



 [Home](#) /  [Programación II](#)

### **Temas**

clase y objetos, constructores, atributos, propiedades, métodos, modificadores de acceso, getter and setters, diagrama de clase, diagrama de secuencia, windows form, ventanas modales y controles básicos, modelo de aplicación.

### **Referencias**

-  [Diagrama de clases e Implementación en c#](#)
-  [Diagrama de secuencia -nomenclatura](#)

## **Guía 1.1 Abstracción y Encapsulamiento - Clases y objetos, Atributos, Propiedades y métodos**

**fork:** [https://github.com/fernandofilipuzzi-utn/tup\\_prog\\_2\\_2024\\_guia1.1](https://github.com/fernandofilipuzzi-utn/tup_prog_2_2024_guia1.1)

**sol:** [https://github.com/fernandofilipuzzi-dev/tup\\_prog\\_2\\_2024\\_guia1.1](https://github.com/fernandofilipuzzi-dev/tup_prog_2_2024_guia1.1)

<b>I. Ejercicio 1. Concesionaria de motos</b>	<b>2</b>
<b>II. Ejercicio 2. Colectivo</b>	<b>4</b>
<b>III. Ejercicio 3. Sistema de peaje</b>	<b>6</b>

# I. Ejercicio 1. Concesionaria de motos

Una concesionaria de motos necesita crear una pequeña aplicación para tasar una moto en particular. De este tipo de vehículo se conoce su marca, modelo (año de fabricación) y también su valor monetario de fábrica (es decir el precio de venta 0 km). Con estos dos últimos valores se puede estimar un precio estimado como valor actual.

Aplice las siguientes relaciones para obtener el valor actualizado del vehículo según los dos criterios de cálculo:

Depreciación lineal

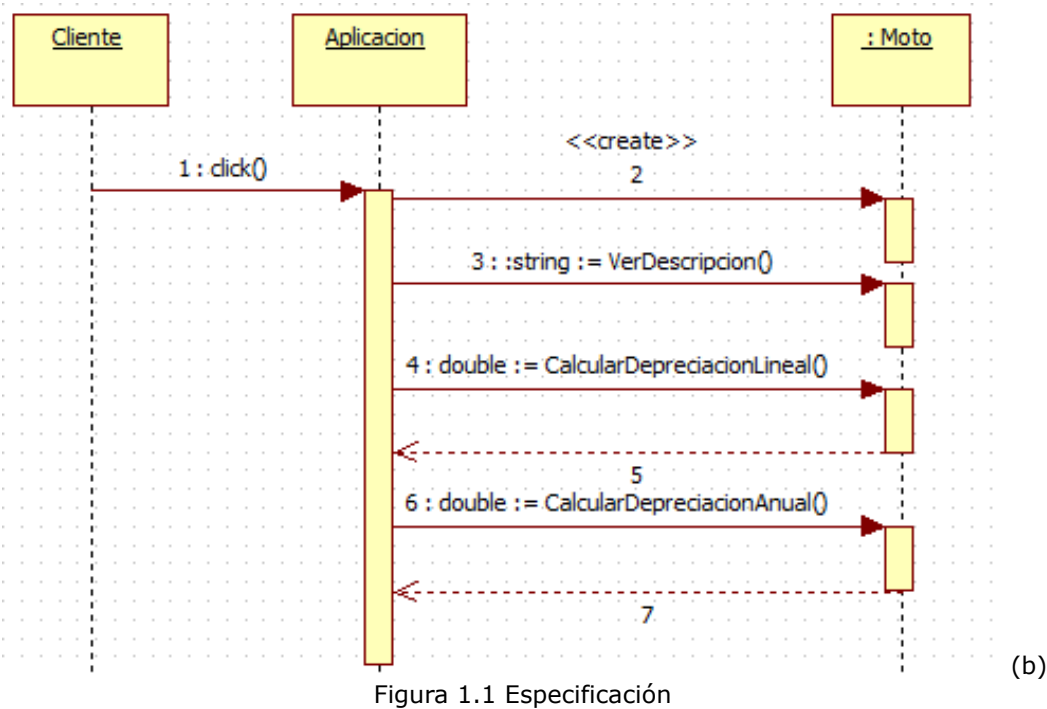
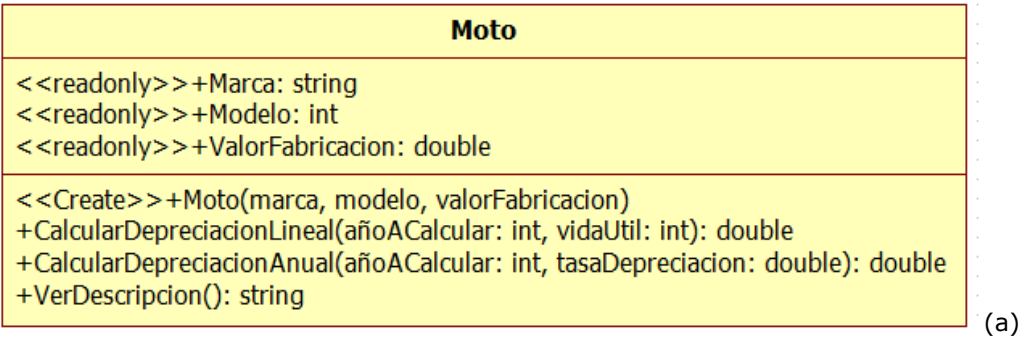
$$"valor\ actualizado" = "valor\ de\ fábrica" - ("valor\ de\ fábrica" \times \frac{"años\ de\ uso"}{"vida\ útil\ estimada"})$$

