**TECNOLÓGICO DE COSTA RICA**

**SEDE REGIONAL SAN CARLOS.**

##### CARRERA:

INGENIERÍA EN COMPUTACIÓN

##### CURSO:

ALGORITMOS Y ESTRUCTURAS DE DATOS 1

##### TEC@WORKSHOP-MANAGER

##### PROFESORA:

LORENA VALERIO SOLÍS.

**ELABORADO POR:**

LUIS ALONSO VEGA BRENES.

YÁSER AUGUSTO CASTILLO BRAVO.

***SANTA CLARA, SEPTIEMBRE DE 2010.***

Descripción del problema

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Tomando en cuenta los aspectos anteriores y las características del curso, se nos presenta un reto muy importante, diseñar e implementar un sistema que sea capaz de manejar las actividades dentro de un taller mecánico, con una interfaz agradable al usuario y sencilla de utilizar, sin embargo para poder realizar todo esto fue necesario conocer algunos aspectos básicos de logística dentro de un taller mecánico que a la postre servirían para poder desarrollar la aplicación de una manera eficaz y eficiente.

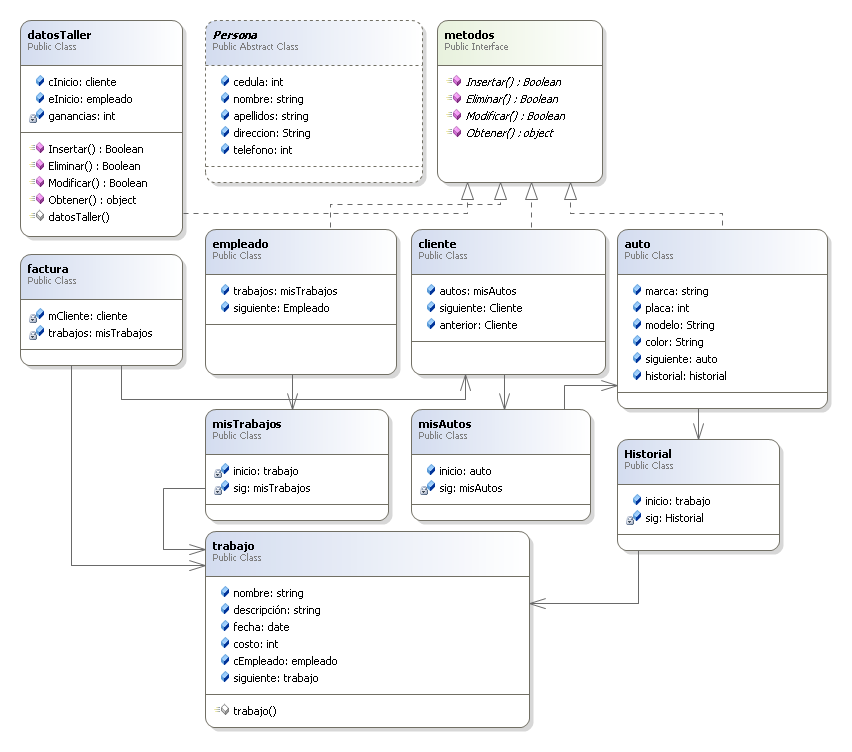
Otro de los aspectos importantes de este proyecto es la utilización de las estructuras de datos dinámicas conocidas como listas, para el manejo correcto de los datos, utilizando listas simples, dobles y circulares. Básicamente el funcionamiento del sistema se divide en ciertas partes:

* **Inserciones, modificaciones y borrado**: el sistema debe ser capaz de permitir agregar, modificar y borrar: autos, empleados, y clientes.
* **Asignaciones:** de autos a clientes, trabajos a autos, y trabajos a empleados.
* **Validaciones:** un auto no puede estar asignado a más de un cliente, un empleado no puede tener más de un trabajo asignado durante un solo día.
* **Facturación:** a una fecha que digita el usuario. Se deben generar todas las facturas e insertarlas en una lista temporal, debe ser una sola factura por cliente, en ella debe incluir todos los trabajos de todos los vehículos de un mismo cliente que se venzan a esa fecha. Observar las facturas de una a una, dar la opción de cancelar (pagar) la factura. En caso de que se pague la factura, se deben borrar todos los trabajos facturados en ella automáticamente y pasarlos al historial de cada auto.
* **Manejo de Consultas:** 
  + 1. Mostrar de los trabajos pendientes de un auto, consultar por número de placa.
    2. Generar y mostrar factura al cliente una vez finalizados los trabajos pendientes por fecha.
    3. Mostrar los todos trabajos pendientes por día.  
       3.1 Mostar lo recaudado por día.
    4. Listar por el número de cédula de cada cliente cuántos y cuáles autos tienen en el taller y que trabajos se le está haciendo al auto.
    5. Listar todo el historial de trabajos de un auto x.
    6. Listar los trabajos asignados a cada empleado.
    7. Mostrar cual es el empleado por mes (el que ha generado mayor dinero al taller) de un año X.
    8. Listar el tiempo de días que están ociosos los empleados por mes/año.
    9. Listar los 10 autos que han permanecido más tiempo de mayor a menor en el taller.
    10. Listar los 10 clientes más frecuentes (en cantidad de veces) del taller, también de mayor a menor.

