PROCESO GESTIÓN DEL TALENTO HUMANO

FORMATO INFORME MENSUAL EJECUCIÓN CONTRACTUAL

Barranquilla, Abril de 2020

Señor (a)
MANUEL GREGORIO HORMECHEA LANCE
SUPERVISOR(A) CONTRATO No. **1333805**
Coordinador de Automatización
Automatización
Barranquilla

Asunto: Informe mensual de ejecución contractual Mes Abril del año 2020

Referencia: No 1333805 del año 2020

RICARDO ENRIQUE CANTILLO CARRILLO, identificado con la cédula de ciudadanía No. 72.201.581 de Barranquilla, en mi calidad de Contratista del SENA, en Automatización, en cumplimiento del Contrato de Prestación de Servicios de la referencia, a continuación presento el Informe de actividades realizadas en el mes objeto de cobro.

Valor y forma de Pago: c). un pago correspondiente al mes de Abril de 2020 por valor de (3.278.181). Los honorarios serán pagados por el SENA al contratista de acuerdo al cronograma definido por la Dirección Administrativa y Financiera de la Dirección General, en la cuenta de Ahorros No. 0098000200063852 BBVA, cuyo titular es el Contratista.

Plazo: Será hasta el 30 de Junio de 2020.

OBJETO:
Prestación de servicios profesionales como instructor para impartir formación profesional titulada y complementaria en el área de teleinformática - análisis y desarrollo de sistemas de información de acuerdo a la programación académica del SENA Centro Nacional Colombo Alemán

CLÁUSULA (No. de cláusula donde se encuentren las obligaciones del contratista)

No	Obligaciones	Acciones realizadas	Evidencias
1	Programar las actividades de enseñanza, aprendizaje y evaluación de conformidad con los de formación y el calendario institucional y el Manual de procedimientos para la ejecución.	REVISIÓN DE ACTIVIDADES, INTERACCIÓN VIRTUAL CON APRENDICES DE LA FORMACIÓN, ENVÍO DE CORREOS DE FOROS DE BUZÓN DE DUDAS E INQUIETUDES. LABORATORIO DE PROGRAMACIÓN, DEL	NODO TIC, de los aprendices de los programas de ADSI, realización de los Laboratorios de Rest y Esquemas de SQL. Programación con

		NODO TIC DEL CNCA - PISO 6. SALA ADSI 2 DE PROGRAMACIÓN.	frameWorks SpringBoot

A continuación relaciono los desplazamientos que realicé previo a la presentación de este informe. Una vez finalizado cada desplazamiento presenté al ordenador del gasto el informe en el Formato Informe Legalización Desplazamiento Contratista GTH-F-087, en el que se describieron las actividades desarrolladas y los resultados de cada desplazamiento. Cada informe cuenta con el visto bueno del Supervisor.
Se lista a continuación el soporte de la legalización de los desplazamientos realizados, los cuales forman parte integral del presente informe de ejecución contractual.

ITEM	No DE LA ORDEN DE VIAJE	LUGAR DE DESPLAZAMIENTO	FECHAS DE DESPLAZAMIENTOS	
			INICIO	FINALIZACIÓN

Nota 1: *Por cada desplazamiento que haya realizado el contratista, adjuntará el respectivo informe que la soporte. En caso de haber realizado el desplazamiento en fecha posterior a la presentación del informe de ejecución contractual, deberá reportarlo en el siguiente informe de ejecución contractual.*

Para el trámite de la cuenta me permito adjuntar: Documentos enunciados como evidencias del cumplimiento de las obligaciones contractuales y los desplazamientos realizados y el No. de la planilla, operador y periodo. (Decreto Ley 2106 de 2019 – " *Decreto Ley Antitrámites"*)

Evidencias en ( ) folios

Cordialmente,

RICARDO ENRIQUE CANTILLO CARRILLO
Contratista
C.C. No. 72.201.581

Recibí a satisfacción:

MANUEL GREGORIO HORMECHEA LANCE
Supervisor(a) Contrato 1333805 de 2020
Coordinador de Automatización

LABORATORIO 10.
Programación Orientada a Objetos en Java

GUÍA DE LABORATORIO Nº 10

Actividad de Proyecto: **CODIFICAR LOS MÓDULOS DEL SISTEMA DE INFORMACIÓN**

Estructura de contenidos.

