

# Algoritmos e instrucciones

Construcción de la solución

# Algoritmos e instrucciones

---

## ▶ Algoritmo:

- ▶ Secuencia de instrucciones para resolver un problema
- ▶ Secuencia ordenada de pasos para realizar una actividad

## ▶ Ejemplos

- ▶ Algoritmo para preparar huevos pericos
- ▶ Algoritmo para amarrarse los zapatos
- ▶ Algoritmo para cambiar una llanta
- ▶ Algoritmo para llegar a una dirección dada



# Las líneas telefónicas

Caso de estudio #3

# Las líneas telefónicas (problema)

---

- ▶ Se quiere crear una aplicación para controlar los gastos telefónicos de una **empresa**. La empresa cuenta con **tres líneas telefónicas** a través de las cuales se pueden realizar llamadas locales, de larga distancia y a celulares
- ▶ La aplicación debe permitir:
  - ▶ Registrar una llamada en alguna de las líneas
  - ▶ Mostrar la **información** detallada de cada línea
    - ▶ **Número de llamadas** realizadas
    - ▶ **Duración** total de las llamadas en minutos
    - ▶ **Costo total** de las llamadas en pesos
  - ▶ Mostrar un consolidado total de la información de todas las líneas (costo total en pesos de las tres líneas, número total de llamadas realizadas, duración total de llamadas en minutos y el cálculo del costo promedio por minuto según el costo total y el total de minutos).
  - ▶ Reiniciar el uso las líneas telefónicas, dejando todos sus valores en cero



# Las líneas telefónicas (**solución**)

MiEmpresa - Manejo Líneas Telefónicas


Manejo Líneas Telefónicas

  
**MiEmpresa**  
Manejo Líneas Telefónicas

Totales

**\$ 0,00**  
Total Llamadas: 0  
Total Minutos: 0  
Costo promedio por minuto: N/A


Línea #1

  
**\$ 0,00**  
Número Llamadas: 0  
Número de Minutos: 0

Línea #2

  
**\$ 0,00**  
Número Llamadas: 0  
Número de Minutos: 0

Línea #3

  
**\$ 0,00**  
Número Llamadas: 0  
Número de Minutos: 0

Opciones

# Las líneas telefónicas: Requerimientos funcionales

---

- ▶ R1: Registrar (agregar) una llamada en alguna de las líneas
- ▶ R2: Mostrar la información detallada de cada línea
- ▶ R3: ...
- ▶ R4: ...