Otros aspectos importantes a destacar son la utilización de las interfaces de Java (interface), para definir los métodos a usar para las listas. La implementación (implements), utilizada para contener todos los métodos de la interfaz tanto de búsqueda como de modificado. La utilización de paquetes para dividir las clases que se utilizan para un objetivo específico.

Solución del Problema

**Diagrama de clases**



# Realización general de la aplicación

Para la construcción del programa, se utilizó el concepto de listas. Manejando los métodos básicos para construir distintos modelos basados en uniones de varios elementos con propiedades de listas, se logra obtener verdaderas estructuras lineales que permiten acceder a los datos que contienen de distintas formas (lineales simples, circulares, bidireccionales).

Específicamente, el sistema utiliza cuatro listas principales, que contienen datos de empleados, clientes, autos y trabajos. Cada elemento de las listas, se puede relacionar con uno del mismo tipo, con otro tipo de las principales o de una sub-lista. De esta manera, obtenemos una organización compleja de objetos relacionados entre sí que conforman la base de la información del taller.

Para las listas mencionadas anteriormente, se crearon métodos de inserción, muy similares entre sí, a excepción de las listas circular y bidireccional, que contienen pequeñas variaciones. Además, métodos de modificar y eliminar que buscan un objeto mediante alguna identificación (código, cédula, placa, etc.) y modifican sus datos, o lo eliminan de la lista.

Por otra parte, cada elemento de algunas listas principales, se provee una sub-lista propia que contiene elementos secundarios que proporcionan direccionamiento a otros elementos principales.

La base del programa se encuentra en una clase (llamada “datosTaller”), la cual proporciona el inicio de las listas principales y los métodos relacionados con ellos. Además de varia información adicional que ayuda a mantener un control del taller (por ejemplo, las ganancias totales).

Aparte de la estructuración de listas, el programa presenta una interfaz agradable (casi intuitiva) para el usuario, que permite el manejo de la información con una total abstracción del funcionamiento interno. Los clientes, autos y empleados se manejan desde tres ventanas distintas que permiten realizar las acciones normales de estos objetos. Además encontramos la ventana de lista visual, que permite ver de una forma lineal los datos que maneja actualmente el taller de una forma virtual. Una ventana de consultas está disponible para la visualización de análisis más específicos relacionados con el taller.

## Lógica interna

Para la creación del programa se establecieron tres paquetes, de los cuales dos funcionan como las bases o columnas del programa. Una de ellas es el de clases y métodos, mientras que el otro posee todo el diseño de interfaz que se muestra al usuario.

La siguiente tabla presenta las clases del paquete de listas.

|  |  |
| --- | --- |
| **Nombre** | **Breve descripción** |
| Auto | Información de autos, historial asociado. |
| Cliente | Información de clientes, autos asociados. |
| datosTaller | Base del taller, listas principales, métodos generales. |
| Empleado | Información de empleado, trabajos asociados. |
| Factura | Trabajos sin pagar para un cliente. |
| Historial | Trabajos realizados (completados) en un auto. |
| misAutos | Autos asignados a un cliente. |
| misClientes | Lista de clientes. |
| misTrabajos | Trabajos asignados a un empleado. |
| Persona | Base de una persona: datos personales. |
| Trabajo | Información de un trabajo. |