Depreciación anual

$$"valor\ actualizado" = valor\ de\ fábrica \times (1 - "tasa\ de\ depreciación")^{años\ de\ uso}$$

donde años de usos es

$$"años\ de\ uso" = "año\ actual\ o\ año\ a\ calcular" - "año\ de\ fabricación\ o\ modelo"$$



The image shows two overlapping windows from a software application. The main window, titled 'Tasación de motos', contains input fields for 'Marca' (Honda), 'Modelo (Año)' (1956), 'Año a calcular' (2024), and 'Valor de fabricación \$' (5000). It also has fields for 'Tasa de depreciación' (0,1) and 'Vida Útil(años)' (50). At the bottom are 'Calcular Costo' and 'Cerrar' buttons. The 'Calcular Costo' button is highlighted with a blue border. Overlaid on this is a smaller window titled 'Ver Resultados' which displays the calculated values: 'Marca: Honda, Modelo: 1956, Valor Fabricación: \$ 5000,00', 'Depreciación lineal: \$ 4932,00', and 'Depreciación anual: \$ 3,87'. It also has a 'Cerrar' button.

Tasación de motos	
Marca	Honda
Modelo (Año)	1956
Año a calcular	2024
Valor de fabricación \$	5000
Tasa de depreciación:	0,1
Vida Útil(años):	50
<b>Calcular Costo</b> <b>Cerrar</b>	

Ver Resultados	
<b>Resultados</b>	
Marca: Honda, Modelo: 1956, Valor Fabricación: \$ 5000,00	
Depreciación lineal: \$ 4932,00	
Depreciación anual: \$ 3,87	
<b>Cerrar</b>	

Figura 1.2 Diseño de pantallas

## II. Ejercicio 2. Colectivo

Una empresa de transportes desea un programa para controlar el recorrido de sus unidades:

- Por ser servicio interurbano cada pasajero debe ir sentado. (controlar la disponibilidad de asientos).
- Al iniciar el recorrido se debe ingresar la hora en formato (HH:MM) de salida y la cantidad de asientos de la unidad.
- En cada parada pueden ascender o descender pasajeros, por lo que se debe informar en procesos separados.
  - a- cantidad de pasajeros que descienden.
  - b- Cantidad de pasajeros que ascienden.
  - c- Tiempo de demora.
- Al finalizar el recorrido se debe informar:
  - a- Cantidad de pasajeros transportados.
  - b- Tiempo total de recorrido.
  - c- Cantidad de paradas.
  - d- Tiempo total de demora en las paradas.

Plantear el UML primero en base a lo expresado en el enunciado y el diseño de la interfaz gráfica. Implemente una clase que modele el colectivo y una aplicación para verificar su funcionamiento.

