



## Actividad | 1 | Estructuras de Control

### Lenguajes de Programación IV

Ingeniería en Desarrollo de Software

---



TUTOR: Aaron Iván Salazar Macias

---

ALUMNO: Dadne Alinne Luna Carranza

---

FECHA: 14 de noviembre de 2025

---

## Indice

Introducción.....	3
Investigación.....	4
Interfaz gráfica.....	7
codificación.....	10
Sistema IMC.....	13
Conclusiones.....	15
Referencia.....	16

## Introducción

El desarrollo de sistemas orientados a la solución de problemas reales se ha convertido en una habilidad fundamental dentro del campo de la programación. En este sentido, la presente actividad tiene como propósito aplicar los principios de la programación orientada a objetos para diseñar y construir un sistema capaz de calcular el Índice de Masa Corporal (IMC) de los pacientes de un hospital ubicado en la Ciudad de México. El IMC es una medida ampliamente utilizada en el ámbito de la salud, ya que permite evaluar la relación entre el peso y la estatura de una persona, proporcionando una referencia general sobre su estado nutricional. Debido a que México es uno de los países con mayores índices de sobrepeso y obesidad, contar con herramientas que faciliten la clasificación del IMC resulta de gran utilidad para la atención médica y la toma de decisiones preventivas.

A través del uso del lenguaje de programación Java 8 y del entorno de desarrollo NetBeans, esta actividad permite reforzar conceptos esenciales de la programación orientada a objetos, como clases, objetos, métodos y encapsulamiento. Además, se hace uso de estructuras de control como condicionales y ciclos para validar datos y generar resultados precisos. El objetivo es que el estudiante no solo comprenda los fundamentos teóricos de estas herramientas, sino que también sea capaz de implementarlas en un proyecto práctico, funcional y documentado correctamente bajo el formato APA. Esta actividad representa un acercamiento integral al proceso de análisis, diseño, construcción y prueba de un sistema computacional sencillo pero significativo dentro del área de la salud.

## Investigación de la Actividad

### **Clase**

Una clase es una plantilla o modelo que permite definir las características y comportamientos de un conjunto de objetos. Contiene atributos y métodos que representan la estructura y las operaciones que dichos objetos pueden realizar. En la programación orientada a objetos, la clase funciona como el plano a partir del cual se crean las instancias.

### **Objeto**

Un objeto es una instancia de una clase. Representa un elemento concreto que posee atributos propios y puede ejecutar los métodos definidos en su clase. Cada objeto puede tener valores distintos en sus atributos, aunque comparta la misma estructura.

### **Encapsulamiento**

El encapsulamiento consiste en proteger los datos de un objeto, permitiendo que solo sean modificados o consultados a través de métodos específicos. Se logra utilizando modificadores de acceso como `private` o `public`. Su propósito es evitar cambios no controlados y mejorar la seguridad del código.

### **Herencia**

La herencia es un mecanismo mediante el cual una clase puede derivarse de otra para reutilizar atributos y métodos ya existentes. La clase que hereda se llama subclase, y la clase de la cual se hereda se denomina superclase. Facilita la reutilización de código y la creación de jerarquías.

## Investigación de la Actividad

### **Abstracción**

La abstracción permite enfocarse únicamente en las características esenciales de un objeto, ocultando los detalles irrelevantes o complejos. Este proceso simplifica el diseño del sistema y facilita la comprensión de los elementos que lo componen.

### **Polimorfismo**

El polimorfismo permite que un mismo método pueda comportarse de manera diferente según el objeto o la situación en la que se utilice. Esto permite que distintas clases respondan de manera propia a un mismo mensaje o acción, mejorando la flexibilidad del programa.

### **Constructores**

Un constructor es un método especial que se ejecuta automáticamente al crear un objeto. Su función principal es inicializar los atributos del objeto. Los constructores tienen el mismo nombre que la clase y no devuelven ningún tipo de dato.

### **Eventos**

Los eventos son acciones o sucesos que ocurren durante la ejecución de un programa y que pueden ser detectados por el sistema, como hacer clic en un botón o introducir datos en un campo. Permiten que el programa responda de manera interactiva a las acciones del usuario.

