

PROYECTO PRÁCTICO NO. 2 PERSONAL URL

19 NOVIEMBRE 2021

Universidad Rafael Landívar

Cristian Fernando González Cuyun 1183221

Julio Anthony Engels Ruiz Coto 1284719

Hugo Roberto Florián Castañeda 1236521



Universidad
Rafael Landívar
Tradición Jesuita en Guatemala

I. INTRODUCCIÓN

Durante el curso hemos estado trabajando acerca del concepto de listas, principalmente en la creación de estas, así como los distintos métodos que existen para ordenarlas y encontrar el índice de los valores dentro de los mismo, ahora se posee los conocimientos teóricos y prácticos para poder aplicar en un panorama real estos conceptos vistos durante el curso.

Como parte del último proyecto práctico de la clase de Programación Avanzada, se dio un enunciado que englobaba todo lo aprendido en el curso presente este enunciado nos dice lo siguiente:

“A través de un sistema automatizado se quieren controlar los datos de todo el personal perteneciente a la Universidad Rafael Landívar, ya sea estudiante o trabajador. De todo el personal se conoce el nombre, primer y segundo apellidos y el DPI. Los estudiantes, en particular, pueden ser de pregrado o de postgrado. En cada caso se conoce el carnet, la facultad en la que estudia, año en que ingresó y la nota de cada curso que recibe. Para los estudiantes de postgrado se conoce el tipo de estudio que realiza (Maestría o Doctorado). Por otro lado, del personal trabajador se conoce el salario, fecha de inicio (año, mes y día) y el departamento en el que labora. Entre los trabajadores existen docentes y no docentes. De los primeros se conoce las asignaturas que imparte, mientras que de los no docentes se conoce el cargo ocupacional.”

Para poder realizar esto se creó un programa usando C++, Windows Forms, que por medio de una lista doblemente enlazada se realizarían los siguientes incisos.

- Gestionar estudiantes de pregrado y posgrado.
- Gestionar trabajadores docentes y no docentes.
- Devolver el nombre completo de una persona dado el DPI
- Mostrar un listado con los estudiantes de Doctorado, Pregrado o Postgrado y exportar la información a un archivo CSV ordenado por el primer apellido y debe guardarse con el siguiente formato <carnet>,<apellidos>,<nombre>,<año de ingreso>,<DPI>,<facultad>,<curso y nota 1>,<curso y nota 2>,...<curso y nota n>
- Mostrar un listado con los docentes y exportar la información a un archivo CSV ordenado por el primer apellido y debe guardarse con el siguiente formato <código de empleado>,<apellidos>,<nombre>,<DPI>,<fecha de inicio>,<salario>,<curso 1>,<curso 2>,...<curso n>
- Mostrar un listado con los datos del personal trabajador no docente y exportar la información a un archivo CSV ordenado por el primer apellido y debe guardarse con el siguiente formato <código de empleado>,<apellidos>,<nombre>,<DPI>,<fecha de inicio>,<cargo ocupacional>,<salario>
- Calcular el salario promedio de todos los trabajadores (docentes, no docentes o ambos).

- Determinar la cantidad de estudiantes en Maestría o por facultad.
- Dado un curso determinar la cantidad de docentes que lo imparten y un listado con los estudiantes que lo reciben ordenados alfabéticamente por el nombre o por nota
- Dado un estudiante de pregrado o postgrado conocer su promedio
- Dado un estudiante de pregrado conocer su nota más alta y el curso al que corresponde dicha nota
- Dada una facultad, conocer el alumno de pregrado con mejor promedio.

II. ANÁLISIS

Gestionar estudiantes de pregrado y posgrado.

ENTRADA: Los datos individuales de cada uno de los estudiantes presente en la lista.

PROCESO: Se crea una clase que posee los atributos y procesos de la lista, entre las cuales se encuentran las necesarias para agregar elementos, eliminarlos, modificarlos o mostrar la lista.

SALIDA: La lista de los alumnos con sus elementos, con todas las funciones necesarias para gestionar la lista y sus atributos.

Gestionar trabajadores docentes y no docentes.

ENTRADA: Los datos individuales de cada uno de los trabajadores y como estos se identifican como trabajadores docentes y no docentes.

PROCESO: Se crea una clase que posee los atributos y procesos de la lista, entre las cuales se encuentran las necesarias para agregar elementos, eliminarlos, modificarlos o mostrar la lista. Además de poder usar la herencia de clases para determinar los atributos compartidos.

SALIDA: La lista de los trabajadores indicando sus elementos y funciones.

Devolver el nombre completo de una persona dado el DPI

ENTRADA: El numero de DPI

PROCESO: Se realiza una búsqueda por medio del numero de DPI, por medio de un metodo, de aquí se obtienen el nombre y los apellidos, tras esto esos se unen a

SALIDA: El nombre completo del alumno.

