

Avance 1 : Proyecto

Angie Lorena Castillo Guerrero - 11232423922

Bryan Daniel Buitrago Avila - 11162428434

Lizbeth Johanna Sarmiento Hernández - 11232478438

Martes y jueves

Horario : 9:00 - 11:00

Lógica Computacional

Universidad Antonio Narino sede
SUR

2025 - 1

C. Explique las etapas del proceso de solución de problemas

-Análisis

El problema se analiza teniendo presente la especificación de los requisitos dados por la persona que encarga el problema.

-Diseño

Después de analizar el problema, se diseña una solución que conduzca a un algoritmo que resuelva el problema.

-Codificación (implementación)

La solución se escribe en la sintaxis del lenguaje y se obtiene un programa fuente que se compila a continuación.

-Ejecución, verificación y depuración

El programa se ejecuta, se comprueba de manera rigurosa y se eliminan todos los errores que puedan aparecer.

-Mantenimiento:

El programa se actualiza y modifica, cada vez que sea necesario de modo que se cumplan todas las necesidades de los usuarios.

-Documentación

Escritura de las diferentes fases del ciclo de vida del software, esencialmente el análisis, diseño y codificación unidos a manuales de usuarios y de referencia. Así como el mantenimiento.

F. Tarea N°2 (pág. 13) - Identificar los requerimientos funcionales de un programa.

Requerimiento funcional 1

Nombre: Sistema de trasterencia

Resumen: Este sistema debe permitir la transferencia de fondos entre cuentas

Entradas: Cuenta origen - Cuenta destino - Valor transferencia

Salidas: Comprobante de transferencia - "transferencia exitosa"

Requerimiento funcional 2

Nombre → Sistema de creación de cuenta Crd.

Resumen → Debe recopilar datos de la persona y crear una cuenta

Entradas → Cedula - Ingresos - trabajo

Salidas → Número de cuenta crédito

Requerimiento funcional 3

Nombre → Bloquear cuenta crédito

Resumen → Se busca permitir bloquear la cuenta en caso de ser necesario

Entradas → Número de cuenta - Usuario

Salidas → Notificación de bloqueo

i Tarea N°5 (Pág. 20)

Cuenta Bancaria

Atributo	Valores Posibles	Diagrama UML
Usuario	Cadena de caracteres	Cuenta Bancaria Usuario
Número de cuenta	Valores enteros positivos	Número de cuenta
Contrasena	Valores enteros positivas	Contrasena

Clase: Cuenta corriente

Atributo	Valores Posibles	Diagrama UML
Saldo	Cadena de caracteres	Cuenta Corriente Saldo
Movimientos	Cadena de caracteres	Movimientos
Transferencias	Valores enteros Positivos	Transferencia

Clase: Cuenta Ahorros

Atributo	Valores Posibles	Diagrama UML
Ahorros	Valores enteras Positivos	Cuenta Ahorros Ahorros
Movimientos	Cadena de caracteres	Movimientos.
Transferencias	Valores enteros Positivos	Transferencias

Requerimiento
Funcional 4

Nombre → Sistema de retiro de dinero

Resumen → Debe generar la procedimiento
y comprobante de retiro.

Entradas → Digitar la cedula -
numero de cuenta -
Monto por retirar

Salidas → Comprobante de retiro

Clase: CDT

Atributo	Variables posibles	Diagrama UML
Credito	Valores enteros Positivos	<pre> classDiagram class CDT { Credito } </pre>
Saldo	Valores enteros Positivos	<pre> classDiagram class CDT { Credito Saldo } </pre>
Ahorro	Valores enteros Positivos	<pre> classDiagram class CDT { Credito Saldo Ahorro } </pre>

Clase: mes

Atributo	Variables posibles	Diagrama UML
Día Semanal	Cadena de caracteres	<pre> classDiagram class mes { DiaSemanal } </pre>
Día	Valores enteros 1 al 31	<pre> classDiagram class mes { DiaSemanal Dia } </pre>
Festivos	Cantidad entre 1 a 2	<pre> classDiagram class mes { DiaSemanal Dia Festivos } </pre>

Angie Castillo - Bryan Buitrago - Lizbeth Samiento.

Lógica computacional - M.J 9 - 11.

al Ciclo de vida de la construcción de un programa

Paso 1 = El cliente tiene un problema y necesita el uso de un computador para poder recordarlo, por lo cual contacta a un programador.

Paso 2 = El programador sigue un proceso para lograr entender el problema del cliente y así lograr construir una solución que formará parte del programa.