## Investigación de la Actividad

### **Atributos y Propiedades**

Los atributos son variables internas de una clase que almacenan el estado o información del objeto.

Las propiedades, en lenguajes que las utilizan, funcionan como una forma controlada de acceder o modificar dichos atributos mediante métodos get y set.

### **Métodos**

Los métodos son funciones definidas dentro de una clase que describen las acciones o comportamientos que un objeto puede realizar. Permiten manipular los atributos, procesar datos y ejecutar tareas específicas asociadas al objeto.

## Interfaz grafica

En la imagen se muestra la interfaz gráfica del sistema desarrollado para el cálculo del Índice de Masa Corporal (IMC). Esta ventana representa la pantalla principal del programa y fue diseñada utilizando los componentes de Java Swing dentro del entorno de desarrollo NetBeans. La interfaz presenta un diseño sencillo, intuitivo y orientado a facilitar la captura de datos por parte del usuario.

En la parte superior se encuentra el título “IMC HOSPITAL”, el cual indica la finalidad de la aplicación. Debajo del encabezado se incluyen dos etiquetas descriptivas que solicitan los datos necesarios para el cálculo: Peso en kilogramos y Estatura en metros. Frente a cada etiqueta se ubican campos de texto donde el usuario puede ingresar sus valores correspondientes.

En la sección inferior derecha se encuentra el botón “Aceptar”, el cual desencadena el proceso de cálculo cuando el usuario lo presiona. Al presionar este botón, el programa recupera la información introducida, calcula el IMC y muestra el resultado mediante una ventana emergente (JOptionPane).



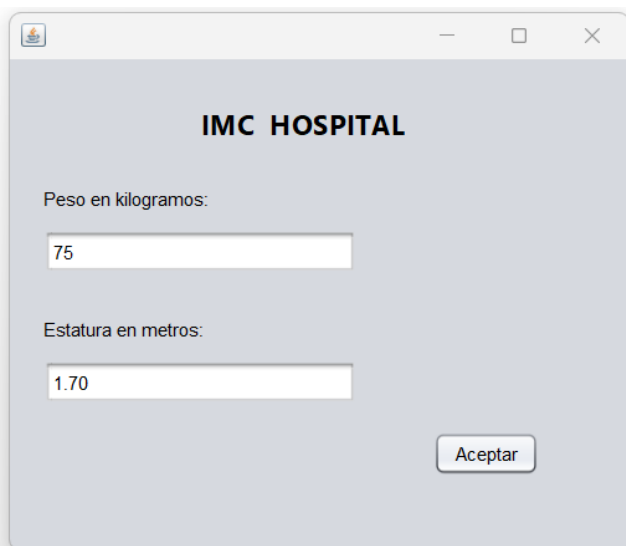
The image shows a Java Swing window titled "IMC HOSPITAL". The window has a light gray background and a standard Windows-style title bar with a minimize button, a maximize button, and a close button. The main content area contains the following elements:

- The title "IMC HOSPITAL" is centered at the top in a bold, black, sans-serif font.
- Below the title, the text "Peso en kilogramos:" is displayed in a black, sans-serif font.
- Underneath the weight label is a white text input field with a thin gray border.
- Below the weight input field, the text "Estatura en metros:" is displayed in a black, sans-serif font.
- Underneath the height label is another white text input field with a thin gray border.
- In the bottom right corner of the window, there is a button labeled "Aceptar" in a black, sans-serif font. The button has a light gray background and a thin gray border.

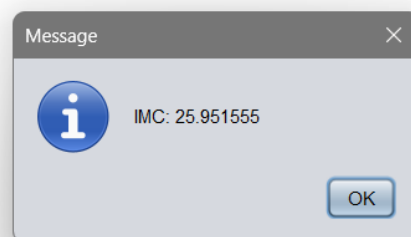
## Interfaz grafica

En esta imagen se observa el funcionamiento del sistema al ingresar valores reales en los campos de la interfaz. El usuario ha introducido un peso de 75 kilogramos y una estatura de 1.70 metros. Después de presionar el botón “Aceptar”, el programa realiza el cálculo del Índice de Masa Corporal aplicando la fórmula correspondiente.