Mostrar un listado con los estudiantes de Doctorado, Pregrado o Postgrado y exportar la información a un archivo CSV

ENTRADA: El listado con todos los alumnos.

PROCESO: Se recorera el listado de alumnos, se comprobara a que grado pertenece cada alumno, y dependiendo del valor del grado, insertara cada uno en una de tres listas dependiendo de su grado, despues de esto las tablas se imprimiran en un archivo CVS

SALIDA: Archivo CVS con el listado de los alumnos.

Mostrar un listado con los docentes y exportar la información a un archivo CSV ordenado por el primer apellido

ENTRADA: El listado con los trabajadores.

PROCESO: Se recorre la lista con los trabajadores, se determina si el trabajador es un docente, tras esto se coloca en una lista exclusiva para los docentes, se coloca cada uno de los miembros y se ordenan por medio del apellido.

SALIDA: La lista ordenado por los apellidos de los docentes de la institución.

Mostrar un listado con los datos del personal trabajador no docente y exportar la información a un archivo CSV ordenado por el primer apellido

ENTRADA: El listado con los trabajadores.

PROCESO: Se recorre la lista con los trabajadores, se determina si el trabajador no es un docente, tras esto se coloca en una lista exclusiva para los no docentes, se coloca cada uno de los miembros y se ordenan por medio del apellido.

SALIDA: La lista ordenado por los apellidos de los trabajadores no docentes de la institución.

Calcular el salario promedio de todos los trabajadores (docentes, no docentes o ambos)

ENTRADA: Los salarios presente en la lista de trabajadores.

PROCESO: El usuario determinara el grupo de trabajadores de los cual quiere el salario promedio, tras esto se recorre la lista con los trabajadores si el trabajador se encuentra entre el grupo empezara a sumar los salarios y utilizara un contador para determinar la cantidad de salarios sumados, con estos datos se encontrara el promedio de los salarios.

SALIDA: Salario de los empleados.

Determinar la cantidad de estudiantes en Maestría o por facultad.

ENTRADA: La facultad o maestria que se desea buscar, la lista de estudiantes.

PROCESO: Se recorre la lista y se verifica la facultad o maestria a la cual pertenece el estudiante, si esta es igual a la que se indico se aumentara en uno el valor de un contador.

SALIDA: Contador con el numero de estudiantes.

Dado un curso determinar la cantidad de docentes que lo imparten y un listado con los estudiantes que lo reciben ordenados alfabéticamente por el nombre o por nota

ENTRADA: Listado de alumnos y docentes, el curso seleccionado

PROCESO: Una vez seleccionado el curso se recorre la lista de docentes, si entre los cursos se encuentran estos el seleccionado, se usa un contador que aumenta su valor. Al mismo tiempo se recorre la lista de estudiantes y se agregan a una nueva lista si entre los cursos se encuentra el seleccionado, se agregan a una lista, esta sera ordenada dependiendo del nombre y las notas.

SALIDA: Numero de docentes y lista de los alumnos.

Dado un estudiante de pregrado o postgrado conocer su promedio

ENTRADA: El nombre del estudiante, lista de estudiantes.

PROCESO: Se buscara al estudiante en la lista, tras esto se obtiene la lista con los cursos del alumno, se obtiene la nota de cada curso, se suman y se divide por el numero de cursos de esta forma se obtiene el promedio.

SALIDA: Promedio del estudiante seleccionado.

Dado un estudiante de pregrado conocer su nota más alta y el curso al que corresponde dicha nota

ENTRADA: El nombre del estudiante, la lista de estudiantes.

PROCESO: Se realizara una busqueda del estudiantes, una vez encontrado se realizara una busqueda dentro de la lista de curso para encontrar la nota más alta y el curso a la que esta pertenece.

SALIDA: Nota mas alta y curso.

Dada una facultad, conocer el alumno de pregrado con mejor promedio

ENTRADA: Facultad, lista de alumnos.

PROCESO: Se buscara a los alumnos cuya facultad equivalga a la que se esta buscando, se calculara el promedio de cada alumno y se ira comparando para encontrar al de mayor valor.

SALIDA: Alumno con mayor promedio.

III. RESTRICCIONES

Como una de las restricciones se tiene que el usuario al momento de cargar su archivo de CSV de docentes y alumnos tiene que verificar que cada linea de estas al finalizar no contenga una coma, punto u otro simbolo raro que afecta la lectura del programa de lo contrario se caera la aplicación.

Otra restricción que se nos presento es que el DPI de la persona que desee ingresar o modificar tiene que tener una separacion entre cada digito como lo muestra en la figura de debajo de lo contrario le aparecera una ventana emergente como la mostrada a continuación.