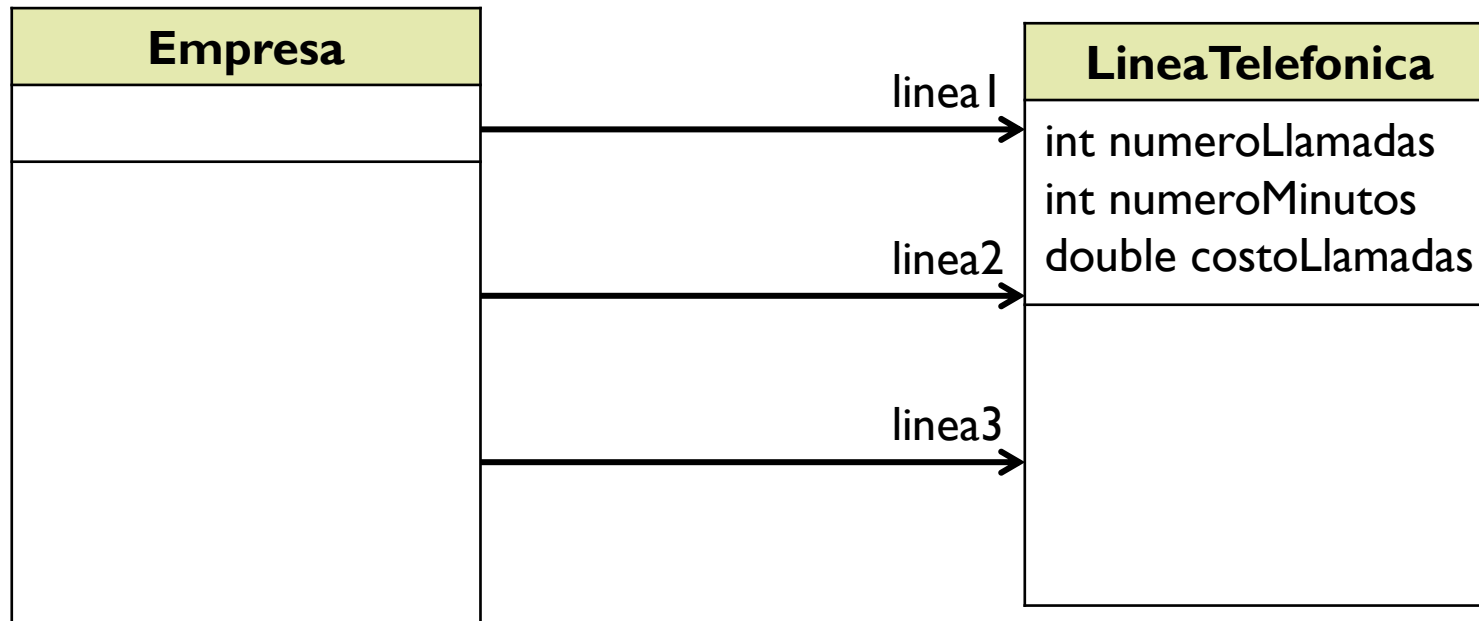


# R1: Registrar (agregar) una llamada en alguna de las líneas

<b>Nombre</b>	R1: Agregar una llamada a una línea telefónica
<b>Resumen</b>	Se agrega una llamada a una línea telefónica. Se debe especificar la cantidad de minutos consumidos, así como el tipo de llamada realizada.
<b>Entradas</b>	
Número de línea, siendo opciones válidas la línea 1, 2 o 3.	
Número de minutos consumidos, sabiendo que el número de minutos es un valor positivo.	
Tipo de llamada realizada. Puede ser local, larga distancia o celular.	
<b>Resultados</b>	
La línea telefónica tiene una llamada más.	
Los minutos consumidos por la línea especificada aumentaron según el número de minutos de la llamada.	
El costo total de llamadas realizadas por la línea especificada se incrementó en el costo de la llamada. El valor por minuto de una llamada local es de \$35, de una llamada de larga distancia es de \$380, y de una llamada a celular es de \$999	
Los totales de toda la empresa se actualizan.	

# Las líneas telefónicas: **Modelo del mundo**

---





# Métodos

---

- ▶ Son los «algoritmos» de la clase
- ▶ Lo que la clase sabe hacer:
  - ▶ Resolver un problema puntual
  - ▶ Servicio que la clase presta a otras clases del modelo
- ▶ Piense que...
  - ▶ Una clase es la responsable de manejar la información contenida en sus atributos
  - ▶ Los métodos son el medio para hacerlo



# Ejemplo: ¿Qué debe saber hacer una línea telefónica?

---

## ▶ Informar

- ▶ El número total de sus llamadas
- ▶ El costo total de sus llamadas
- ▶ La cantidad de minutos consumidos

## ▶ Agregar

- ▶ Una llamada local
- ▶ Una llamada de larga distancia
- ▶ Una llamada celular



# Métodos

---

- ▶ Cada una de las acciones que sabe hacer una clase es un método



# Ejemplo: ¿Qué debe saber hacer una línea telefónica?

---

## ► Informar

- El número total de sus llamadas `darCostoLlamadas`
- El costo total de sus llamadas `darNumeroLlamadas`
- La cantidad de minutos consumidos `darNumeroMinutos`

## ► Agregar

- Una llamada local `agregarLlamadaLocal`
- Una llamada de larga distancia `agregarLlamadaLargaDistancia`
- Una llamada celular `agregarLlamadaCelular`



# Métodos en Java

---

- ▶ Un método tiene dos partes importantes
    - ▶ Signatura
      - ▶ Visibilidad (scope) – siempre public
      - ▶ Tipo de respuesta – Tipo de dato al que pertenece el resultado que va a calcular el método. Si no hay respuesta se indica vacío (void)
      - ▶ Nombre
      - ▶ Parámetros – Conjunto de valores necesarios para resolver el problema
    - ▶ Cuerpo
      - ...
- ```
public void agregarLlamadaLocal(int minutos)
```



# Métodos en Java

---

- ▶ Un método tiene dos partes importantes
  - ▶ Signatura
  - ▶ **Cuerpo**
    - ▶ Lista de instrucciones que representa el algoritmo que resuelve el problema puntual
    - ▶ En el cuerpo se explica la forma de utilizar los valores de los atributos para calcular alguna información o la forma de modificarlos

```
public void agregarLlamadaLocal(int minutos){  
    numeroLlamadas = numeroLlamadas+1;  
    numeroMinutos = numeroMinutos + minutos;  
    costoLlamadas = costoLlamadas + (minutos * 35)  
}
```