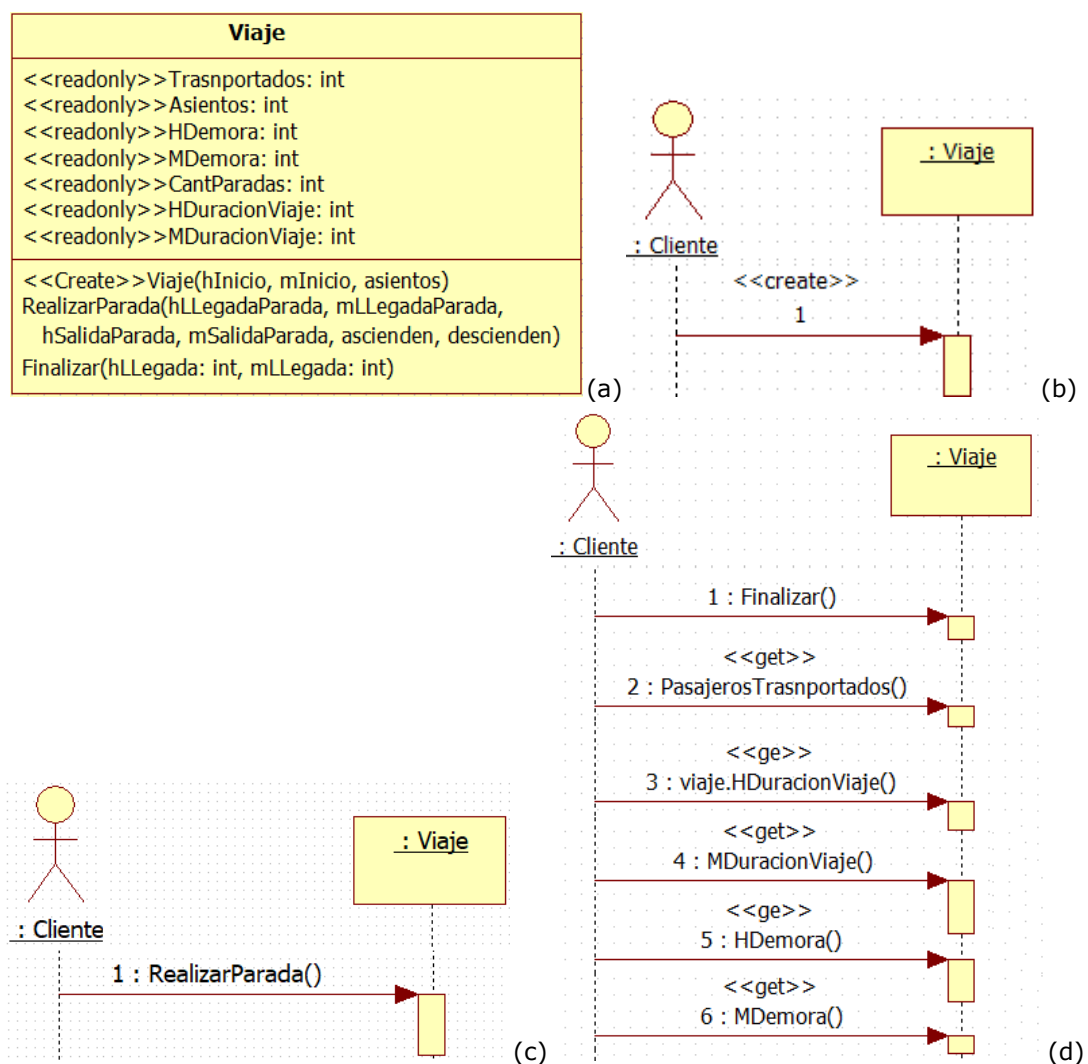


Figura 1.1 Especificación (a) Diagrama de clases (b) Diagrama de secuencia-inicio viaje. (c) Realizando una parada. (d) finalizando e informando el viaje

The 'Empresa de viajes' window contains three main sections. The top section has a button labeled 'Iniciar Viaje'. The middle section has a button labeled 'Ingresada Parada'. The bottom section is titled 'Finalizar Parada' and contains a label 'Hora de Finalización (HH:MM)' followed by two empty text boxes separated by a colon, and a button labeled 'Finalizar Viaje'.

(a)

The 'Ingreso datos parada' window is titled 'Parada'. It contains two rows of time input fields. The first row is labeled 'Llegada (HH:MM)' and the second 'Salida (HH:MM)', each followed by two empty text boxes separated by a colon. Below these are two more rows: 'Ascienden' and 'Descienden', each followed by an empty text box. At the bottom, there are two buttons, both labeled 'Ingresada Parada'.

