

Asignatura Introducción a la Programación

Ingeniería Informática

Guía de Aprendizaje

Tema : Programación funcional.

Unidad didáctica: Introducción a las funciones y métodos de clase.

1. Objetivos de la unidad didáctica:

- Caracterizar un método de clase.
- Identificar los principales elementos que componen un método de clase.

2. Introducción a la unidad didáctica:

La mayoría de los programas de cómputo que resuelven los problemas reales son mucho más extensos que los programas que hemos desarrollado hasta ahora. La experiencia ha demostrado que la mejor manera de desarrollar y mantener un programa extenso es construirlo a partir de pequeñas piezas sencillas, o **módulos**. A esta técnica se le llama **divide y vencerás**.

Existen tres tipos de módulos en Java: métodos, clases y paquetes. Para escribir programas en Java, se combinan los nuevos métodos y clases que usted escribe con los métodos y clases predefinidos, que están disponibles en la **Interfaz de Programación de Aplicaciones de Java** (también conocida como la API de Java o biblioteca de clases de Java) y en diversas bibliotecas de clases. Por lo general, las clases relacionadas están agrupadas en paquetes, de manera que se pueden importar a los programas y reutilizarse.

En esta guía abordaremos sobre en el módulo mas pequeño de los tres anteriores: lo métodos.

Los métodos (también conocidos como funciones o procedimientos en otros lenguajes) permiten al programador dividir un programa en módulos, por medio de la separación de sus tareas en unidades autónomas. Usted ha declarado métodos en todos los programas que ha escrito; a estos métodos se les conoce algunas veces como métodos declarados por el programador. Las instrucciones en los cuerpos de los métodos se escriben sólo una vez, y se reutilizan tal vez desde varias ubicaciones en un programa; además, están ocultas de otros métodos.

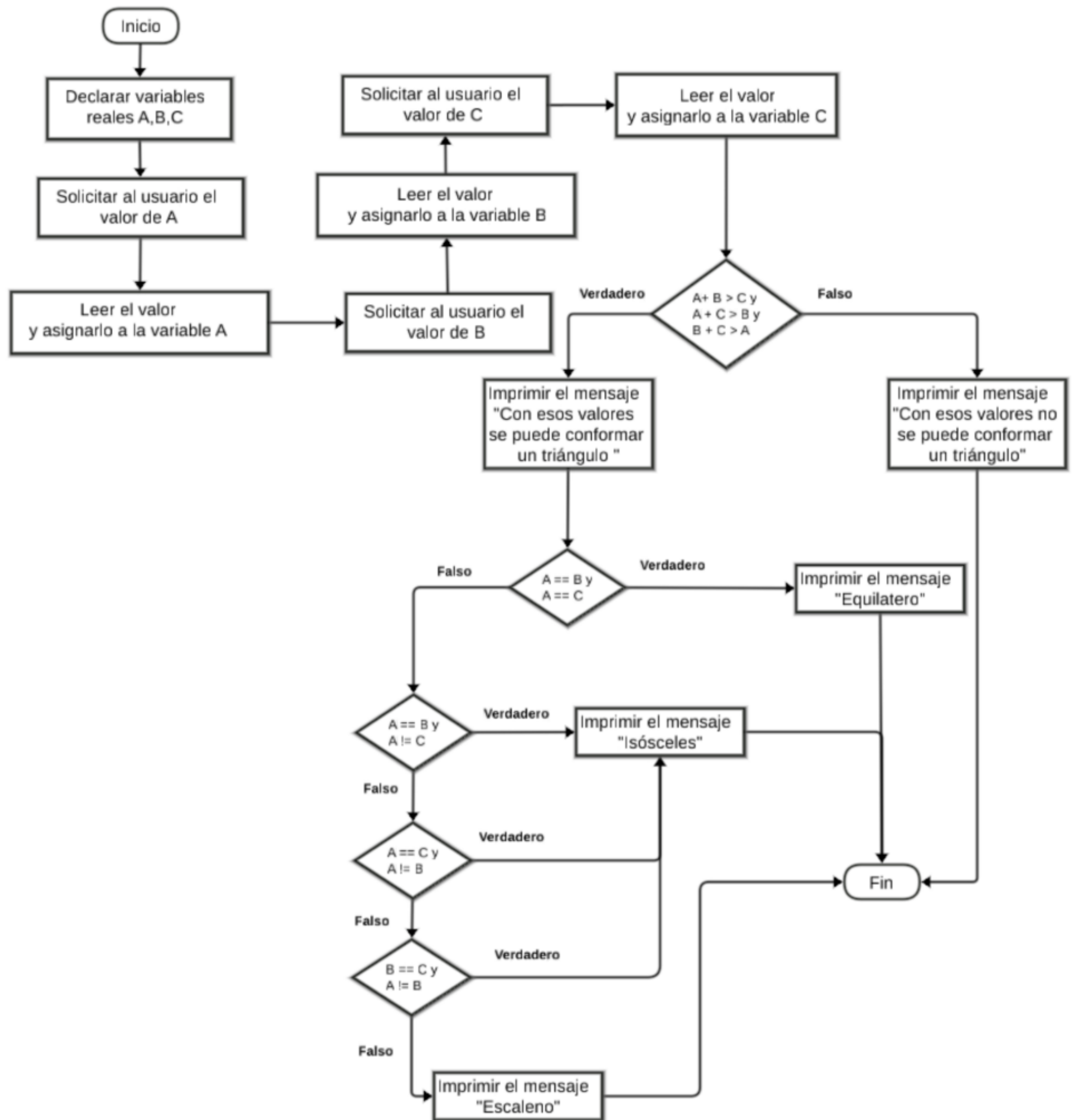
Los métodos son declarados siempre dentro de una clase. En el lenguaje de programación Java existe dos tipos de métodos: los métodos de clase y los métodos de un objeto de clase. En esta guía y en las posteriores que aborden este tema trabajaremos con los métodos de clase. Vamos a partir de un ejemplo para entender lo que se ha mencionado hasta ahora para eso vamos a partir del ejercicio 2 dejado en una guía de aprendizaje evaluativa anterior el