# Instrucciones en Java

Tipos de instrucciones

# Instrucción de **asignación**

---

- ▶ Cambiar/asignar el valor asignado a un atributo
- ▶ Se construye con un igual «**=**»

```
public void agregarLlamadaLocal(int minutos)
{
    numeroLlamadas = numeroLlamadas + 1;
    numeroMinutos = numeroMinutos + minutos;
    costoLlamadas = costoLlamadas + ( minutos * 35 );
}
```





# Instrucción de **asignación**

---

atributo = expresión;

**numeroLlamadas** = **numeroLlamadas** + 1;

Atributo que va  
a ser modificado

Expresión que indica el  
nuevo valor que debe  
guardarse en el atributo

Pueden hacer parte de una expresión:  
atributos, parámetros y valores constantes

Con operadores aritméticos  
+, -, \*, /



# Instrucción de **retorno**

---

- ▶ Para devolver un resultado como solución del problema puntual
- ▶ Utiliza la palabra reservada «**return**»

```
public int darNumeroLlamadas( )  
{  
    return numeroLlamadas;  
}
```



# Instrucción de llamado a otro método

---

- ▶ Se hace para construir métodos complejos a partir de métodos mas simples que ya están escritos.
  - ▶ Usar métodos de la misma clase
  - ▶ Usar métodos de un objeto de otra clase con la cual existe una asociación
- ▶ Ejemplo: calcular el monto de los impuestos que debe pagar el empleado en un año. Los impuestos se calculan como el 19,5% del total de salarios recibidos en un año
  - ▶ Calcular el valor total del salario anual
  - ▶ Calcular el monto del impuesto usando el método anterior



# Invocación de un método de la **misma** clase

---

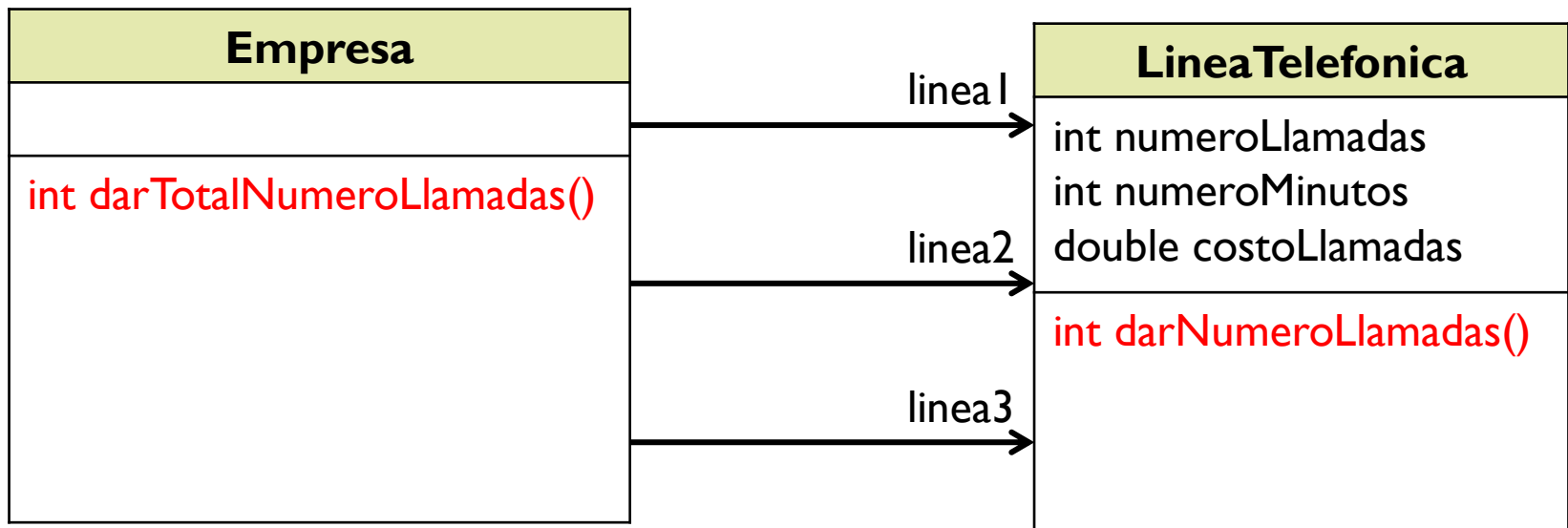
| Empleado                                                    |
|-------------------------------------------------------------|
| String nombre<br>String apellido<br>int sexo<br>int salario |
| int calcularSalarioAnual()<br>int calcularImpuesto()        |

```
public class Empleado{  
    ...  
    public int calcularSalarioAnual(){  
        return (salario * 12);  
    }  
    public int calcularImpuesto(){  
        return (calcularSalarioAnual()*19.5)/100  
    }  
    ...  
}
```