1. Introducción. 2

2. Objetivos. 2

3. Consideraciones. 3

4. Procedimiento. 3

5. Evidencias a entregar. 8

GLOSARIO 9

RECURSOS BIBLIOGRÁFICOS 10



1. INTRODUCCIÓN

En el presente laboratorio desarrollará una serie de ejercicios de Programación Orientada a Objetos (POO), empleando el lenguaje de programación JAVA, estos ejercicios le ayudarán a aplicar los conceptos de la POO en contextos reales.

Con el desarrollo de aplicaciones orientadas a objetos el Analista y Desarrollador de Sistemas de Información está preparado para sacar el máximo provecho a las plataformas y lenguajes de programación más modernos y utilizados en la industria del desarrollo de software actual.

2. OBJETIVOS

- Aplicar el concepto de Abstracción en el desarrollo de aplicaciones de Software Orientado a Objetos.
- Aplicar el concepto de Encapsulación en el desarrollo de aplicaciones de Software Orientado a Objetos.
- Aplicar el concepto de Herencia en el desarrollo de aplicaciones de Software Orientado a Objetos.
- Seguir un diseño definido y aplicarlo en el código de programas orientados a objetos.
- Comprender la diferencia entre los modificadores de acceso en Java.
- Definir métodos estáticos en el desarrollo de aplicaciones orientadas a objetos.

3. CONSIDERACIONES

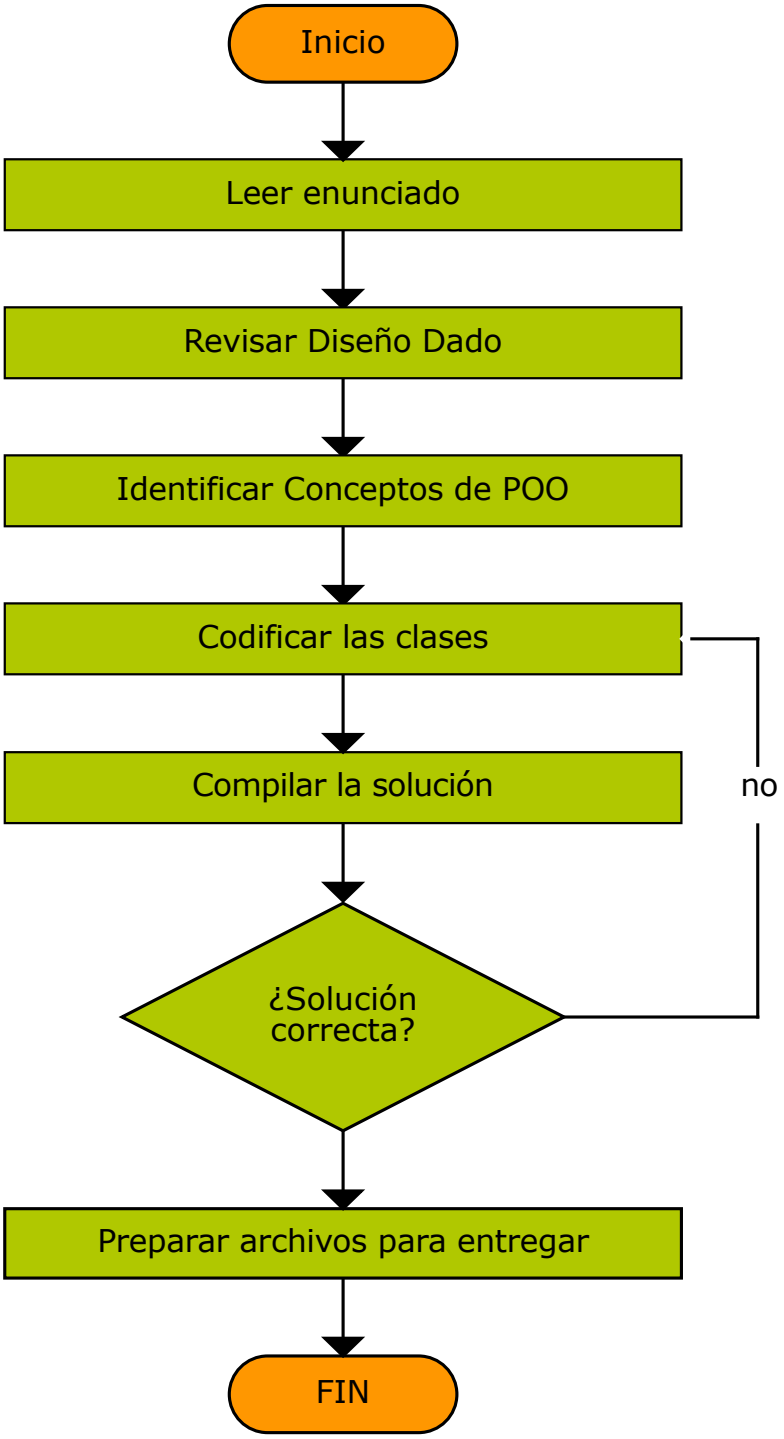
Ítem	Descripción
Soporte Teórico	Para el desarrollo del laboratorio se recomienda la lectura y seguimiento del objeto de aprendizaje: • Programación Orientada a Objetos Aplicada.
Productos requeridos	• Enunciados de los ejercicios a desarrollar disponibles en este mismo documento. • API de Java
Herramientas SW	El laboratorio puede ser desarrollado en cualquier IDE (Ambiente de Desarrollo Integrado) para Java tal como: • Netbeans • JCreator • BlueJ • Eclipse