Esta restricción aplica también para cuando el usuario quiere realizar una búsqueda con el DPI de la persona, si el usuario ingresa un DPI incorrecto o con el formato incorrecto la aplicación le mostrara un label con las palabras “No se encontro registros”.

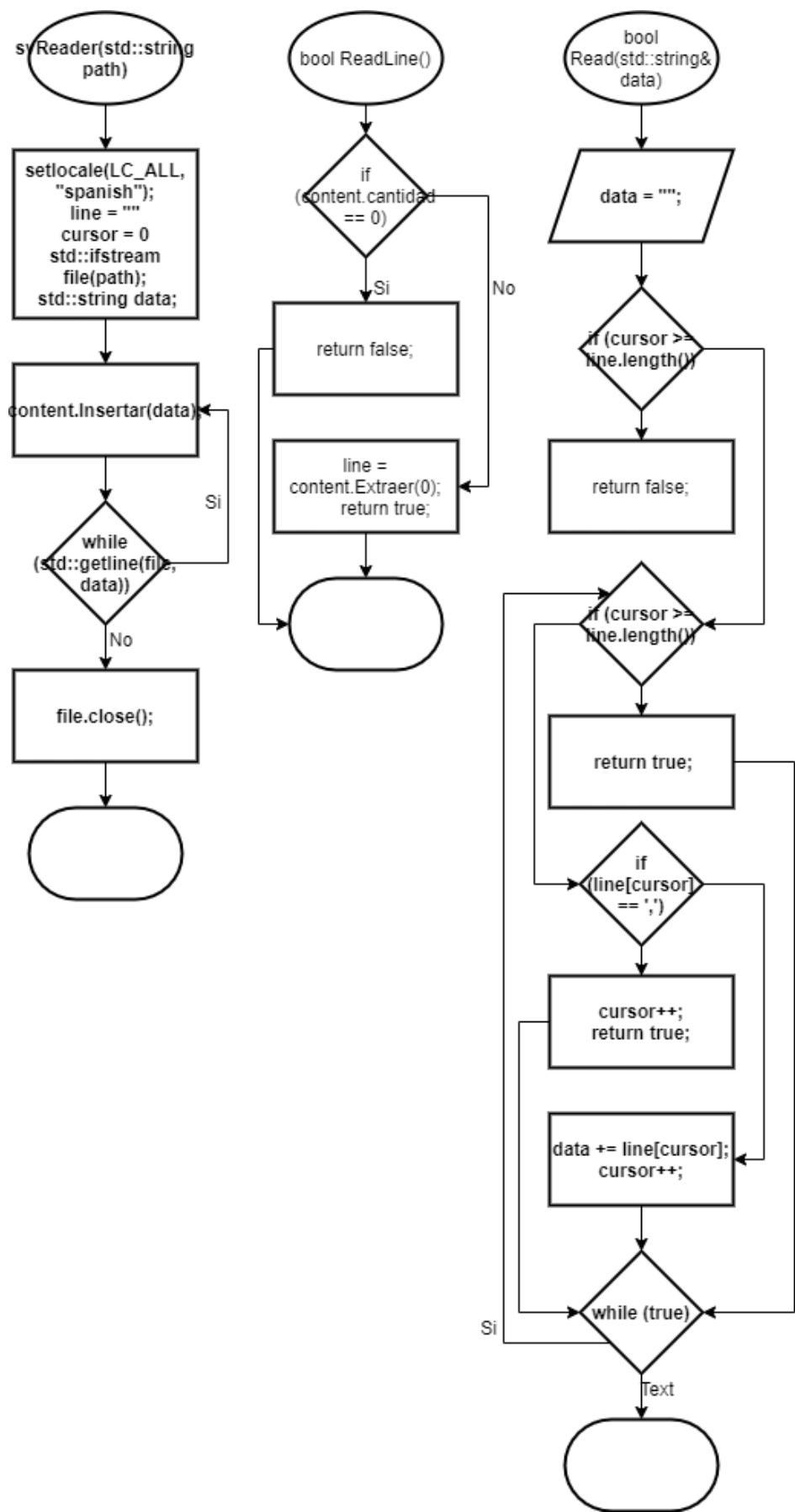
Como otra restricción se tiene que al momento de cargar un archivo CSV el usuario coloca fechas en el apartado del año del estudiante el programa se caera ya que lo recomendado es que solo ingrese el año.