# Invocación de un método de **una asociación**

- ▶ Se hace cuando se necesita obtener o modificar alguna información de un objeto de otra clase con el cual existe una asociación



## Invocación de un método de una asociación

```
public class LineaTelefonica{
```

• • •

```
public int darNumeroLlamadas(){
```

```
return _____;
```

}

• • •

}

| LineaTelefonica                                                 |
|-----------------------------------------------------------------|
| int numeroLlamadas<br>int numeroMinutos<br>double costoLlamadas |
| int darNumeroLlamadas()                                         |



# Invocación de un método de **una asociación**

---

```
public class LineaTelefonica{
```

```
...
```

```
public int darTotalNumeroLlamadas(){  
    return linea1.darNumeroLlamadas()+  
        linea2.darNumeroLlamadas()+  
        linea3.darNumeroLlamadas();
```

```
}
```

```
...
```

```
}
```

| LineaTelefonica                                                 |
|-----------------------------------------------------------------|
| int numeroLlamadas<br>int numeroMinutos<br>double costoLlamadas |
| int darNumeroLlamadas()                                         |



# Llamado de métodos con parámetros

---

- ▶ **¿Cuándo necesita parámetros un método?**
  - ▶ Cuando la información que tiene el objeto en sus atributos no es suficiente para resolver el problema
- ▶ **¿Cómo se declara un parámetro?**
  - ▶ En la signature del método se define el tipo del dato del parámetro y se le asocia un nombre
- ▶ **¿Cómo se utiliza el valor de un parámetro?**
  - ▶ Basta con utilizar el nombre del parámetro en el cuerpo del método de la misma manera que se utilizan los atributos





# Ejemplo llamado de métodos con parámetros

---

```
public class LineaTelefonica{
```

```
...
```

```
    public void agregarLlamadaLocal( int minutos ){  
        numeroLlamadas = numeroLlamadas + 1;  
        numeroMinutos = numeroMinutos + minutos;  
        costoLlamadas = costoLlamadas + ( minutos * 35 );  
    }
```

```
...
```

```
}
```



# Ejemplo llamado de métodos con parámetros

---

```
public class Empresa{  
    ...  
    public void agregarLlamadaLocalLinea l ( int minutos )  
    {  
        linea l .agregarLlamadaLocal( minutos );  
    }  
    ...  
}
```