4. PROCEDIMIENTO

Para el desarrollo de cada uno de los ejercicios tenga en cuenta el siguiente procedimiento:

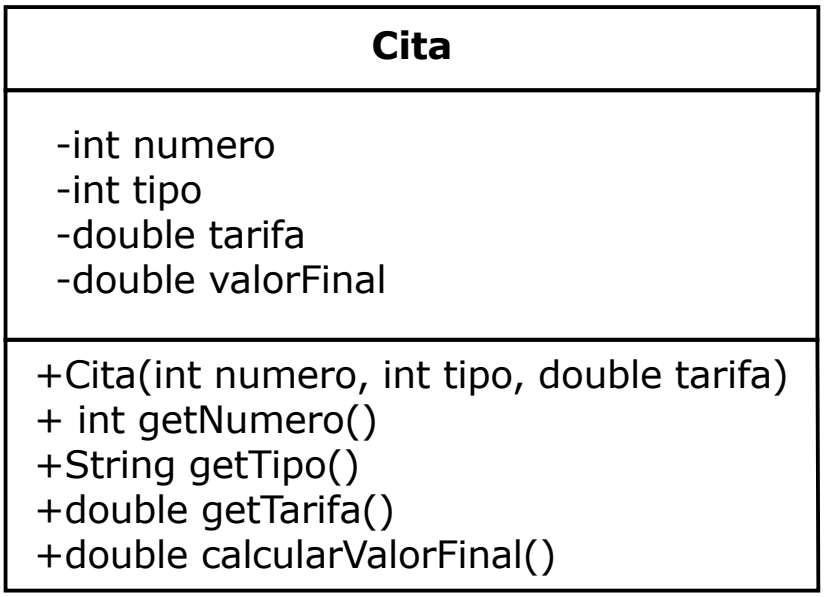
1. Lea completamente el enunciado de cada ejercicio hasta comprender con claridad lo que se está pidiendo.
2. Revise el diseño dado e identifique los conceptos de la Programación Orientada a Objetos que debe aplicar.
3. Codifique cada ejercicio propuesto documentando las acciones incorporadas.
4. Compile la solución.
5. Realice las pruebas requeridas para asegurar su correcto funcionamiento.
6. Si esta correcto continue con el punto 7, si no volver al paso 3.
7. Guarde cada ejercicio en una carpeta independiente con todos los archivos requeridos para su ejecución.