El sistema muestra una ventana emergente (JOptionPane) con el resultado numérico del IMC, en este caso 25.951555, lo que confirma que el programa efectivamente está procesando los datos ingresados y ejecutando la operación matemática de forma correcta. Esta ventana permite al usuario conocer de inmediato el valor obtenido antes de recibir la clasificación nutricional.



The screenshot shows a window titled "IMC HOSPITAL". It contains two input fields: "Peso en kilogramos:" with the value "75" and "Estatura en metros:" with the value "1.70". Below these fields is a button labeled "Aceptar".



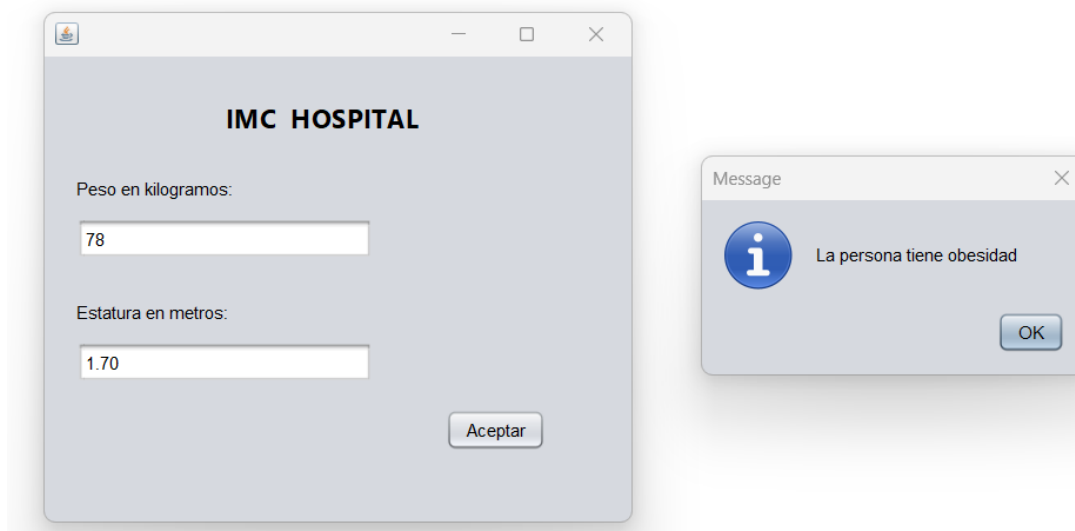


## Interfaz grafica

En esta captura se muestra la segunda parte del proceso del sistema, correspondiente a la clasificación del Índice de Masa Corporal. Después de ingresar un peso de 75 kg y una estatura de 1.70 m, y una vez calculado el IMC, el programa evalúa el resultado numérico mediante una serie de estructuras condicionales.

Como se aprecia en la imagen, el sistema despliega una ventana emergente (JOptionPane) con el mensaje “La persona tiene obesidad”, lo que indica que el valor calculado previamente cae dentro del rango de obesidad según los parámetros establecidos. Esta respuesta confirma que la lógica interna de comparación funciona correctamente y que el sistema es capaz de identificar la categoría correspondiente de forma automatizada.

Esta evidencia demuestra que el programa no solo calcula el IMC, sino que también interpreta los resultados y comunica al usuario la condición nutricional asociada. Con ello, se cumple uno de los objetivos principales de la actividad: ofrecer un sistema funcional y claro para la evaluación del IMC dentro de un entorno hospitalario.



## Codificación

### Lectura de datos de la interfaz

```
99  
100 String peso = jtfPeso.getText();  
101 String estatura = jtfEstatura.getText();  
102
```

En esta parte del código, el sistema obtiene la información que el usuario ingresó en la interfaz gráfica.

- `jtfPeso` corresponde al campo de texto ubicado debajo de la etiqueta “Peso en kilogramos”.
- `jtfEstatura` corresponde al campo de texto debajo de “Estatura en metros”.

El método `.getText()` permite recuperar el valor como una cadena de texto. Esta sección está directamente enlazada con la captura donde se muestran los campos de entrada de la interfaz.