mismo planteaba lo siguiente:

Diseñe un algoritmo que permita determinar si con tres números se puede o no conformar un triángulo cuyos lados tendrían como longitudes dichos valores. En caso que se pueda conformar el triángulo el algoritmo debe ser capaz de imprimir la clasificación de dicho triángulo atendiendo a las longitudes de sus lados. Tenga en cuenta que el algoritmo debe solicitar o pedir los valores por pantalla. Utilice una de las técnicas para representar algoritmo que sea mas de su agrado. Transcriba dicho algoritmo a lenguaje de programación de Java.

Análisis del problema

Bueno es evidente que mi programa debe ser capaz de solicitar al usuario 3 números y ver si con dichos valores se cumple la desigualdad triangular y en caso de cumplir tendría que clasificar el triángulo que se formaría acorde a los valores siendo esos valores las longitudes de los lados del supuesto triángulo. Analizado esto lo primero será nuestro realizar nuestro diagrama para representar el algoritmo el mismo queda de la siguiente manera.



Una vez realizado el diagrama de nuestro algoritmo vamos transcribirlo a lenguaje de programación Java.

```
import java.util.Scanner;
```

```
public class Main{
```

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

```
        double A,B,C;
```

```
        Scanner input = new Scanner(System.in);
```

```
        System.out.print("Entre el valor de A: ");
```

```
        A = input.nextDouble();
```

```
        System.out.print("Entre el valor de B: ");
```

```

        B = input.nextDouble();

        System.out.print("Entre el valor de C: ");

        C = input.nextDouble();

        if (A+B>C && A+C>B && B+C>A){

            System.out.print("Con esos valores se puede conformar un triangulo ");

            if (A==B && A==C){

                System.out.print("Equilatero");

            } else if ( (A==B && A!=C) || (A==C && A!=B) || (C==B && A!=C)){

                System.out.print("Isosceles");

            } else {

                System.out.print("Escaleno");

            }

        } else {

            System.out.println("Con esos valores no se puede conformar un triangulo");

        }

    }
}