DIAGRAMA


Inicio
Leer enunciado
Revisar Diseño Dado
Identificar Conceptos de POO
Codificar las clases
Compilar la solución
¿Solución correcta?
no
Preparar archivos para entregar
FIN

EJERCICIOS:

1. Cree y compile la siguiente clase:



Notas:

- El método getTipo() debe retornar un String que puede ser "especialista" o "general", será general si la cita es tipo 1, 2 o 3 y será especialista si el tipo de cita es 4 o 5.

- El valor final de la cita depende de la tarifa y el tipo de cita, las citas de tipo "especialista" tienen un incremento del 50% sobre la tarifa y las citas de tipo "general" tiene un descuento del 50% sobre la tarifa.

Cree la clase Principal.java donde solicite al usuario los datos para crear una cita (objeto de la clase Cita) y a continuación presente los siguientes mensajes:

El número de la cita es: XXXXXX
Esta cita es de tipo: XXXXXXXX
Su tarifa normal es: XXXXXXXX
Pero por ser de tipo XXXXXXXX queda con un valor final de XXXXXXXX

2. Cree y compile la siguiente clase:

Bus
-String placa -int capacidadPasajeros -double precioPasaje -int pasajerosActuales - int totalPasajeros
+Bus (String placa, int capacidad, double pasaje) +String getPlaca() +int getCapacidad() +double getPrecioPasaje() +int getPasajerosTotales() +int getPasajerosActuales() +void subirPasajeros(int pasajeros) +void bajarPasajeros(int pasajeros) +double getDineroAcumulado()

Notas:

- getPlaca: Obtiene la placa del bus
- getCapacidad: Obtiene la capacidad de pasajeros del bus
- getPrecioPasaje: Obtiene el valor del pasaje
- getPasajerosActuales: Obtiene el número de pasajeros que se encuentra en el bus
- getPasajerosTotales: Obtiene el número de pasajeros que se han subido al bus
- subirPasajeros: sube una cantidad de pasajeros al bus, teniendo en cuenta que no sobrepase la capacidad del bus.
- bajarPasajeros: baja una cantidad de pasajeros del bus, teniendo en cuenta que no baje más pasajeros de los que realmente existen en el bus.
- getDineroAcumulado: Obtiene todo el dinero que ha acumulado el bus teniendo en cuenta todos los pasajeros que se han subido y el valor del pasaje.

Cree y compile la clase Principal.java donde instancie un objeto de la clase Bus y pruebe cada uno de sus métodos.