### Conversión de datos y preparación para el cálculo

```
102  
103 float p = Float.parseFloat(peso);  
104 float e = Float.parseFloat(estatura);
```

Esta sección convierte las cadenas de texto obtenidas desde la interfaz en valores numéricos de tipo `float`. Este paso es indispensable porque el cálculo del IMC requiere valores numéricos.

- `p` almacena el peso ingresado.
- `e` almacena la estatura ingresada.

Si el usuario ingresa letras o deja los campos vacíos, aquí se produciría un error que será capturado posteriormente por el `catch`.

## Codificación

### Cálculo del IMC

```

105
106
107
108
float imc = p / (e*e);

JOptionPane.showMessageDialog(null, "IMC: " + imc);

```

En esta parte se realiza la operación principal:

$$\text{IMC} = \text{peso} / (\text{estatura} \times \text{estatura})$$

El resultado se almacena en la variable imc. Luego se muestra una ventana emergente (JOptionPane) con el valor numérico calculado. Esta sección se enlaza con la captura donde aparece el cuadro emergente "IMC: 25.95...".

### Clasificación del IMC

```

109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
if(imc < 18.5){
    JOptionPane.showMessageDialog(null, "La persona tiene bajo peso");
} else if(imc > 18.5 && imc < 24.5){
    JOptionPane.showMessageDialog(null, "La persona tiene peso normal");
} else if(imc >= 25 && imc <= 29.9){
    JOptionPane.showMessageDialog(null, "La persona tiene obesidad ");
} else if(imc >= 30 && imc <= 34.9){
    JOptionPane.showMessageDialog(null, "La persona tiene obesidad I");
} else if(imc >= 35 && imc <= 39.9){
    JOptionPane.showMessageDialog(null, "La persona tiene obesidad II");
} else {
    JOptionPane.showMessageDialog(null, "La persona tiene obesidad III");
}

```

Este bloque contiene las estructuras condicionales que determinan la categoría del IMC.

El programa compara el resultado obtenido con los rangos establecidos y, según corresponda, muestra un mensaje indicando:

- Bajo peso
- Peso normal
- Obesidad
- Obesidad grado I
- Obesidad grado II
- Obesidad grado III

Esta parte del código se enlaza con tu captura donde aparece el mensaje "La persona tiene obesidad".

## Codificación

### Manejo de errores (Excepciones)

```
124 |  
    |  
126 | } catch (Exception e) {  
127 |     JOptionPane.showMessageDialog(null, e);  
    | }
```

Esta sección evita que el programa se cierre si el usuario ingresa datos incorrectos, como letras o símbolos.

El bloque **catch** captura la excepción y muestra un mensaje con el error detectado. Es fundamental para la estabilidad del sistema durante la ejecución.

### Limpieza de los campos de texto

```
128 |  
129 | jtfPeso.setText("");  
130 | jtfEstatura.setText("");
```

Finalmente, el sistema limpia automáticamente los campos después de cada cálculo. Esto permite que el usuario realice una nueva consulta sin borrarlos manualmente.

Esta parte se enlaza con la interfaz vacía después de realizar un cálculo.

## Sistema IMC

El código desarrollado tiene como objetivo calcular el Índice de Masa Corporal (IMC) de un paciente y clasificarlo según los rangos establecidos en la tabla de valoración nutricional. Su funcionamiento combina la interacción con la interfaz gráfica y la aplicación de estructuras de control, todo bajo el paradigma de programación orientada a objetos.

El proceso inicia cuando el usuario ingresa los valores de peso y estatura en los campos de texto de la interfaz. Dichos datos son obtenidos mediante los métodos **getText()** asociados a los componentes **jtfPeso** y **jtfEstatura**, lo cual permite capturar la información que el usuario proporciona. Posteriormente, esas cadenas son convertidas a valores numéricos tipo *float* utilizando **Float.parseFloat()**, lo que prepara la información para realizar el cálculo matemático.