Paso 3 = El programador instala el programa que resuelve el problema y deja que el usuario lo utilice para resolver el problema. El cliente y el usuario no tienen que ser la misma persona necesariamente.

d) ¿Cuáles son los elementos que se le deben entregar a un cliente?

1. El diseño
2. El programa
3. Las pruebas de corrección del programa.
4. Manual de usuario

g)

Requerimiento funcional 1

Nombre = Cálculo de área

Resumen = El programa debe permitir al usuario ingresar la base y la altura del triángulo para poder así calcular el área correspondiente.

Entrada = Base, Altura

Salida = Área del triángulo y su operación

Requerimiento funcional 2

Nombre: Cálculo del perímetro

Resumen: El programa debe permitir al usuario ingresar el valor de las longitudes de los lados del triángulo, para así calcular el perímetro.

Entrada = Longitudes de los lados; Lado ①, Lado ②, Lado ③

- Lado ①
- Lado ②
- Lado ③

Salida = Resultado del valor del perímetro y su operación.

Requerimiento funcional 3

Nombre = Cálculo de la altura

Resumen: El programa debe permitirle al usuario calcular la altura de un triángulo dependiendo del tipo de triángulo.

Entrada = Tipo de triángulo (equilátero, isósceles, escaleno)
Lado ①, Lado ②, Lado ③

Salida = Resultado de la altura y su operación

j) Tarea 6.

Conclusiones:

En general, la secuencia tiene un orden cronológico bien establecido, lo que facilita la comprensión del texto. Sin embargo en el paso 4, se deberá agregar más información sobre el paso a seguir en caso de que en la estación en donde me encuentre no pase una línea hacia mi destino.

10

B. Explique los aspectos que hacen parte del análisis de un problema

El primer paso para analizar un problema es entender el problema que tiene el cliente y expresarlo de forma que cualquier persona del equipo pueda entender la solución que el cliente espera. Para especificar el problema se deben identificar al menos tres aspectos: los requerimientos del usuario, el mundo en el que debe resolverse y las restricciones del cliente. Si esto no se cumple puede perder la confianza del cliente. Entonces, los tres aspectos en los que se divide un problema son:

1. Requerimiento funcional: Identificar lo que espera de la solución, en programación es el servicio que el programa debe proveer al usuario.
2. Contexto (mundo): Debe entender la funcionalidad, la estructura y funcionamiento del mundo en que se va a resolver.
3. Requerimientos no funcionales: Corresponde a las restricciones o condiciones que impone un cliente.

Tarea 1

Objetivo: Identificar los aspectos que forman parte de un problema

Problema: Un banco quiere crear un programa para manejar sus cajeros automáticos. Dicho programa solo debe permitir retirar dinero y consultar el saldo de una cuenta.

Identifique y discuta los aspectos que constituyen el problema. Si el enunciado no es explícito con respecto a algún punto, intente imaginar la manera de completarlo.

Cliente: El banco

Usuario: Clientes del banco

Requerimiento: Retirar dinero.

funcional : Consultar saldo de una cuenta.

Verificar la identidad.

Verificar que en la cuenta haya suficiente dinero para el retiro, si no, mostrar un mensaje de error y devolverlo a que digite otro monto.

Mundo del problema: El programa debe funcionar las 24h del día.

El cajero debe estar conectado con la base de datos del banco para monitorizar y gestionarlo.

Debe priorizar la seguridad de las cuentas con verificación de identidad y protección antifraude.

Requerimientos: La interfaz del programa debe de ser simple y no funcionales fácil de usar para cualquier persona.

El programa debe actualizar el saldo después de un retiro.

Debe funcionar rápido para las solicitudes del usuario.

Punto de Reflexión: ¿Cómo decidir si se trata de una entidad y no solo de una característica de una entidad ya identificada?

Podemos definir a las entidades por los sustantivos que se usan para explicar el problema. Para no confundir entidades con sus características, analizaremos si algunas de las entidades podrían derivar de algo más detallado de otra entidad más general.

K. Estudia los siguientes aspectos del ejemplo seleccionado: Enunciado, Requerimientos funcionales (casos de uso) y el Modelo (clases del proyecto). A continuación, redacta el enunciado del problema y el nombre cada uno de los requerimientos funcionales del proyecto.

- Enunciado del problema: El Empleado

Se desea desarrollar una aplicación que gestione de manera eficiente la información de los empleados de una empresa. La aplicación debe permitir, registrar, visualizar y modificar datos relacionados con los empleados, así como realizar cálculos relevantes para determinar aspectos como su edad, antigüedad en la empresa y prestaciones económicas a las que tienen derecho según su antigüedad y salario.

