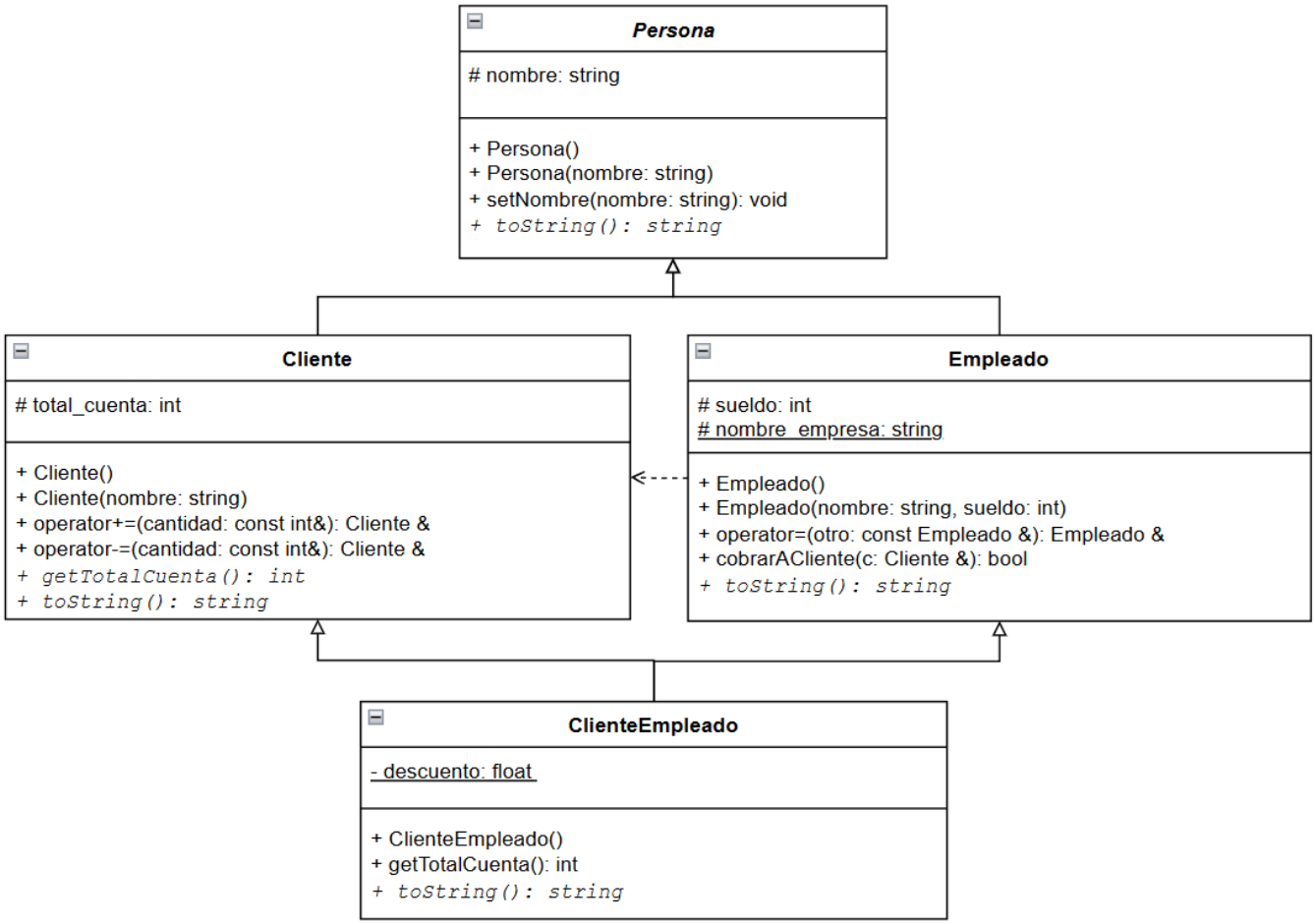
Escuela Técnica Superior de Ingeniería
Asignatura: Fundamentos de Programación II
Curso: Primero
Fecha: 25 de abril de 2025

Bloque 2. Debe comprimir todos los ficheros que utilice y entregar solamente un único fichero llamado Apellido_Nombre.zip con su primer apellido y primer nombre. **Se corregirán solamente los archivos que se ejecuten sin errores**. Comente todo aquello que produzca errores. Documente los ficheros. Descargue los ficheros iniciales.

Problema 1: (10.0 p) Usted ha sido contratado en una empresa que crea aplicaciones para restaurantes y hoteles. En su primer día, ha encontrado que el programador anterior había hecho un diagrama de clases y había codificado parte de una clase . Su trabajo es continuar desarrollando la aplicación a partir de los ficheros que se encuentran en prueba2.zip y del diagrama de clases mostrado a continuación:



Es obligatorio que cada clase tenga un par de ficheros (.h y .cpp). Cada clase se evalua de manera individual. Está prohibido agregar funciones y métodos que no se definan en el diagrama.

Clase Persona: (1 p.)

1. El fichero de encabezado (.h) se define según los atributos y métodos mostrados en el diagrama, además de definir las relaciones mostradas en el diagrama.
2. En el .cpp se deben implememententar todas las funciones posibles con sus comportamientos esperados, el nombre por defecto es "Desconocido".
3. La función toString() debe devolver un string con el nombre de la persona.