```

Lo primero a notar es que la transcripción a Java no fue al 100% (esto puede pasar perfectamente ya sea porque una varias ideas en nuestro diseño de algoritmo se puede realizar en una sola sentencia o bien a la inversa) sino que hizo una mejora en cuanto al caso que se pueda conformar un triángulo del tipo isósceles. Ahora pensemos que tenemos otros problemas donde bien tenemos que comprobar la desigualdad triangular o saber el tipo de triángulo acorde a sus lados. Lo primero que se nos viene a la mente es copiar y pegar pero esta acción nos puede resultar un poco engorrosa por el hecho que tenemos que adaptar o acomodar fragmentos de código dentro de otro. Bueno y si en vez ajustar código creamos un fragmento de código un tanto autónomo encargado únicamente de realizar una acción o tarea y lo utilizó dentro de mi programa cuando haga falta. Vamos a aplicar esto a nuestro código anterior para ver como nos queda.

```

import java.util.Scanner;

public class Main{

    public static boolean desigualdadTriangular(double ladoA, double ladoB, double ladoC){

        if (ladoA+ladoB>ladoC && ladoA+ladoC>ladoB && ladoB+ladoC>ladoA){

            return true;

        } else {

            return false;

        }

    }

}

```

```

public static String clasificarTrianguloPorLados(double ladoA, double ladoB, double ladoC){

    if (ladoA==ladoB && ladoA==ladoC){

        return "Equilatero";

    } else if ( (ladoA==ladoB && ladoA!=ladoC) ||

        (ladoA==ladoC && ladoA!=ladoB) ||

        (ladoC==ladoB && ladoA!=ladoC)){

        return "Isosceles";

    } else {

        return "Escaleno";

    }

}

public static void main(String[] args){

    double A,B,C,x1=0,x2=0;

    Scanner input = new Scanner(System.in);

    System.out.print("Entre el valor de A: ");

    A = input.nextDouble();

    System.out.print("Entre el valor de B: ");

    B = input.nextDouble();

    System.out.print("Entre el valor de C: ");

    C = input.nextDouble();

    if (desigualdadTriangular(A,B,C) == true){

        System.out.print("Con esos valores se puede conformar un triangulo ");

        System.out.println(clasificarTrianguloPorLados(A,B,C));

    } else {

        System.out.println("Con esos valores no se puede conformar un triangulo");

    }

}

}

```

Es evidente que el código del método *main* se simplificó notablemente. Claro va existir detractores que van a criticar el aumento la cantidad de código del programa de forma general. Pero vamos analizar que ya contamos con dos funciones como son *desigualdadTriangular* y *clasificarTrianguloPorLados* las cuales realizan acción o tarea en específico y como están encapsuladas dentro de un método son altamente exportable hacia otros programas donde lo debemos copiarla tal y como están y usarla como se hizo en el método *main* . Con este último código vamos a trabajar en las actividades de auto-aprendizajes.

