los parámetros.

```
//Metodos

//Metodo para sumar dos numeros

public float sumar(float numero1, float numero2) {
```

```
    float suma = numero1 + numero2;
}

//Metodo para restar dos numeros

public float restar(float numero1, float numero2) {

    float resta = numero1 - numero2;
}

//Metodo para multiplicar dos numeros

public float multiplicar(float numero1, float numero2) {

    float multiplicacion = numero1 * numero2;
}

//Metodo para dividir dos numeros

public float dividir(float numero1, float numero2) {

    float division = numero1 / numero2;
}
```

Ahora, ya hemos visto lo que son los métodos y los parámetros. ¿Pero qué hay de los argumentos? Para poder explicar lo que es un argumento, tendremos que explicar los dos tipos de variables existentes en una clase: las variables “globales” y las “locales”.

Variables globales

Las variables globales (conocidas como atributos) se declaran dentro de la clase, pero fuera del método, dichas variables se pueden llamar en cualquier línea del programa, mientras esté dentro de la clase.

```
//Atributos

float resta;
float suma;
float multiplicacion;
float division;
```

Variables locales

Las variables locales se declaran dentro del método y solo existen dentro del mismo, por lo tanto, no se pueden llamar desde afuera, la única forma de poder utilizar el valor de la misma fuera del método es mediante el “return”, declarándolo como valor de return. Por ejemplo:

```
public float sumar(float numero1, float numero2) {

    float suma = numero1 + numero2;
    return suma;
}
```

En dicho ejemplo hemos creado la clase sumar que recibe como parámetros: numero1 y numero2, así mismo, en el cuerpo del método hemos definido la variable de tipo flotante “suma”, la cual es la suma de las variables flotantes (parámetros) numero1 y numero2. Para poder utilizar el valor que adquirimos de suma fuera del método, hemos puesto *return suma;* en el cuerpo y *float* como valor de retorno en la declaración del método.

En base a lo expuesto, os mostramos una pequeña clase llamada Operaciones, la cual tiene los métodos: “sumar, restar, multiplicar y dividir”. El primer ejemplo será con variables globales, sin retorno y el segundo con variables locales y retorno.

Ejemplo 1:

```
public class Operaciones {

    //Atributosma;

    float resta;
    float su
    float multiplicacion;
    float division;

    //Metodos

    //Metodo para sumar dos numeros

    public float sumar() {

        suma = numero1 + numero2;
    }

    //Metodo para restar dos numeros

    public float restar() {
```

```
        resta = numero1 - numero2;
    }

    //Metodo para multiplicar dos numeros

    public float multiplicar() {

        float multiplicacion = numero1 * numero2;
    }

    //Metodo para dividir dos numeros

    public float dividir() {

        float division = numero1 / numero2;
    }
```

Ejemplo 2:

```
public class Operaciones {

    //Metodos

    //Metodo para sumar dos numeros

    public float sumar(float numero1, float numero2) {

        float suma = numero1 + numero2;
        return suma;
    }

    //Metodo para restar dos numeros

    public float restar(float numero1, float numero2) {

        float resta = numero1 - numero2;
        return resta;
    }

    //Metodo para multiplicar dos numeros

    public float multiplicar(float numero1, float numero2) {

        float multiplicacion = numero1 * numero2;
        return multiplicacion;
    }
```

```
    //Metodo para dividir dos numeros

    public float dividir(float numero1, float numero2) {

        float division = numero1 / numero2;
        return division;
    }
```

Hemos aprendido como crear métodos, pero seguimos sin mencionar su relación con los objetos. A continuación, se os explicará la relación entre los métodos y los objetos.

Relación Método – Objeto

Hasta el momento, hemos visto que los métodos funcionan igual a una función, ¿no? Pero, como está dicho en su definición, estos se relacionan con el concepto de objeto, ¿por qué? Hasta este punto, como están definidos los métodos, no hacen nada más que almacenar un valor que, en el primer caso lo almacena una variable global y en el segundo caso retorna a algún lado, el cual todavía no hemos mencionado. Para no hacer el cuento largo, lo explicaremos todo en base al segundo ejemplo y luego lo expandiremos al primero. El valor de retorno del segundo objeto se va al objeto, el cual, previamente, le entregó –en este caso– dos números como argumentos. Dicho objeto lo crearemos en una nueva clase dentro del proyecto en el que está almacenada la clase “Operaciones”, dado que es el arranque, le llamaremos clase “Main”. En el punto siguiente ahondaremos más a fondo la relación de Main y Operaciones.