Una vez que los datos han sido validados y convertidos, el programa aplica la fórmula oficial del IMC: peso dividido entre la estatura elevada al cuadrado. El resultado obtenido se muestra inmediatamente al usuario mediante una ventana emergente. Después de mostrar este valor, el programa evalúa el IMC usando una serie de estructuras condicionales **if-else if**. Estas comparan el IMC con los distintos rangos establecidos, determinando si la persona tiene bajo peso, peso normal, sobrepeso u obesidad en sus diferentes grados. La clasificación correspondiente se comunica al usuario mediante otro mensaje emergente.

El bloque **try-catch** mantiene la estabilidad del sistema al capturar errores comunes, como el ingreso

## Sistema IMC

de datos no numéricos o campos vacíos, evitando que el programa se detenga de forma inesperada.

Finalmente, al concluir el proceso, los campos de texto se limpian automáticamente mediante `setText("")`, preparando la interfaz para un nuevo cálculo sin necesidad de reiniciar el sistema.

En conjunto, el código demuestra el uso correcto de estructuras de control, manejo de excepciones, interacción con la interfaz gráfica y la aplicación de conceptos fundamentales de programación orientada a objetos. Además, cumple con los requerimientos del proyecto al proporcionar un sistema funcional, claro y útil para la evaluación del IMC en un entorno hospitalario.

```

98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131

try{

    String peso = jtfPeso.getText();
    String estatura = jtfEstatura.getText();

    float p = Float.parseFloat(peso);
    float e = Float.parseFloat(estatura);

    float imc = p / (e*e);

    JOptionPane.showMessageDialog(null, "IMC: " + imc);

    if(imc < 18.5){
        JOptionPane.showMessageDialog(null, "La persona tiene bajo peso");
    } else if(imc > 18.5 && imc < 24.5){
        JOptionPane.showMessageDialog(null, "La persona tiene peso normal");
    } else if(imc >= 25 && imc <= 29.9){
        JOptionPane.showMessageDialog(null, "La persona tiene obesidad I");
    } else if(imc >= 30 && imc <= 34.9){
        JOptionPane.showMessageDialog(null, "La persona tiene obesidad II");
    } else if(imc >= 35 && imc <= 39.9){
        JOptionPane.showMessageDialog(null, "La persona tiene obesidad III");
    } else {
        JOptionPane.showMessageDialog(null, "La persona tiene obesidad IIII");
    }

} catch (Exception e){
    JOptionPane.showMessageDialog(null, e);
}

jtfPeso.setText("");
jtfEstatura.setText("");

```

## Conclusión

La realización de esta actividad permitió reforzar de manera significativa los conceptos fundamentales de la programación orientada a objetos y su aplicación práctica en el desarrollo de sistemas funcionales. A través de la creación de un programa capaz de calcular el Índice de Masa Corporal (IMC), fue posible comprender cómo elementos como clases, objetos, métodos y encapsulamiento se integran para formar una solución clara, ordenada y eficiente. Asimismo, la implementación de estructuras de control como condicionales permitió generar resultados precisos y adecuados para cada categoría del IMC, demostrando la importancia de la lógica en la programación.

El uso del entorno de desarrollo NetBeans y el lenguaje Java 8 resultó esencial para familiarizarse con herramientas actuales empleadas en la industria, permitiendo experimentar desde el diseño de la interfaz gráfica hasta la ejecución del código. Este proceso también destacó la relevancia de seguir una arquitectura bien definida y aplicar buenas prácticas de programación, lo cual contribuye a crear sistemas más estables, legibles y fáciles de mantener.

En conjunto, esta actividad no solo fortaleció conocimientos técnicos, sino que también permitió comprender el impacto real que un programa sencillo puede tener en contextos como el área de la salud. La experiencia adquirida servirá como base para futuros proyectos más complejos, demostrando que el dominio de los fundamentos es clave para un desarrollo profesional sólido dentro del campo de la programación. Si se continúa practicando y aplicando estos principios, será posible crear sistemas cada vez más completos y útiles.

## Referencias

López Blasco, J. (2023, 27 de octubre). *Introducción a POO en Java: Objetos y clases*.

OpenWebinars. <https://openwebinars.net/blog/introduccion-a-poo-en-java-objetos-y-clases/>