- Requerimientos funcionales:

1. Visualizar la información del empleado
2. Modificar el salario del empleado
3. Calcular la edad del empleado
4. Calcular la antigüedad del empleado en la empresa
5. Calcular las prestaciones del empleado
6. Cambiar el empleado

Tarea 4

Objetivo: Identificar las entidades del mundo para el caso de estudio 3: Un programa que maneje un triángulo

Leyendo el enunciado del caso y tratando de quitarle por los sustantivos para identificar las entidades del mundo del problema.

- Entidad:

Nombre: Triángulo

Descripción: Es la entidad más importante del mundo del problema, puesto que define su frontera

- Entidad:

Nombre: Puntos

Descripción: La clase puntos se emplea para representar el concepto de los puntos para cada vértice del triángulo en coordenadas x, y.

- Entidad:

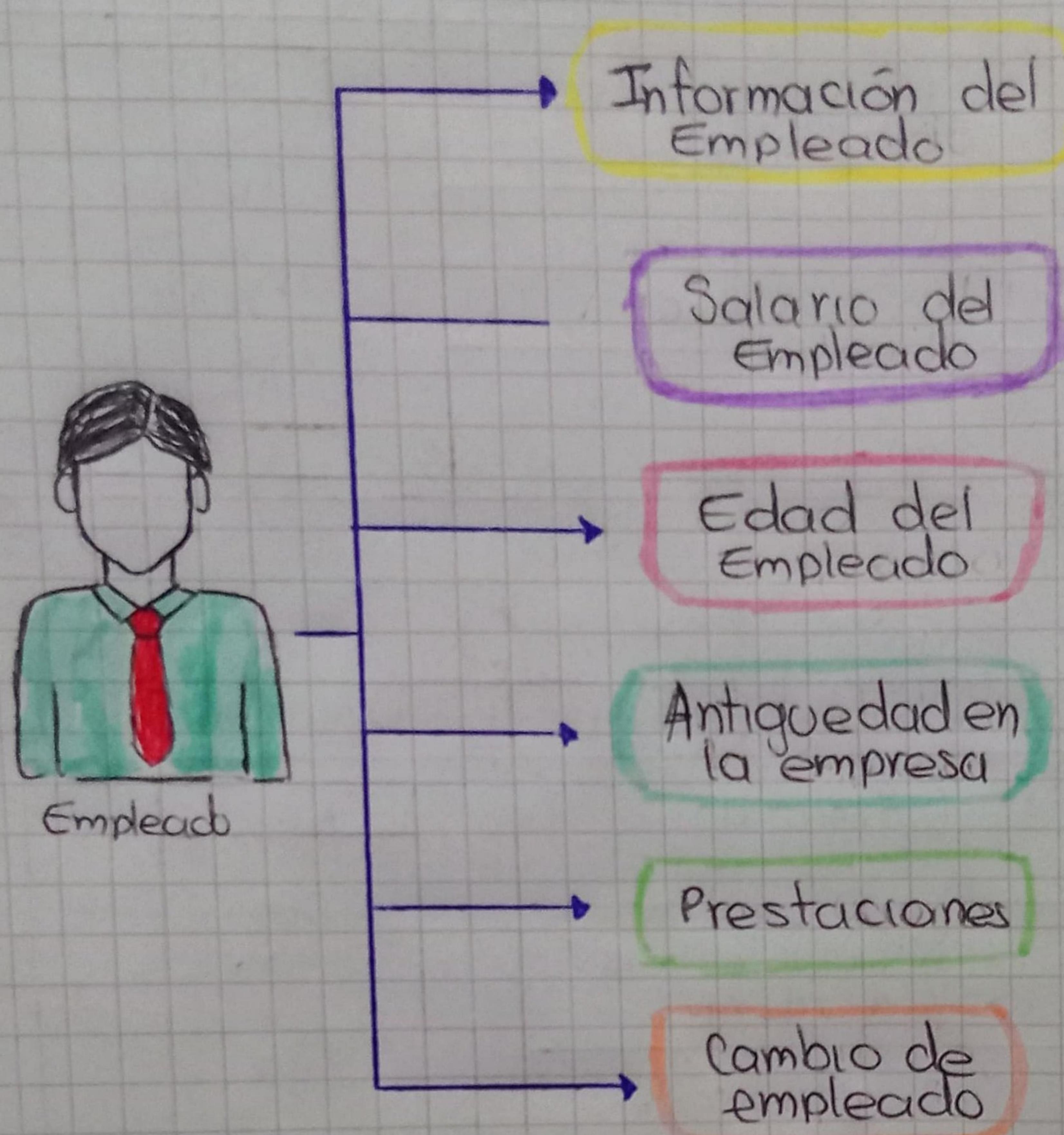
Nombre: Color

Descripción: La clase color representa el concepto del color que tendrán las líneas del contorno del triángulo y el color del relleno de este.