V. DIAGRAMA DE FLUJO

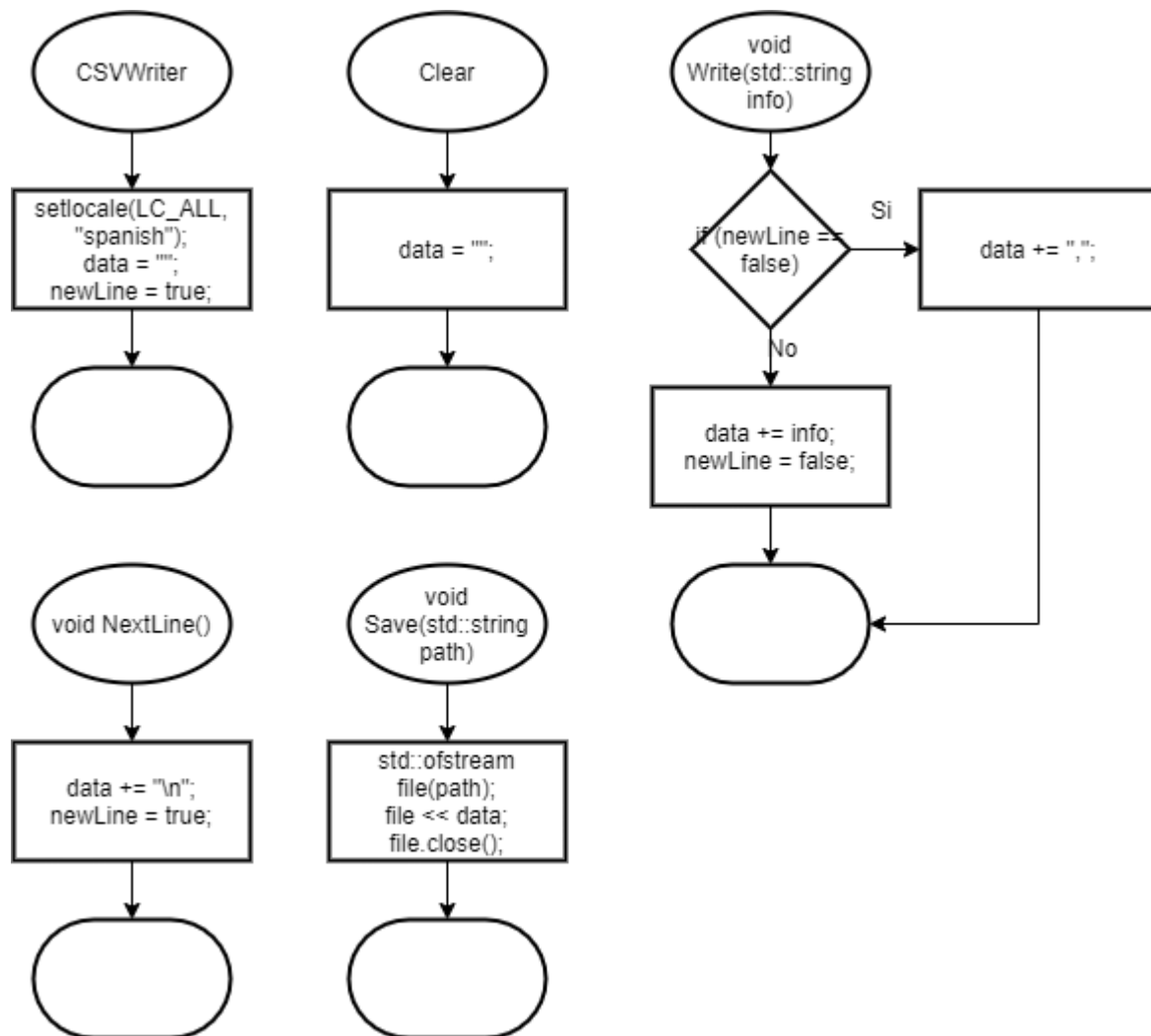
```

classDiagram
    class CsvReader {
        private:
            int cursor;
            Lista<std::string> content;
            std::string line;
        public:
            CsvReader(std::string path)
            bool ReadLine()
            bool Read(std::string& data)
    }
    class CsvWriter {
        std::string data;
        bool newLine;
        public:
            CsvWriter()
            void Clear()
            void NextLine()
            void Save(std::string path)
    }
    class Persona {
        public:
            std::string Nombres;
            std::string Apellidos;
            std::string DPI;
    }
    class Alumno {
        public:
            int Carnet;
            int Ingreso;
            std::string Facultad;
            std::string Modalida;
            Lista<Curso*>* Cursos = new
    }
    class Empleado {
        public:
            int Codigo;
            int Inicio;
            float Salario;
    }
    class Docente {
        public:
            Lista<Curso*>* Cursos =
            new Lista<Curso*>;
    }
    class Trabajador {
        public:
            std::string Cargo;
    }
    class Cursos {
        public:
            std::string Nombre;
            float Nota;
    }
    class Nodo {
        Nodo* siguiente;
        Nodo* anterior;
        T valor;
        Nodo()
    }
    class Lista {
        public:
            Nodo<T*> primero;
            Nodo<T*> ultimo;
            int cantidad;
        Lista()
        void Insertar(T valor)
        T Extraer(int posicion)
        Nodo<T*> ObtenerNodo(int posicion)
        void Intercambiar(int pos1, int pos2)
        T Modificar(T valor, int posicion)
        T Obtener(int posicion)
        T operator[](int posicion)
    }
    class frmAlumno {