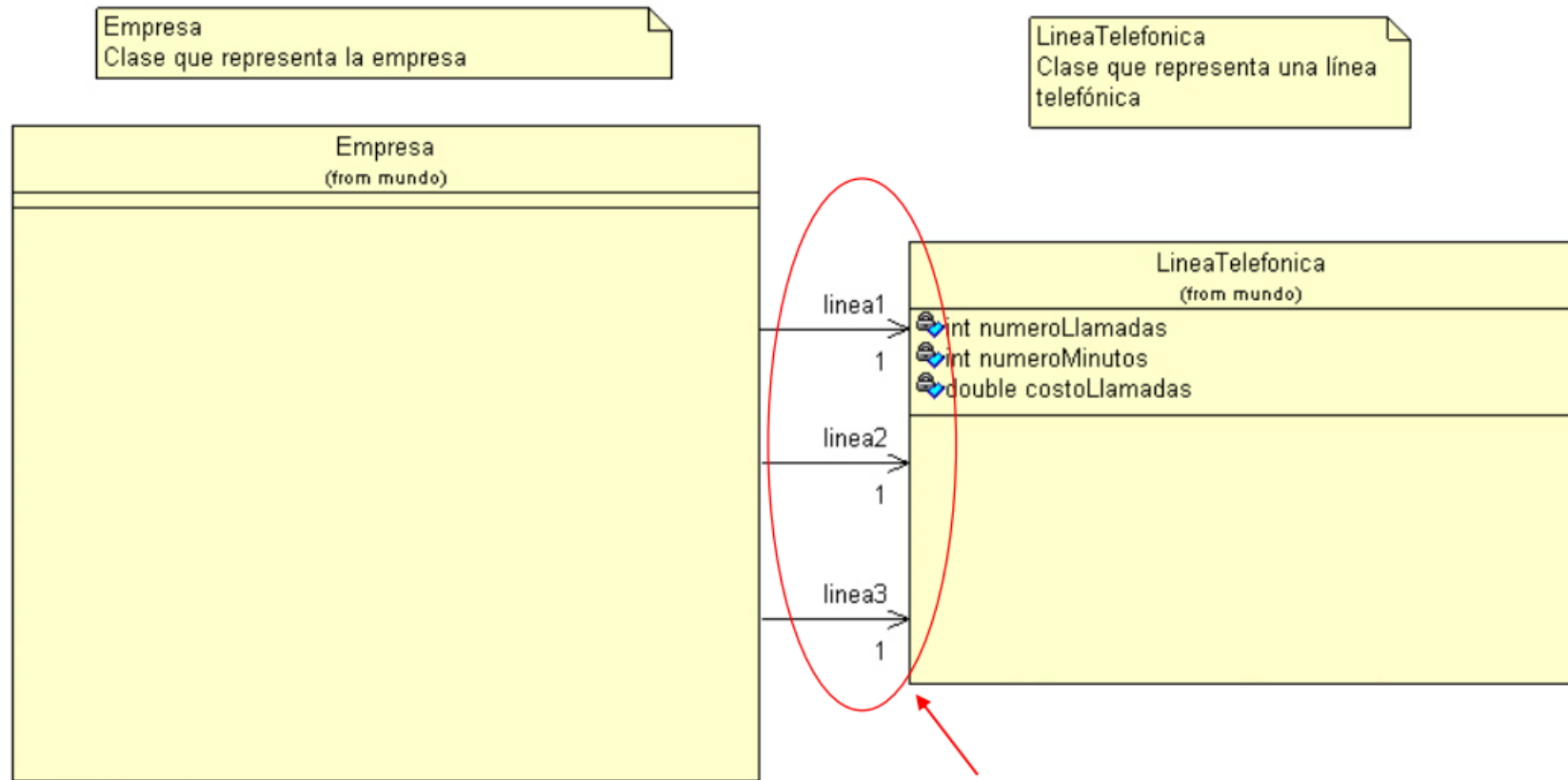
# Instrucción de creación de objetos

---

- ▶ Utiliza la palabra reservada **new**
- ▶ Los objetos de las asociaciones los crea la **clase dueña** de las mismas en alguno de sus métodos
  - ▶ inicializar



# Creación de los objetos de las clases con quienes hay asociaciones



Hay 3 líneas telefónicas  
(asociaciones: SON OBJETOS)



# Ejemplo: Creación de las 3 líneas telefónicas en la clase Empresa

```
/**  
 * Inicializa las líneas telefónicas de la empresa <br>  
 * <b>post: </b> Se inicializaron las 3 líneas telefónicas.  
 */  
public void inicializar( )  
{  
    //  
    //Inicializa la línea 1  
    linea1 = new LineaTelefonica( );  
    linea1.inicializar( );  
    //  
    //Inicializa la línea 2  
    linea2 = new LineaTelefonica( );  
    linea2.inicializar( );  
    //  
    //Inicializa la línea 3  
    linea3 = new LineaTelefonica( );  
    linea3.inicializar( );  
}
```