3. Cree y compile la siguiente clase:

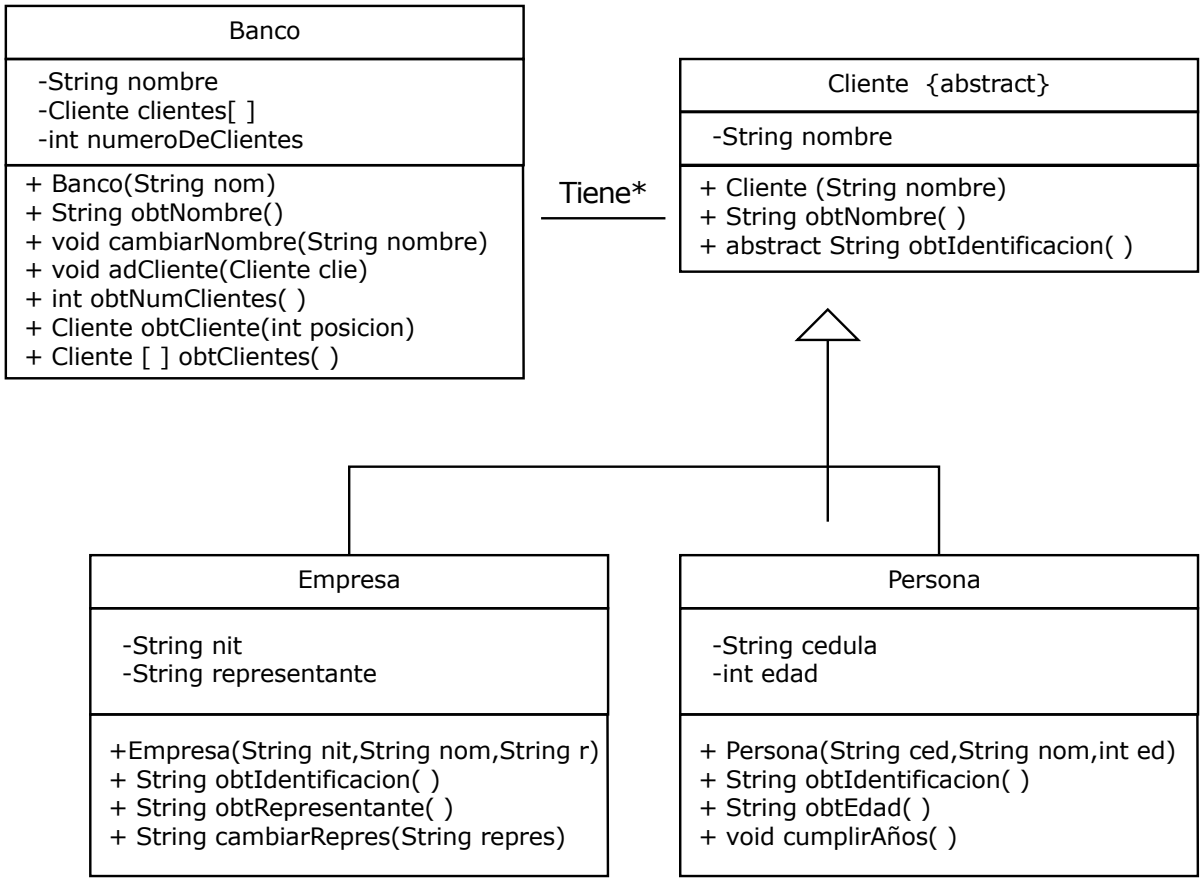
Cuenta
- String numero - String tipo - double saldoActual - double saldoMinimo
+ Cuenta(int numero, String tipo, double valorInicial) + void consignar (double monto) + void retirar (double monto) + double getSaldo() + double getSaldoMinimo() + double getCapacidadCredito()

Notas:

- Las cuentas solo pueden ser de tipo "ahorro" o "crédito"
- El saldo mínimo de una cuenta es el 10% del valor inicial con que se crea la cuenta.
- Consignar (double monto): incrementa el dinero de la cuenta con base en el monto de dinero que se desea consignar.
- Retirar (double monto): retira el monto dado de la cuenta siempre y cuando la cuenta no quede con un saldo inferior al saldo mínimo, si esto ocurre, solo se retirará el dinero autorizado.
- getSaldo(): obtiene el saldo actual de la cuenta.
- getSaldoMinimo(): obtiene el saldo mínimo de la cuenta.
- getCapacidadCredito(): Para cuentas de ahorro, la capacidad de crédito es igual a la diferencia entre al saldo actual y el saldo mínimo y para cuentas de crédito, la capacidad de crédito es 3 veces el saldo actual.

Cree y compile la clase Principal.java y en ella cree un objeto de la clase Cuenta y pruebe cada uno de sus métodos.