        Alumno* alumnoModificar = nullptr;
        frmAlumnos(void)
        void LeerCSV()
        void AgregarColumna(String^ nombre)
        void LlenarDatosAlumnos()
        ~frmAlumnos()
        Void btnCrearRegistrar_Click
        btnCrearAgregarCurso_Click
        btnBorrar_Click
        btnCrearEliminarCurso_Click
        cbxModificarAlumnos_SelectedIndex()
        btnModificarAgregarCurso_Click
        btnModificarEliminarCurso_Click
        btnModificar_Click
        tabPage5_Enter
    }
    class frmDocentes {
        Docente* docenteModificar = nullptr;
        frmDocente(void)
        ~frmDocente()
        void LeerCSV()
        void AgregarColumna(String^ nombre)
        void LlenarDatosDocentes()
        btnCrearRegistrar_Click
        btnCrearAgregarCurso_Click
        btnCrearEliminarCurso_Click
        btnModificar_Click
        btnModificarAgregarCurso_Click
        btnModificarEliminarCurso_Click
        cbxModificarDocentes_SelectedIndexChanged
        btnBorrar_Click
    }
    class frmEmpleados {
        Trabajador* trabajadorModificar = nullptr;
        frmEmpleado(void)
        void LeerCSV()
        void AgregarColumna(String^ nombre)
        void LlenarDatosTrabajadores()
        ~frmEmpleado()
        Void btnCrearRegistrar_Click
        Void btnModificar_Click
        Void cbxModificarTrabajadores_SelectedIndexChanged
        Void btnBorrar_Click
    }
    class Classname {
        frmMenu(void)
        ~frmMenu()
        Void btnAlumnos_Click
        Void btnDocentes_Click
        Void btnEmpleados_Click
        Void btnPersonas_Click
        Void btnReportes_Click
    }
    CsvReader --> Empleado
    Empleado --> Docente
    Empleado --> Trabajador
    Persona --> Alumno
    Empleado --> Cursos
    Docente --> Cursos
    Trabajador --> Cursos
    
```