# Ver método constructor del simulador

---

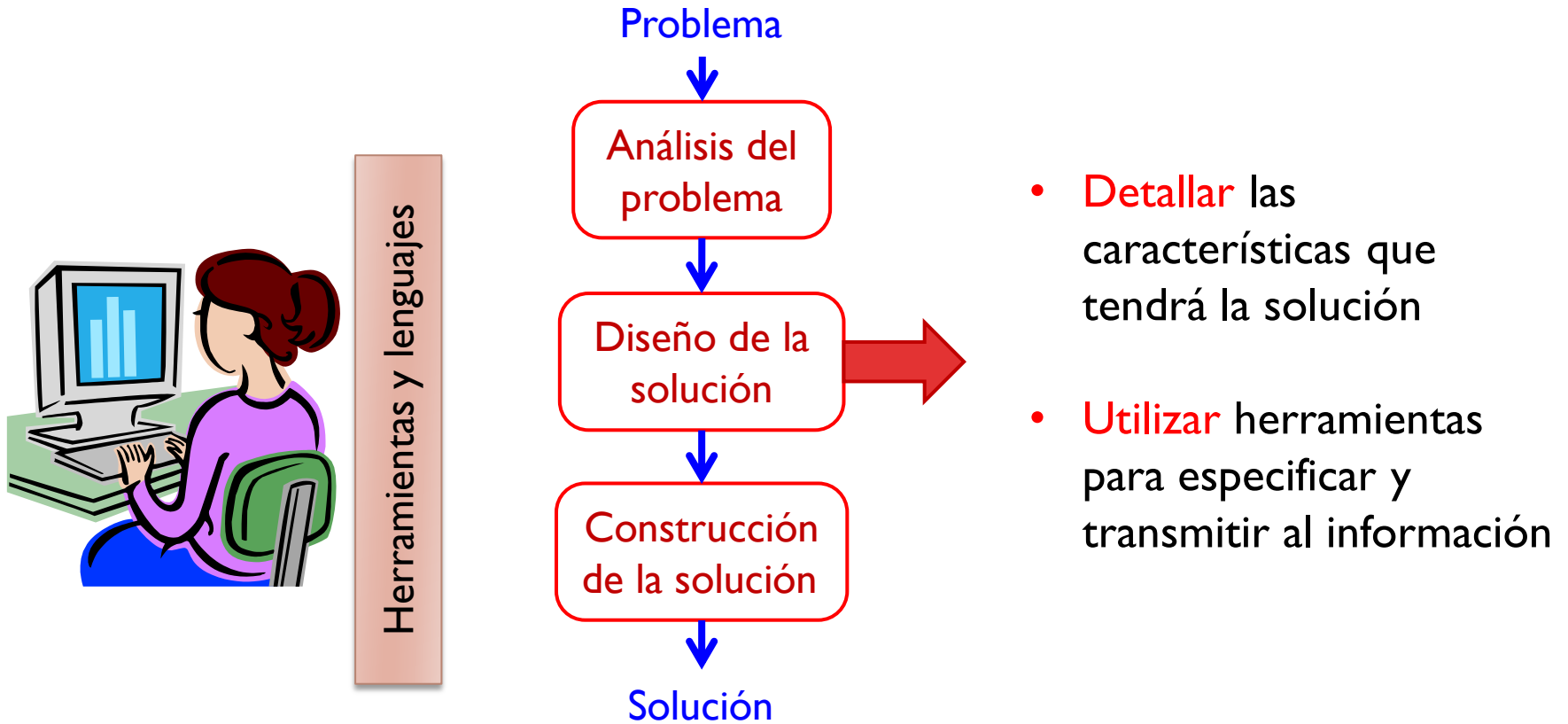
```
public SimuladorBancario( String pCedula, String pNombre )
{
    // Inicializa los atributos personales del cliente
    nombre = pNombre;
    cedula = pCedula;
    // Inicializa el mes en el 1
    mesActual = 1;
    // Inicializa las tres cuentas en vacío
    corriente = new CuentaCorriente( );
    ahorros = new CuentaAhorros( );
    inversion = new CDT( );
}
```