4. A partir del siguiente diseño, cree las clases correspondientes y su respectivo código para generar los listados que se solicitan a continuación:



a. Todos los nombres de los clientes que tiene el banco
b. Todas los nombres y cédulas de las personas clientes del banco
c. El nombre y representante de cada Empresa
d. Nombres de los clientes menores de edad
e. El nombre y la edad del cliente más Joven
f. El nombre y la edad del cliente más viejo

5. EVIDENCIAS

Carpeta comprimida con el nombre laboratorio10 y en su interior subcarpetas (una subcarpeta por cada ejercicio) con los respectivos archivos .java para su ejecución.

GLOSARIO

POO: Programación Orientada a Objetos.

Abstracción: Característica de la programación orientada a objetos que permite centrar la atención en los elementos necesarios e importantes para el desarrollo de un programa.

Encapsulación: Característica de la POO que se da en un programa se da cuando se definen los atributos de una clase con el nivel de acceso más restrictivo para que el acceso o la modificación de los atributos de esa clase Solo sea posible a través de sus métodos.

Herencia: Característica de la POO que permite a una clase adquirir el comportamiento de otra.

RECURSOS BIBLIOGRÁFICOS

Oracle (2013). Java 2 Platform Standard Edition 5.0. API Specification, consultada en Julio de 2013 y disponible en http://docs.oracle.com/javase/6/docs/api/

LABORATORIO 10	Programación Orientada a Objetos en Java
Desarrollador de contenido Experto temático	**Andrés Julián Valencia Osorio**
Asesor Pedagógico	**Rafael Neftalí Lizcano Reyes**
Productor Multimedia	**José Jaime Luis Tang Pinzón**
Líder expertos temáticos	**Ana Yaqueline Chavarro Parra**
Líder línea de producción	**Santiago Lozada Garcés**

Atribución, no comercial, compartir igual

Este material puede ser distribuido, copiado y exhibido por terceros si se muestra en los créditos. No se puede obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de licencia que el trabajo original.

Creative Commons

BootCamp Virtual de API Rest. ADSI CNCA

Item	Tema	Apoyo Virtual - Recurso
1	Introducción y conceptos API REST en Java	Sesión 1. En línea con el Tutor.
2	Fundamentos de Java EE y Creación proyectos Maven en **NetBeans y Eclipse.**	https://www.youtube.com/watch?v=nhPuhgkinps https://www.youtube.com/watch?v=k5RKM3r2Srk Sesión 2. En línea con el tutor.
3	Crear Mi primer API Rest. Agregando Dependencias en mis Archivo **pom.xml**	Sesión 3. En línea con el tutor.
4	Conectar mi API a una Base de Datos MySql	Sesión 4. En línea con el tutor.
5	Trabajando en mi Back-End, Migración de Datos, y consultas en el Controlador.	Sesión 5. En línea con el tutor.
6	Envío de Parámetros a mi API Rest**. @PathParam @QueryParam**	Sesión 6. En línea con el tutor.
7	Verbos REST **@GET @POST @PUT @DELETE**	Sesión 7. En línea con el tutor.
8	Probar API Rest **- Postman**	Sesión 8. En línea con el tutor.
9	**JPA** para agilizar la comunicación con la Base de Datos.	Sesión 9. En línea con el tutor.
10	**Spring-Boot** API Rest Framework.	Session 10. En línea con el tutor.
11	**Spring-Boot** API Rest Framework.	Session 11. En línea con el tutor.
12	Proyecto Completo API Rest	Session 12. En línea con el tutor. -WorkOut
13	Proyecto Completo API Rest	Session 13. En línea con el tutor. -WorkOut
14	Proyecto Completo API Rest	Session 14. En línea con el tutor. -WorkOut
15	Proyecto Completo API Rest	Session 15. En línea con el tutor. -WorkOut

Sólo están Publicados / Seleccionados los materiales de este fin de semana, a este cronograma se le Seguirán Los adicionando links de cada tema, y/o grabaciones de las sesiones.