Clase Cliente: (3 p.)

Es el Cliente del restaurant, tiene un atributo total_cuenta que se utiliza para almacenar la valor de la cuenta total que debe pagar el Cliente.

1. El fichero de encabezado (.h) se define según los atributos y métodos mostrados en el diagrama, además de definir las relaciones mostradas en el diagrama.
2. En el .cpp se deben implememententar los 2 constructores utilizando las funciones de construcción en la clase superior que sean equivalentes, con la indicación adicional de que el atributo **total_cuenta** debe ser asignado a un valor de 0.
3. Los operadores de suma-asignación y de resta-asignación deben modificar el valor del atributo **total_cuenta** del Cliente y devolver dicho Cliente.
4. La función toString() debe devolver un string que posea el nombre de la persona y el valor de total_cuenta.

Clase Empleado: (3 p.)

Es el empleado de la empresa, posee un atributo sueldo que indica el sueldo de la persona y otro atributo que indica el nombre de la empresa. Se asume que el nombre de la empresa es la misma para todos los objetos tipo Empleado.

1. El fichero de encabezado (.h) se define según los atributos y métodos mostrados en el diagrama, además de definir las relaciones mostradas en el diagrama.
2. En el .cpp se deben implememententar los dos constructores, también construyendo las clases superiores,y asumiendo que el valor por defecto de sueldo es igual a 12000 euros.
3. El operador de asignación por copia debe copiar el sueldo, pero no el nombre, es decir que se utiliza para facilitar la generación de diferentes empleados que tienen el mismo sueldo.
4. La función cobrarACliente(c: Cliente &): bool recibe un cliente realiza lo siguiente:
   - Si el total_cuenta es mayor a 0, imprime en pantalla "[nombre] cobra al cliente [nombre de Cliente c] un total de [total_cuenta de Cliente c] euros", luego asigna el valor de total_cuenta del Cliente c a 0.0 y devuelve false.
   - En el resto de los casos, imprime en pantalla "La cuenta del cliente [nombre de Cliente c] es inválida o nula" y devuelve true. Es decir que la función devuelve true si hay errores.

   Reemplaze cada [·] por los atributos correspondientes de un Empleado o de un Cliente, respectivamente.
5. La función toString() debe devolver un solo string que contenga información del nombre, del sueldo y del nombre de la empresa de un empleado.

Clase ClienteEmpleado: (1.5 p.)

Corresponde a un Empleado que come en el mismo restaurant, por lo que es un Cliente. Esta clase posee un atributo que indica el descuento que recibe el empleado al ser cliente de la empresa. Por supuesto, se asume que el descuento es el mismo para todos y siempre es un valor flotante entre 0 y 1.

1. El fichero de encabezado (.h) se define según los atributos y métodos mostrados en el diagrama, además de definir las relaciones mostradas en el diagrama.
2. En el .cpp se debe implememementar el constructor por defecto, que debe construir las clases superiores.
3. La función getTotalCuenta devuelve la cuenta del cliente aplicando el descuento, es decir total_cuenta * (1 - descuento), sin cambiar el valor de total_cuenta.
4. La función toString() debe devolver un solo string que contenga información del Cliente, del Empleado y, además, del descuento a aplicar.

Fichero main.cpp (1.5 p.)

Plantilla: Crear una función tipo plantilla que se base en el siguiente prototipo:

```
ostream & operator<<(ostream & os, Cliente & c)
```

en donde se debe mantenter el ostream y generalizar Cliente a una clase cualquiera. Esta función debe devolver un ostream , el cuál posee el toString del objeto c.

Atributos estáticos: Asignar dentro del main, el nombre de la empresa de los objetos de la clase Empleado a "ULoyola" y el descuento de los objetos tipo ClienteEmpleado a "0.15".

Programa Principal: Cree un programa principal ordenado y legible **(0.1 p.)** que cumpla con las siguientes instrucciones.

1. Cree un objeto c1 de la clase Cliente con su nombre.
2. Utilizando alguno de los operadores, sume 10 euros al total_cuenta de c1.
3. Lo sentimos, la cuenta incluia algo que el cliente c1 no ha solicitado. Utilizando el operador de resta-asignación, reste 3 euros al total_cuenta de c1.
4. Realice un cout de c1.
5. Cree un objeto e1 de la clase Empleado con el nombre inicial igual a "Eduardo" y sueldo igual a 15000.
6. Cree un objeto e2 de la clase Empleado con el constructor por defecto.
7. Realice un cout de e1 y otro de e2.
8. Realice una asignación por copia tal que e1 sea asignado a e2.
9. Realice un cout de e2.
10. Cree un objeto ce1 de la clase ClienteEmpleado con el constructor por defecto.
11. Defina el nombre de ce1 igual a "Alicia" usando el setter de la clase Persona.
12. Utilizando el operador de suma-asignación aumente la cuenta de ce1 a 15 euros.
13. Imprima en pantalla lo que devuelve getTotalCuenta() de ce1.
14. Realice un cout de ce1.

15. Imprima por pantalla el resultado de que el Empleado e1 use la función cobrarACliente siendo c1 el argumento de entrada.
16. Imprima por pantalla el resultado de que el ClienteEmpleado ce1 use la función cobrarA-Cliente siendo c1 el argumento de entrada.