## Etapa 2: Diseño de la solución

# Solucionar un problema (Construir un programa)

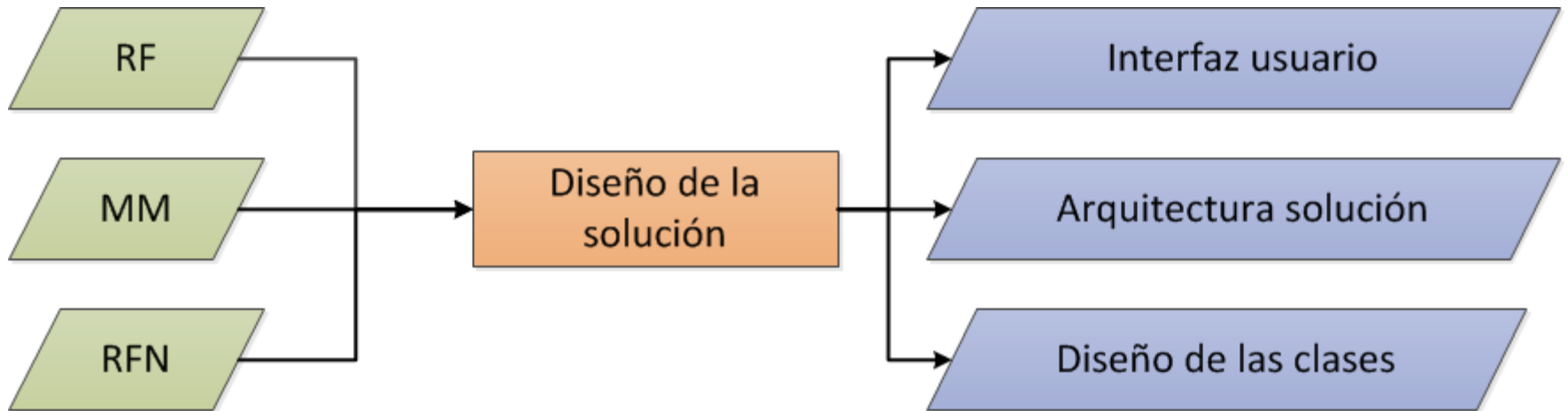
---



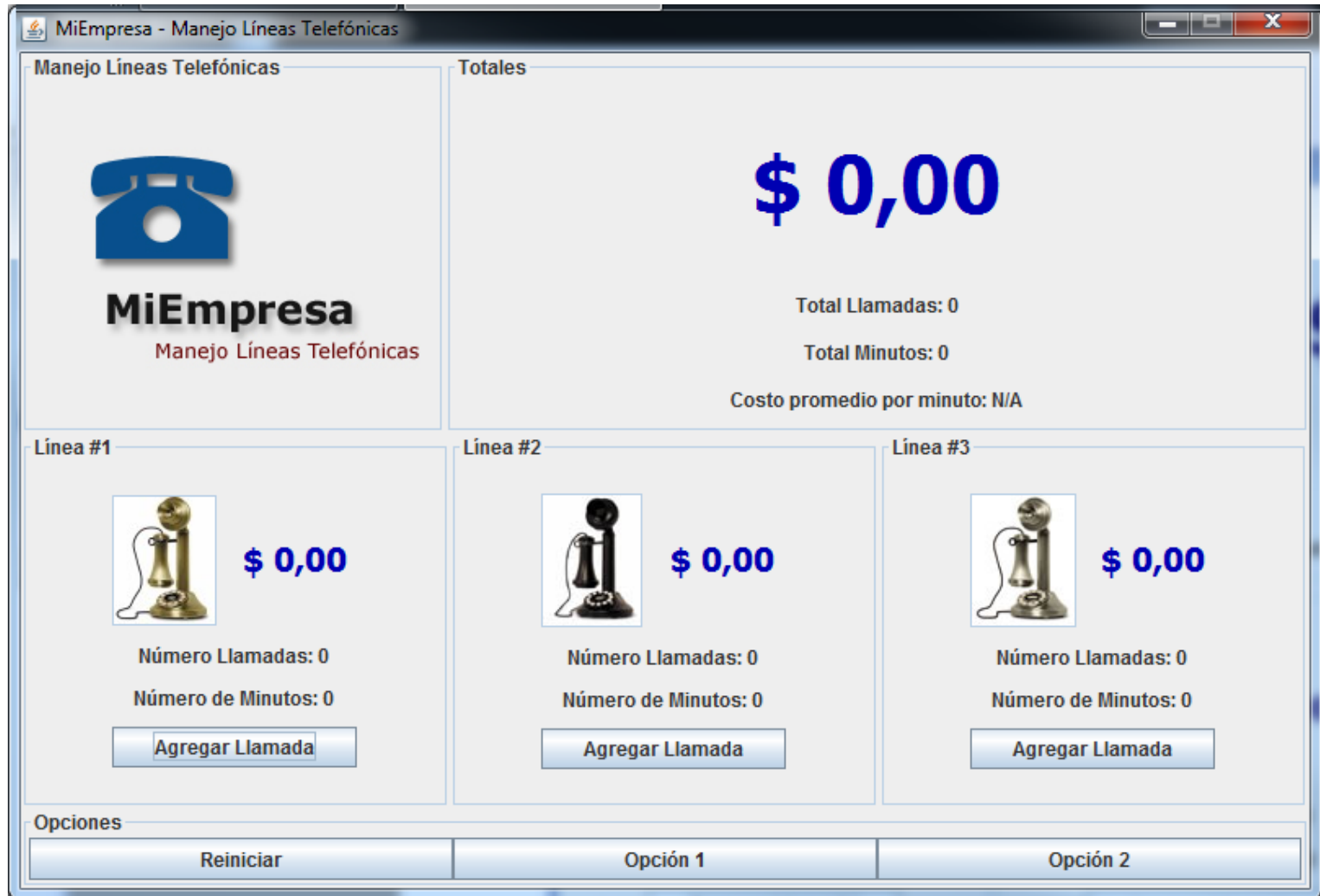


# Diseño de la solución

---

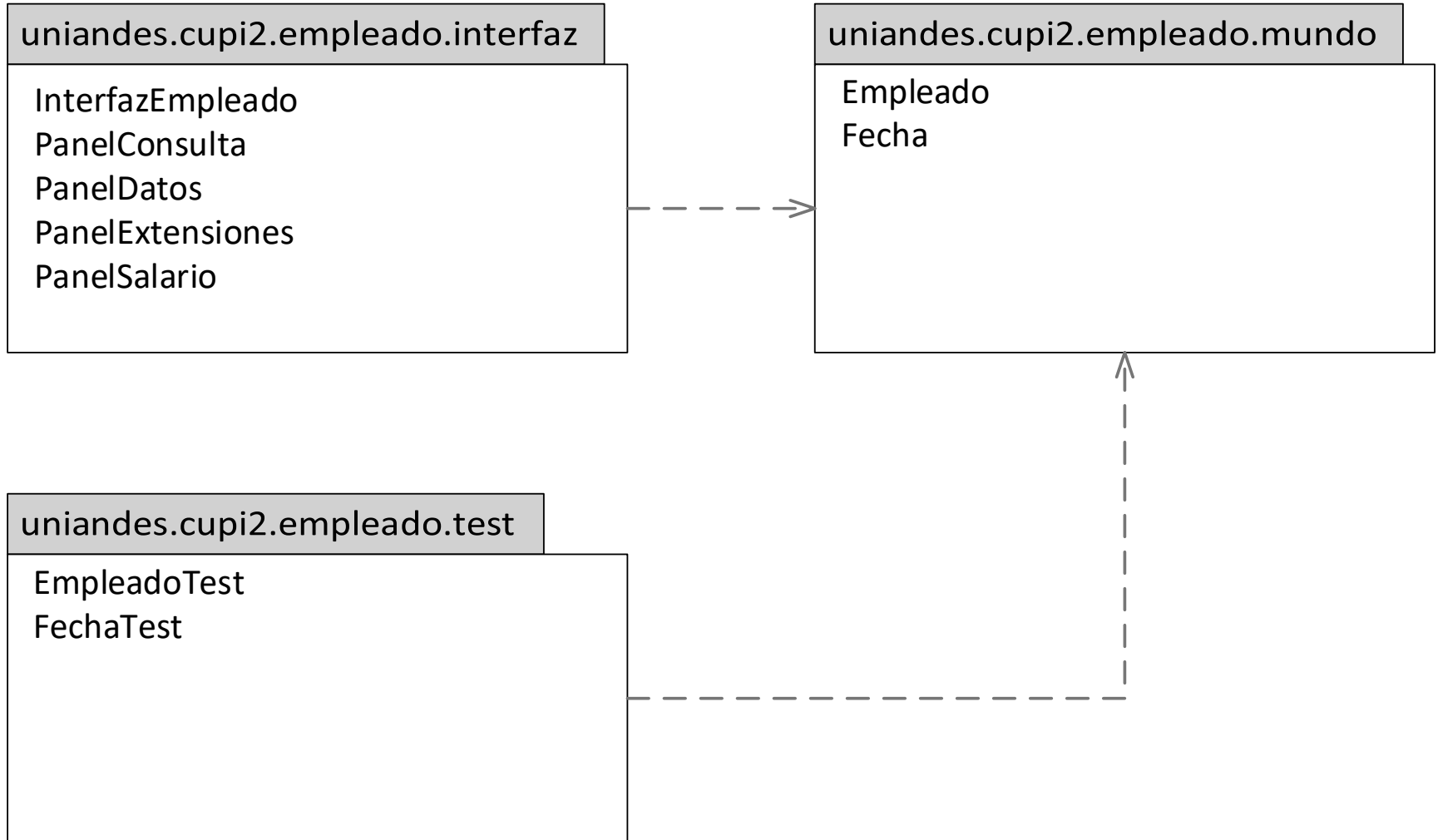


# Diseño de la solución: **Interfaz de usuario**



# Diseño de la solución: **Arquitectura de solución**

---



# Diseño de la solución: **Diseño de las clases**

---

- ▶ **Mostrar los detalles de cada una de las clases que van a hacer parte del programa**
  - ▶ Clases
  - ▶ Atributos
  - ▶ Asociaciones
  - ▶ Signaturas de los métodos



# Hoja de Trabajo

1. Agregue a la clase **LineaTelefonica** un nuevo atributo llamado **numeroMinutosCelular** para guardar la cantidad de minutos consumidos en llamadas a celular.
2. Escriba el método **inicializar** de la clase **LineaTelefonica** que asigna el valor cero a todos sus atributos.
3. Escriba el método **inicializar** de la clase **Empresa** que inicializa las 3 líneas telefónicas.
4. Complete el método **darNumeroMinutosCelular** de la clase **LineaTelefonica** que devuelva (retorne) el número de minutos consumidos en llamadas a celular sobre la línea.
5. Construya el método **darTotalMinutosCelular** de la clase **Empresa** que retorne el número total de minutos consumidos en llamadas a celular realizadas en las 3 líneas telefónicas.
6. La empresa necesita conocer qué porcentaje representa la cantidad total de minutos de llamadas a celular con respecto al total de minutos consumidos en todos los tipos de llamadas en todas las líneas. Construya el método **darPorcentajeCelular** de la clase **Empresa** que calcule y retorne este valor.