3. Orientaciones para el estudio:

1. Le sugerimos que para realizar el estudio de una materia debe elegir un lugar agradable, limpio, ventilado, cómodo, iluminado y si necesita recursos tecnológicos deben estar disponibles.
2. Cree hábitos de estudio sistemático; para esto:
 - Planifique su tiempo y no deje para mañana lo que pueda hacer hoy; una buena planificación hace manejable las responsabilidades diarias que tenemos para con la vida, permite cumplir con todas las tareas programadas y así evita agobios de última hora.
 - Calendarice las fechas más importantes de estudio y entrega de tareas. Localice con antelación los materiales que necesita para realizar el estudio individual o colaborativo.
3. Utilice técnicas de estudio:
 - Elija un entorno de estudio que resulte agradable y sin elementos que lo que distraigan
 - Estudie activamente, para esto lea en voz alta, tome notas, elabore esquemas o mapas conceptuales, realice resúmenes.
 - Reflexione sobre lo que va aprendiendo, para esto relacione lo nuevo con lo anterior o conocido, asegúrese de que entiende y es capaz de aplicar lo que está aprendiendo antes de pasar adelante.
4. Interactúe con los materiales de estudio en tres fases:
 - Fase de aproximación: Revise el objetivo de la unidad y después la acción o acciones a lograr; busque los materiales sugeridos para desarrollar la tarea, verifique cuál es el tiempo de que dispone.
 - Fase de lectura profunda: aproxímese al material a través de una lectura ligera, poniendo especial interés en los títulos y subtítulos. Trate de relacionar lo que va leyendo con conocimientos adquiridos previamente.
 - Fase de evaluación: Una vez realizada la lectura, intente realizar las actividades de auto-aprendizaje.

4. Requisitos Previos:

Tener conocimientos básicos de computación: los estudiantes matriculados deben conocer los procedimientos básicos para el manejo de PC y/o dispositivos móviles, aplicaciones de ofimática; así como uso de navegadores de páginas Web y el correo electrónico.

5. Actividades de auto-aprendizajes:

1. Cual es la palabra reservada del lenguaje de programación Java define que un método sea

de clase o de un objeto de clase. Identifique dicha palabra en el código presentado en la **Introducción a la unidad didáctica**.

2. Por qué el método *main* se declara como **static** ?.

3. Cual es la palabra reservada del lenguaje de programación Java permite que un método devuelva o retorne un resultado. Identifique dicha palabra en el código presentado en la **Introducción a la unidad didáctica**.

4. En que partes o secciones se divide un método de clase. Identifique dichas partes o secciones en cada método de los presentados en el código presentado en la **Introducción a la unidad didáctica**.

5. En la cabecera de un método que palabra reservada del lenguaje de programación Java se utiliza para indicar que el método no retorna ni devuelve un valor, en caso de que devuelva o retorna algún valor que se debe colocar. Utilice el código presentado en la **Introducción a la unidad didáctica** de apoyo.

6. Las variables *ladoA*, *ladoB*, *ladoC* que papel desempeñan dentro de los métodos *desigualdadTriangular* y *clasificarTrianguloPorLados* presentados en el código de la **Introducción a la unidad didáctica**

7. Que palabras reservadas del lenguaje de programación define la visibilidad o nivel de acceso a variables, métodos y clases. Identifique una de esas palabras en el código presentado en la **Introducción a la unidad didáctica** .

6. Actividades de evaluación:

1. Tomando el código presentado en la **Introducción a la unidad didáctica** haga en el las adecuaciones necesarias para que ahora sea un programa que solicite tres valores que representan la amplitud de 3 ángulos en grados sexagesimales y diga si se puede o no formar un triangulo cuyos ángulos tengan esa amplitud en caso de poder formarse que clasifique dicho triángulo basado en la amplitud de sus ángulos.

7. Resumen:

Una razón para dividir un programa en módulos mediante los métodos es la metodología **“divide y vencerás”**, que hace que el desarrollo de programas sea más fácil de administrar, ya que se pueden construir programas a partir de piezas pequeñas y simples. Otra razón es la **reutilización de software** (usar los métodos existentes como bloques de construcción para crear nuevos programas). A menudo se pueden crear programas a partir de métodos estandarizados, en vez de tener que crear código personalizado.

8. Glosario de términos:

Consulte la bibliografía y defina el funcionamiento de las siguientes palabras reservadas:

void:

static:

9. Bibliografía:

Aprenda Java como si estuviera en primero. *Colectivo de autores*. Cap 3 epígrafes 3.1, 3.2, 3.3, 3.4.

Cómo programar en Java. *Deitel, Paul J. Y Harvey M. Deitel*. Cap 6 completo

10. Recursos educativos digitales RED:

11. **Próxima unidad:** Cabeceras de las funciones y métodos de clase.