## Consultas y Operaciones

* Para insertar, modificar y eliminar clientes, empleados y autos, se utilizaron métodos básicos mencionados anteriormente.
* Asignar autos a clientes, sin hacer asignaciones dobles: Cada auto posee un puntero a su cliente asociado. Al asignar, se buscan los autos que no tengan ningún cliente asociado.
* Asignar trabajos a los autos: Se realiza desde las opciones del empleado, al agregar un trabajo, se debe seleccionar el auto al que se le realiza el mismo.
* Asignar trabajo a empleado: Ver ítem anterior.
* Facturar por fecha: Se buscan los trabajos sin cancelar para cada cliente y se muestran solo los que cumplen con el filtro de fecha.
* Permitir cancelar cada factura: Se muestran cada trabajo por usuario y se permite la cancelación de una a una o de todas las del mismo cliente.
* Mostrar trabajos pendientes de un auto: Se selecciona el auto y se buscan los trabajos que no estén completados para ese auto.
* Mostrar factura al cliente: Similar al de permitir cancelar cada factura.
* Mostrar lo recaudado por día: Se buscan los trabajos que cumplan con la fecha dada y se calcula el total de ganancias para ese día tomando en cuenta el costo de cada trabajo.
* Listar los autos de cada cliente y sus trabajos: Se recorre la lista de autos para cada cliente y, a su vez, la de historial para cada uno de los autos que posee.
* Listar el historial de un auto: Se recorre la sub-lista de historial y se imprime cada trabajo en ella.
* Listar los trabajos asignados a cada empleado: Se muestra cada elemento en la sub-lista de mis trabajos del empleado dado.
* Mostrar empleado por mes: Se revisa para cada empleado cuánto ha generado revisando los trabajos que coincidan con la fecha dada, al final se busca el que obtuvo mayor ganancias.
* Listar días ociosos de los empleados: Se calcula los días que tiene trabajos cada empleado y se le resta a la cantidad total de días que se deben tomar en cuenta (por mes o por año).
* Listar los diez autos con más tiempo en el taller: Se comprueba la fecha actual con la fecha de ingreso de cada auto y se buscan los diez con mayor tiempo.
* Listar los diez clientes con mayor cantidad de trabajos asignados: Se calcula el número de trabajos asignados a cada empleado, se comparan todos los resultados y se muestran únicamente los diez con la mayor cantidad.

Análisis de Resultados

|  |  |
| --- | --- |
| Operación | Estado |
| *Insertar Autos* | *Completado* |
| *Insertar Clientes* | *Completado* |
| *Insertar Empleados* | *Completado* |
| *Modificar Clientes* | *Completado* |
| *Modificar Empleados* | *Completado* |
| *Modificar Autos* | *Completado* |
| *Borrar Empleados* | *Completado* |
| *Borrar Clientes* | *Completado* |
| *Borrar Autos* | *Completado \** |
| *Asignar Trabajos a Autos* | *Completado* |
| *Asignar trabajo a empleado* | *Completado* |
| *Asignar autos a clientes* | *Completado* |
| *Facturación* | *Completado* |

*Análisis de Resultados funciones base del sistema.*

\* Eliminar auto no quita el puntero de cada cliente

|  |  |
| --- | --- |
| *Consulta* | *Estado* |
| *Mostrar Trabajos pendientes de un auto* | *Completado* |
| *Generar y mostrar factura de un cliente* | *Completado* |
| *Mostrar los trabajos pendientes de un día* | *Completado* |
| *Mostrar lo recaudado por día* | *Completado* |
| *Listar autos en el taller, y los trabajos de éstos* | *Completado* |
| *Listar todo el historial de trabajos en un auto* | *Completado* |
| *Listar todos los trabajos asignados a un empleado* | *Completado* |
| *Mostrar empleado del mes* | *Completado* |
| *Listar el tiempo de ocio de los empleados* | *Completado* |
| *Listar los diez autos que han permanecido más tiempo de mayor a menor en el taller* | *Completado* |
| *Listar los diez clientes más frecuentes* | *Completado* |

*Análisis de Resultados de consultas.*

Conclusiones y Recomendaciones

1. El manejo de los datos se facilita al utilizar una clase para contener todos los métodos y los punteros de las clases, esto con el fin de no perder los datos en los diferentes procesos realizados dentro de la aplicación y poder aplicar el concepto de estructuras de datos tipo listas.
2. La implementación y el diseño de una aplicación para su aplicación dentro de un taller mecánico, implica pensar en qué tipo de usuario va a hacer uso de la aplicación, y que grado académico pueda tener, es decir la aplicación debió ser implementada para cualquier tipo de usuario.
3. La utilización de paquetes (packages) facilita la creación de la aplicación en dos capas distintas: la separación de los métodos y la interfaz de usuario.
4. En relación con el tiempo de entrega del proyecto es un periodo comprensible y aceptable, el problema está básicamente en cómo se aprovecha este tiempo, que quizá no fue de la mejor manera, dada la prórroga de una semana las posibilidades de completar el proyecto aumentaron y el tiempo fue bien aprovechado.
5. Respecto al tiempo de explicación en clase, en opinión personal se han impartido las lecciones de la manera adecuada y con el tiempo apropiado para el entendimiento correcto de los conceptos.

Referencias Bibliográficas

1. Example Depot (2009). *Adding and removing an item in a JList component*. Consultado en septiembre, 06, 2010 en <http://www.exampledepot.com/egs/javax.swing/list_ListAddRem.html>.
2. Java Forums (2007*). JFrame at Center Screen*. Consultado en septiembre, 12, 2010 en <http://www.java-forums.org/awt-swing/3491-jframe-center-screen.html>.
3. The Java tutorials (2010). Creating a GUI With JFC/Swing. Consultado en septiembre , 18, 2010 en <http://download.oracle.com/javase/tutorial/uiswing/TOC.html>.