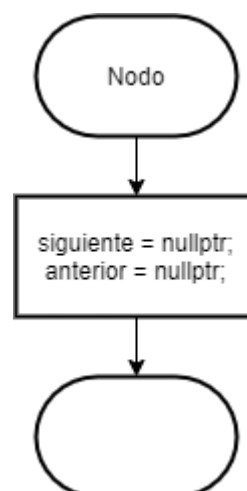
CSVReader



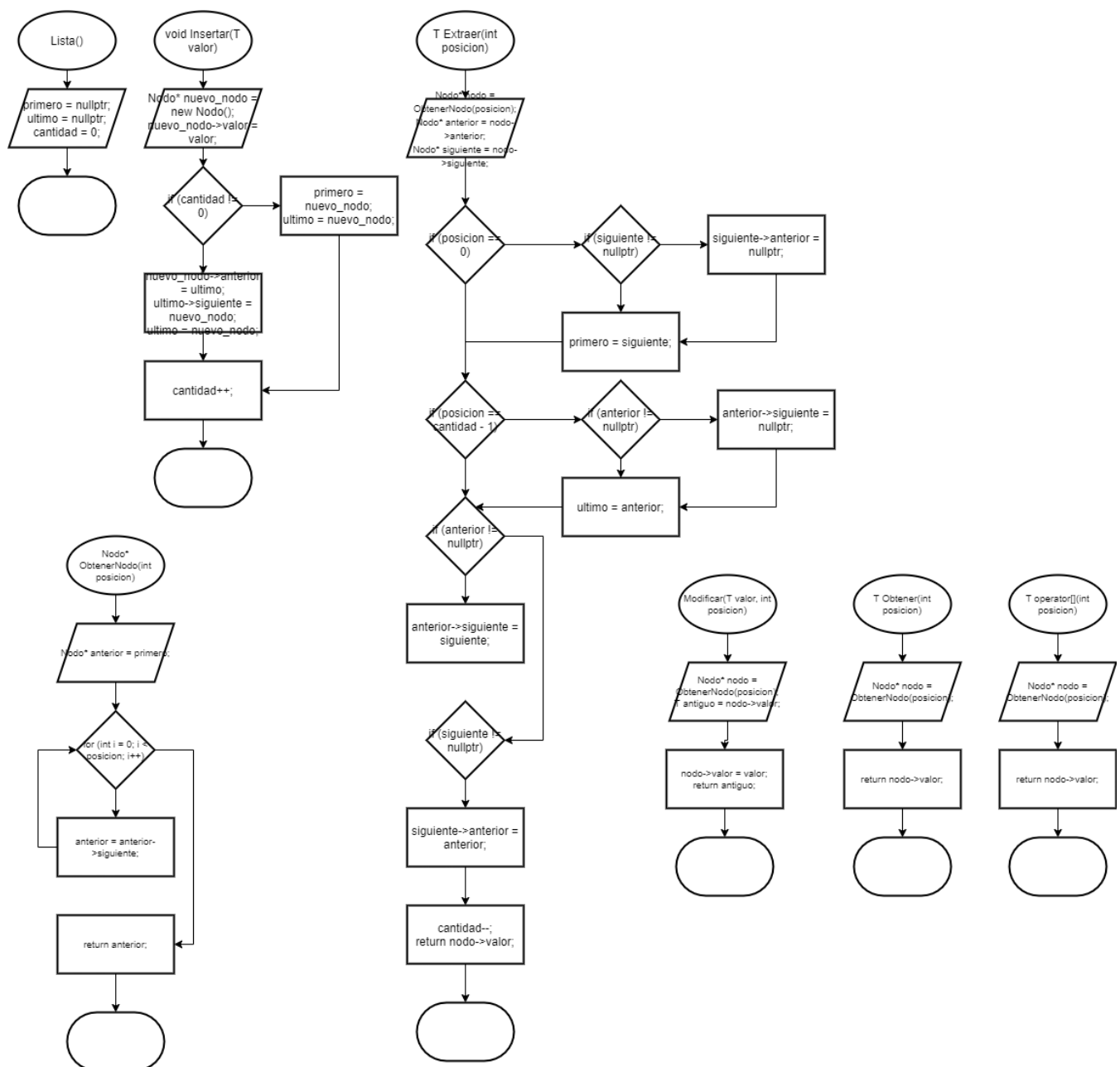
CSVWriter



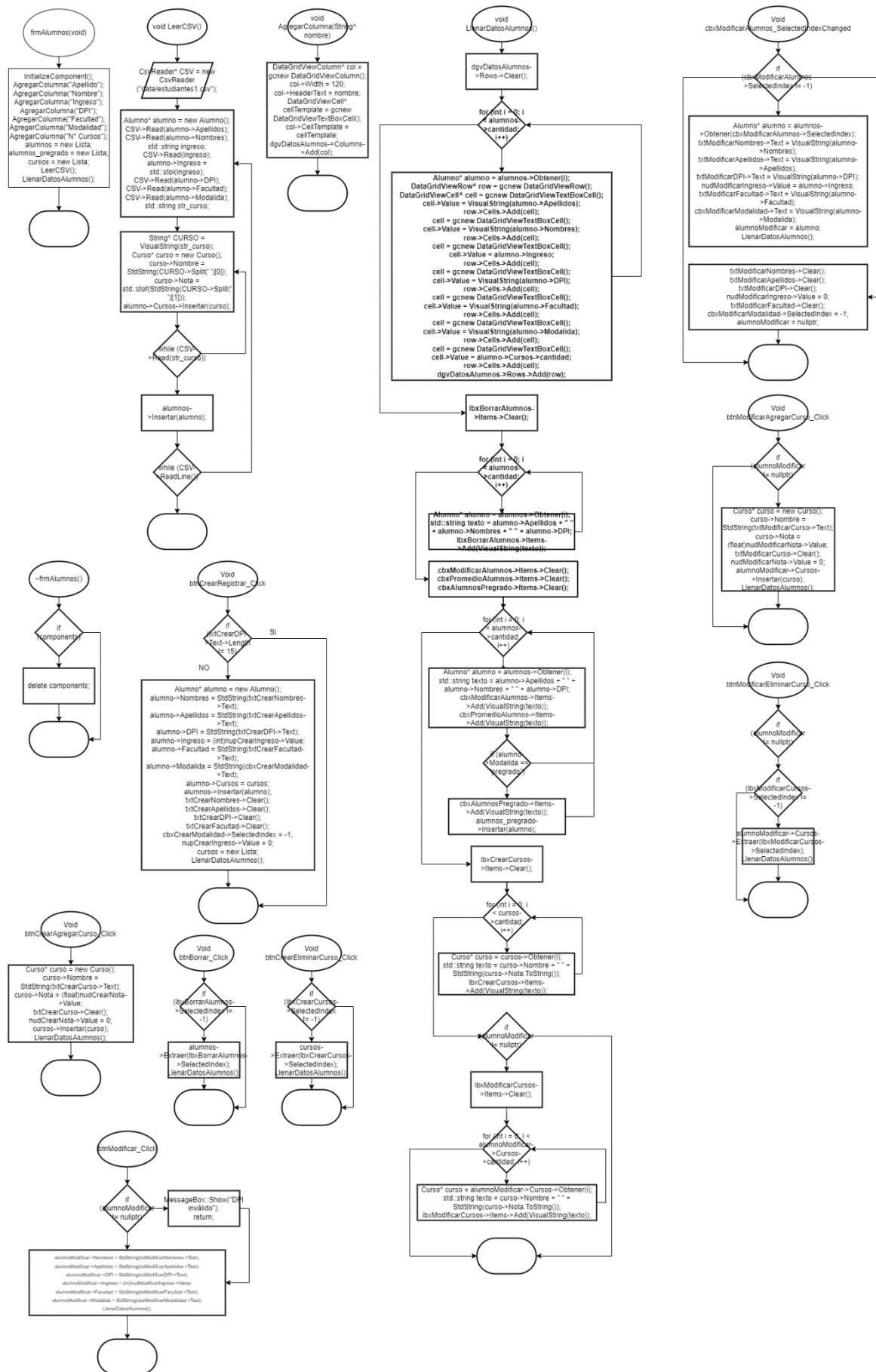
Nodo