Main y Operaciones

Como dijimos antes, la clase Main será quien le proporcione los argumentos a la clase Operaciones. Para esto, la clase Main debe tener definidos los argumentos que, en el ejemplo dos, eran dos números. Supongamos que dichos números son llamados “num1 y num2”, ambos de tipo Float, tal cual como se muestra en el siguiente ejemplo, donde nosotros pondremos los números:

```
public class Main {
    public static void main (String [] args){
        float num1 = 10;
        float num2 = 5;
    }
}
```

o en el siguiente ejemplo, donde haremos que el usuario declare los números:

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

```
        float num1 = Integer.parseInt(JOptionPane.showInputDialog("Digite
un numero"));
        float num2 = Integer.parseInt(JOptionPane.showInputDialog("Digite
un numero"));
    }
}
```

Ahora, ¿cómo hacemos para conectar las clases? Utilizando lo que dijimos al comienzo de este documento, la llamada del método mediante "*objeto.Metodo*". Para esto crearemos, dentro de la clase Main, un objeto cualquiera, supongamos que se llama "op" (de operaciones), para crear el objeto y que este se relacione a nuestra clase Operaciones (la cual, para el ejemplo, se encuentra dentro del paquete operaciones), declaramos:

Operaciones.Operaciones op = new Operaciones.Operaciones( );

Ya creado nuestro objeto "op" relacionado con la clase Operaciones, procedemos a llamar los métodos de la clase Operaciones y a darle como argumentos "num1 y num2", además, guardaremos el valor de retorno de "sumar, restar, multiplicar y dividir" en las variables "suma, resta, multiplicación y division" respectivamente, tal y como se muestra en el siguiente ejemplo:

```
public class Main {
    public static void main (String [] args){
        float num1 = Integer.parseInt(JOptionPane.showInputDialog("Digite
un numero"));
        float num2 = Integer.parseInt(JOptionPane.showInputDialog("Digite
un numero"));

        Operaciones.Operaciones op = new Operaciones.Operaciones();

        float suma = op.sumar(num1, num2);
        float resta = op.restar(num1, num2);
        float multiplicacion = op.multiplicar(num1, num2);
        float division = op.dividir(num1, num2);

    }
}
```

En el ejemplo 1 se haría exactamente lo mismo, solo que no habría que crear una variable que almacene un valor –ya que no hay retorno–, ni tampoco habría que darles argumentos, al utilizar las variables globales, pero sí habría que agregar los números dentro de las variables globales, tal y como se muestra en el siguiente ejemplo:

```
public class Operaciones {

    //Atributos
```

```java
    float resta;
    float suma;
    float multiplicacion;
    float division;
    float numero1 = 5;
    float numero2 = 10;

    //Metodos

    //Metodo para sumar dos numeros

    public void sumar() {

        suma = numero1 + numero2;
    }

    //Metodo para restar dos numeros

    public void restar() {

        float resta = numero1 - numero2;
    }

    //Metodo para multiplicar dos numeros

    public void multiplicar() {

        float multiplicacion = numero1 * numero2;
    }

    //Metodo para dividir dos numeros

    public void dividir() {

        float division = numero1 / numero2;
    }

    //Metodo para mostrar resultado

    public void mostrarResultado() {

        System.out.println("La suma es:" + suma);
        System.out.println("La resta es:" + resta);
        System.out.println("La multiplicacion es:" + multiplicacion);
        System.out.println("La division es:" + division);
    }
}
```

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

        Operaciones.Operaciones op = new Operaciones.Operaciones();

        op.sumar();
        op.restar();
        op.multiplicar();
        op.dividir();
    }
}
```

Por último, se os mostrará un ejemplo ya más trabajado, el cual muestra en pantalla el resultado de los cuatro métodos:

```
public class Main {
    public static void main (String [] args){
        float num1 = Integer.parseInt(JOptionPane.showInputDialog("Digite
un numero"));
        float num2 = Integer.parseInt(JOptionPane.showInputDialog("Digite
un numero"));
        Operaciones.Operaciones op = new Operaciones.Operaciones();

        float suma = op.sumar(num1, num2);
        float resta = op.restar(num1, num2);
        float multiplicar = op.multiplicar(num1, num2);
        float dividir = op.dividir(num1, num2);
        op.mostrarResultado(suma, resta, multiplicar, dividir);

    }
}
```

```
public class Operaciones {

    //Atributos

    //float resta;
    //float suma;
    //float multiplicacion;
    //float division;

    //Metodos

    //Metodo para sumar dos numeros

    public float sumar(float numero1, float numero2) {
```

```
        float suma = numero1 + numero2;
        return suma;
    }

    //Metodo para restar dos numeros

    public float restar(float numero1, float numero2) {

        float resta = numero1 - numero2;
        return resta;
    }

    //Metodo para multiplicar dos numeros

    public float multiplicar(float numero1, float numero2) {

        float multiplicacion = numero1 * numero2;
        return multiplicacion;
    }

    //Metodo para dividir dos numeros

    public float dividir(float numero1, float numero2) {

        float division = numero1 / numero2;
        return division;
    }

    //Metodo para mostrar resultado

    public void mostrarResultado(float suma, float resta, float multipli-
cacion, float division) {

        System.out.println("La suma es:" + suma);
        System.out.println("La resta es:" + resta);
        System.out.println("La multiplicacion es:" + multiplicacion);
        System.out.println("La division es:" + division);
    }
}
```

Como último dato: las variables globales se ven afectadas cuando son llamadas en el método y su valor cambia.