Punto de Reflexión: Que pasa si no identificamos bien las entidades del mundo?

Si no identificamos correctamente las entidades, nuestro código puede quedar desordenado, incompleto e incluso con errores con los que no podremos solucionar el problema correctamente.

1. Diagrama de casos de uso - Ejemplo el empleado



n. Debo plantar 2 ideas de proyecto (problemas solubles y algorítmicos).

Idea 1: Aplicación para registrar clases en una universidad, que deje a los usuarios registrar se, reservar clases disponibles y ver su horario académico.

Requerimientos Funcionales

① Nombre: Registro de usuario

Resumen: Permitir a los usuarios registrarse en el sistema con sus datos

Entradas: - Nombre
- Usuario
- Correo Electrónico (Institucional)
- Contraseña

Resultado: Un usuario registrado con perfil único

② Nombre: Registrar Clase

Resumen: Permitir a los usuarios seleccionar y registrar clases disponibles en los horarios disponibles

Entradas: - Código Estudiantil
- Clase Seleccionada (nombre y horario)

Resultado: Confirmación del registro para la clase seleccionada

③ Nombre: Cancelar Clase

Resumen: Permitir a los usuarios cancelar una clase previamente registrada

Entrada: - Código Estudiantil
- Código de la Clase

Resultado: Clase cancelada y disponibilidad para agregar otra.

- ④ Nombre: Visualizar horario
- Resumen: Mostrar a los usuarios el horario con las clases registradas
- Entradas: - Código estudiantil
- Resultado: Horario con clases registradas por el usuario

Idea 2: Aplicación para clasificar la basura en Reciclable, no reciclable y orgánica

Requerimientos Funcionales

- ① Nombre: Clasificar la basura
- Resumen: Permitir al usuario seleccionar el tipo de material del residuo
- Entradas: - Tipo de material - Nombre de residuo
- Resultado: El programa categoriza el tipo de residuo
- ② Nombre: Almacenamiento de Basura
- Resumen: Permitir al usuario visualizar en que contenedor se almaceno el residuo
- Entradas: - Nombre de residuo
- Resultado: Visualizar en que contenedor el sistema almaceno el residuo
- ③ Nombre: Repetir proceso o no?
- Resumen: Permitir al usuario seleccionar si almacenar más basura o no.
- Entradas: - Sí o No
- Resultado: Para "no" termina el proceso, para "Sí" se reinicia el proceso.

m. Observar nuevamente el modelo conceptual del ~~aboo~~ y escribe el nombre de cada una de las ~~clases~~ identificando sus respectivas variables (atributos) y funciones:

- Clase Empleado

- Atributos:

- Nombre: String
 - Apellido: String
 - Genero: int
 - Imagen: String
 - FechaNacimiento: String
 - FechaIngreso: String
 - SalarioBasico: double.

- Funciones:

- + darNombre(): String
 - + darApellido(): String
 - + darGenero(): int
 - + darImagen(): String
 - + darFechaNacimiento(): String
 - + darFechaIngreso(): String
 - + darSalarioBasico(): double
 - + CalcularEdad(): int
 - + cambiarSalarioBasico(NuevoSalario: double): void
 - + calcularAntiguedad(): int
 - + calcularPrestaciones(): double

- Clase Empleado

- Atributos:

- Emplados

- Funciones:

- + agregarEmpleado(Empleado nuevoEmpleado): void
 - + eliminarEmpleado(String Nombre, String Apellido): bool
 - + buscarEmpleado(String Nombre, String Apellido): Empleado
 - + listarEmplados():

Clase InterfazUsuario

- Atributos:

- Empresa: Empresa

- Funciones:

- + mostrarMenu(): void

- + mostrarInformacionEmpleado(): void

- + cambiarSalarioEmpleado(Nombre String, Apellido String, nuevoSalario double): bool

- + calcularYMostrarEdadEmpleado(Nombre String, Apellido String): Void o Int

- + calcularYMostrarAntiguedadEmpleado(Nombre String, Apellido String): Void o int

- + calcularYMostrarPrestacionesEmpleado(Nombre String, Apellido String): Void o double

- + cambiarEmpleado(): void