(b)

The 'Informe' window is titled 'Informe del viaje'. It contains a large text area labeled 'ItbResultados'. At the bottom right, there is a button labeled 'Cerrar'.

(c)

Figura 2.2. Formularios propuestos para el ejercicio 2. (a) Formulario principal.  
(b) Formulario para dar inicio al viaje.  
(c) Formulario para mostrar el resumen final

### III. Ejercicio 3. Sistema de peaje

Un sistema de peaje requiere de un sistema que permita generar estadísticas sobre la cantidad de automóviles que pasan por el mismo. Para esto se debe cargar en el sistema el número del día del mes (a lo más 31) y la cantidad de autos que pasaron ese día. Luego el sistema debe generar un informe indicando:

- Los días en que ingresaron más autos que el promedio del mes.
- Cuál de los tercios del mes tuvo mayor movimiento

Nota: Considere el primer tercio del 1 al 10 inclusive, el segundo tercio del 11 al 20 inclusive y el último desde el 21 a fin de mes.

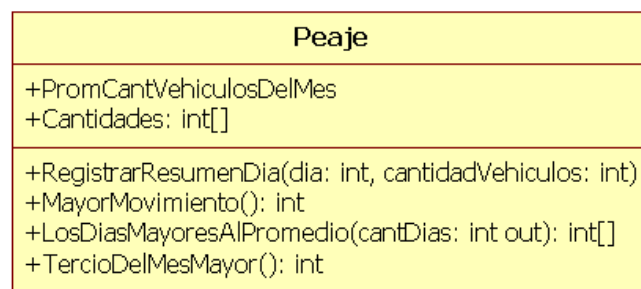


Figura 3.1 Diagrama UML del ejercicio 3

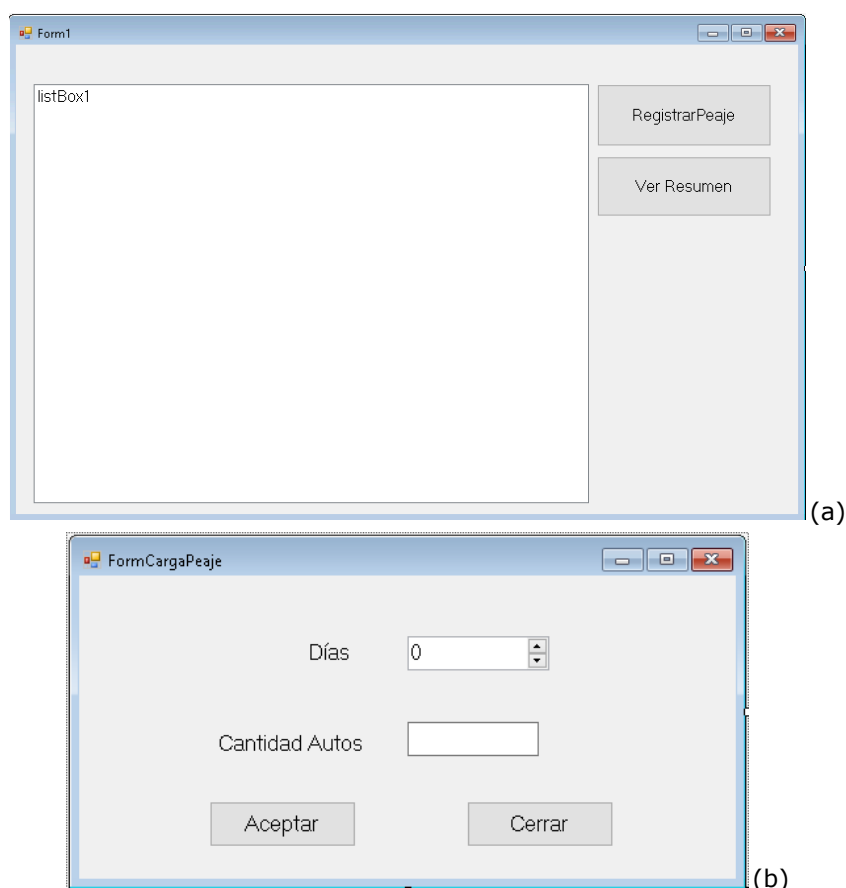


Figura 4.1 Diseño de las pantallas. (a) Pantalla principal. (b) Pantalla modal para la carga de los resúmenes diarios del peaje