

Universidad Central “Marta Abreu”

De Las Villas



**Proyecto Final de Programación en C/C++ de Segundo Año de la carrera Lic. en Ciencia de la Computación**

**Título: Metrocontador**

**Autor: Yailedainis Rodríguez Morfa**

**Curso: 2024**

Problemática**:** “Metrocontador”

La empresa eléctrica de la provincia de Villa Clara desea llevar un control estricto del consumo de electricidad por sus clientes. Con el objetivo de tener un mejor control sobre el consumo de la energía eléctrica en las viviendas, se desea implementar un sistema que automatice diferentes análisis.

Inicialmente debe tener en cuenta que la empresa desea tener el control para un año de consumo. Usted solo tiene la tarea de automatizarlo para un mes (1 mes=30 días). La cantidad de clientes y el consumo en kWh de cada día es dada por el jefe de operaciones.

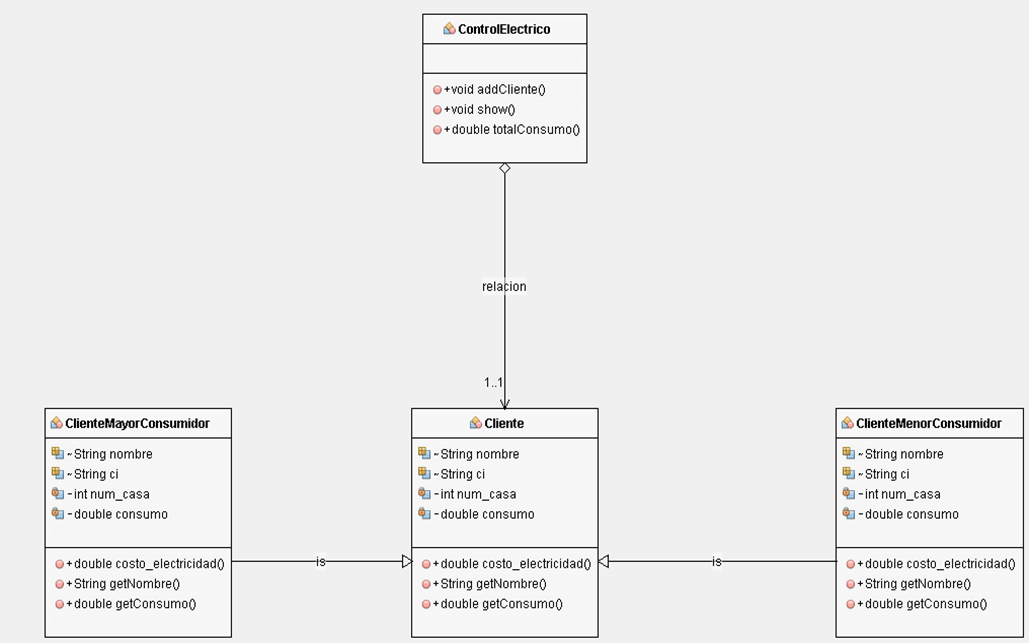
El programa debe contener una lista con los siguientes datos de clientes en el municipio de Villa Clara. Los datos que se conocen de cada cliente es el carné de identidad (CI), el nombre, número de la casa, y el consumo de energía eléctrica en el mes de marzo (el consumo de energía se da en kWh). La empresa los separa en dos grupos uno dado por los clientes que sobrepasan el consumo máximo del mes de 500kWh(ClienteMayorConsumidor) y el otro por los clientes que no llegan ni sobrepasan ese consumo (ClienteMenorConsumidor).

1- De los clientes se quiere saber el costo de la electricidad en el mes de marzo (consumo de energía en kWh \*0.09) al ClienteMayorConsumidor se le aplica un aumento de un 25%.

2-Determinar el total de kWh consumidos por esos clientes por mes.

3-Determinar el nombre del cliente con mayor consumo de energía en el mes.

**Diagrama de Clases**

****

**Solución a la problemática:**

La clase Cliente se utiliza para representar información sobre un cliente en un sistema automatizado para el control del consumo de electricidad en la provincia de Villa Clara. Aquí está una breve descripción de los miembros y métodos de la clase:

1. **Miembros protegidos**:
   * nombre: Una cadena que almacena el nombre del cliente.
   * ci: Una cadena que representa el número de cédula de identidad del cliente.
   * num\_casa: Un entero que indica el número de la casa del cliente.
   * consumo: Un número de punto flotante que registra el consumo eléctrico del cliente.
2. **Constructor**:
   * Cliente(const string& nombre, string ci, int num\_casa, double consumo): Este constructor inicializa los miembros de la clase con los valores proporcionados. Es posible que deba implementarse en una subclase específica.
3. **Método virtual**:
   * virtual double costo\_electricidad(): Este método calcula el costo de electricidad para el cliente. Dado que es virtual, se espera que las subclases proporcionen su propia implementación.
4. **Métodos públicos**:
   * string getNombre(): Devuelve el nombre del cliente.
   * double getConsumo(): Devuelve el consumo eléctrico del cliente.

Las subclases ClienteMayorConsumidor y ClienteMenorConsumidor heredan de esta clase.

**ClienteMayorConsumidor:**

1. **Herencia**:
   * ClienteMayorConsumidor hereda públicamente de la clase base Cliente. Esto significa que todos los miembros protegidos y públicos de Cliente estarán disponibles en ClienteMayorConsumidor.
2. **Constructor**:
   * ClienteMayorConsumidor(const std::string& nombre, string ci, int num\_casa, double consumo): Este constructor inicializa los miembros de la clase ClienteMayorConsumidor con los valores proporcionados. Es posible que deba implementarse en la subclase.
3. **Método sobrescrito**:
   * double costo\_electricidad()override: Este método sobrescribe el método virtual costo\_electricidad() de la clase base Cliente. Aquí, puedes proporcionar una implementación específica para calcular el costo de electricidad para un cliente mayor consumidor.

**ClienteMenorConsumidor:**

1. **Herencia**:
   * ClienteMenorConsumidor hereda públicamente de la clase base Cliente. Esto significa que todos los miembros protegidos y públicos de Cliente estarán disponibles en ClienteMenorConsumidor.
2. **Constructor**:
   * ClienteMenorConsumidor(const string& nombre, string ci, int num\_casa, double consumo): Este constructor inicializa los miembros de la clase ClienteMenorConsumidor con los valores proporcionados. Es posible que deba implementarse en la subclase.
3. **Método sobrescrito**:
   * double costo\_electricidad()override: Este método sobrescribe el método virtual costo\_electricidad() de la clase base Cliente. Aquí, puedes proporcionar una implementación específica para calcular el costo de electricidad para un cliente menor consumidor.

La clase **ControlElectrico** se utiliza para administrar y rastrear información sobre los clientes en un sistema automatizado para el control del consumo de electricidad en la provincia de Villa Clara. Los métodos proporcionados permiten agregar clientes, mostrar detalles y calcular el consumo total.

1. **Miembro privado**:
   * clientes: Un vector que almacena punteros a objetos de tipo Cliente. Esto permite que ControlElectrico mantenga una lista de clientes.
2. **Constructor**:
   * ControlElectrico(): Este constructor inicializa un objeto de la clase ControlElectrico. Puede ser utilizado para crear una instancia de ControlElectrico.
3. **Método público**:
   * void addCliente(Cliente\* cliente): Este método permite agregar un cliente a la lista de clientes. Recibe un puntero al objeto Cliente como argumento.
   * void show(): Este método podría mostrar información relevante sobre los clientes almacenados en la lista. La implementación específica dependerá de las necesidades del sistema.
   * double totalConsumo(): Este método calcula el consumo total de electricidad de todos los clientes en la lista y devuelve un valor de punto flotante.

Este sistema automatizado proporcionará una gestión más eficiente y un control más preciso del consumo eléctrico en las viviendas, beneficiando tanto a la empresa como a los clientes. Esta automatización permite una respuesta más ágil al centrarse en mejorar el control y la gestión del consumo de la electricidad a través de la automatización, la empresa eléctrica no solo fortalece su posición en el mercado, sino que también contribuye activamente a la sostenibilidad ambiental y al uso responsable de los recursos energéticos. Este enfoque estratégico no solo beneficia a la empresa en términos de eficiencia y rentabilidad, sino que también tiene un impacto positivo en la comunidad al fomentar prácticas energéticas más conscientes y sostenibles.