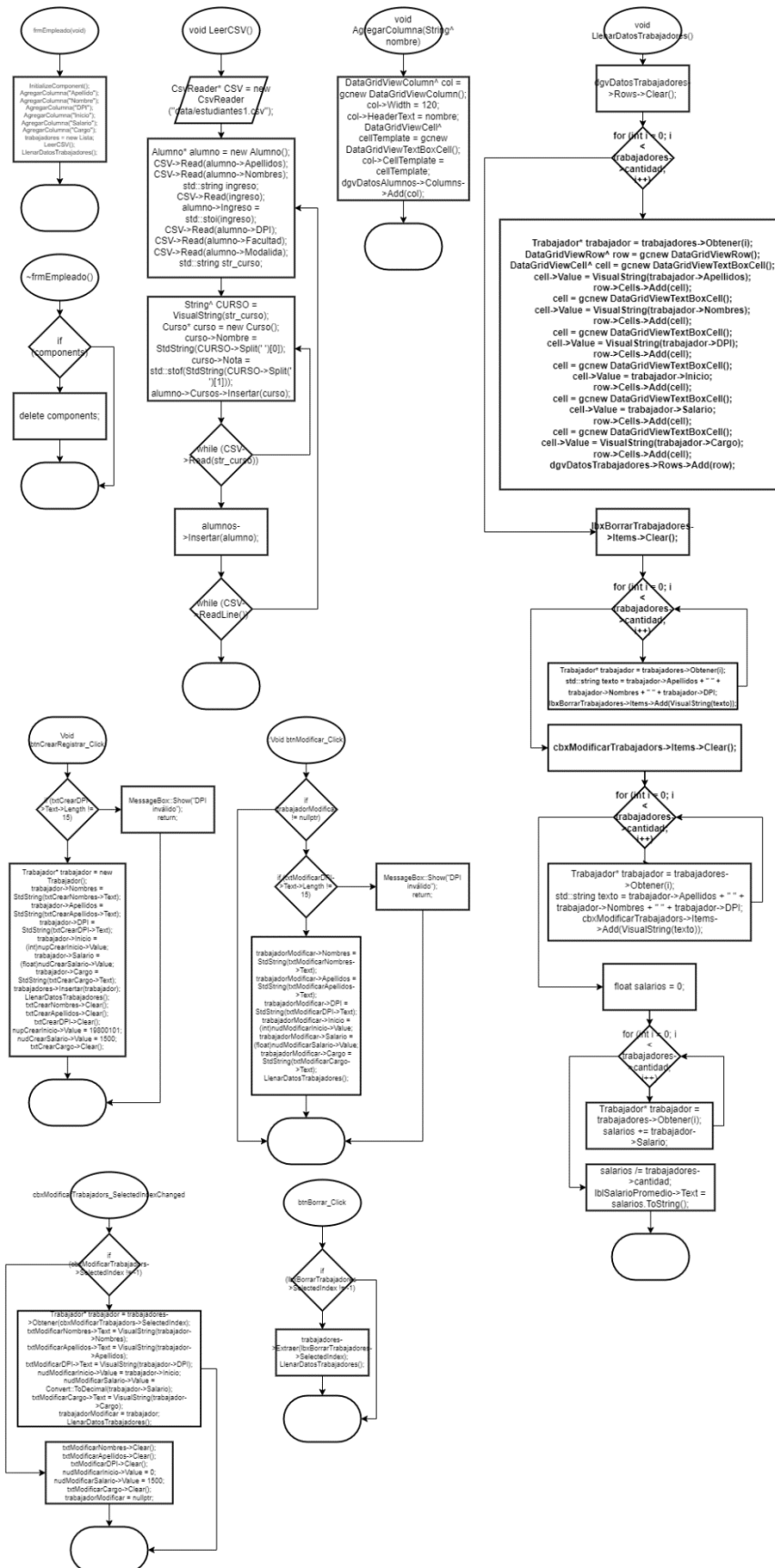
Lista



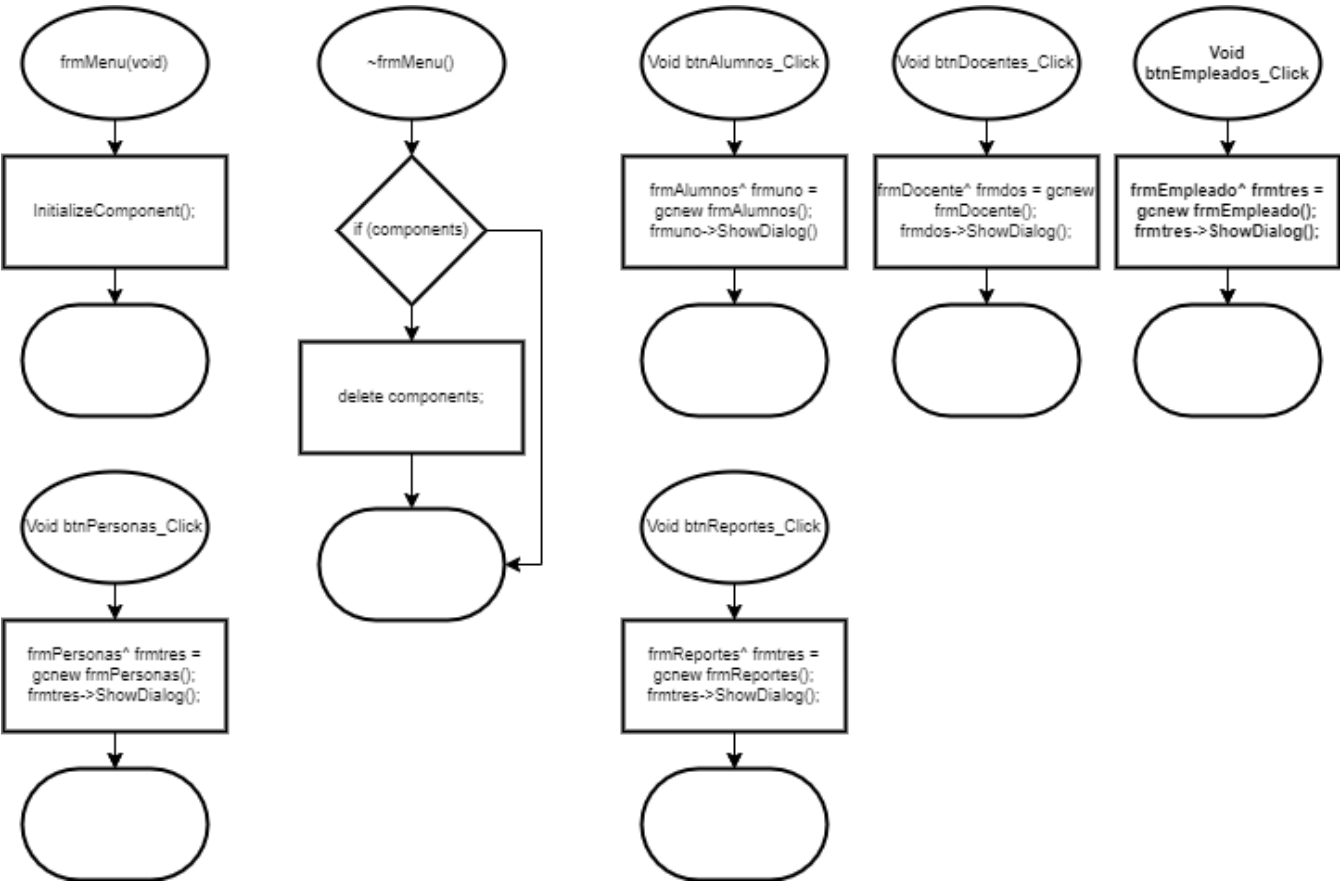
frmAlumnos y frmDocentes



frmEmpleados



frmMenu



VI. CONCLUSIONES

Conclusión No.1 La importancia de la informática e Ingeniería en Sistemas en la implementación de soluciones

A lo largo de la historia se han presentado problemas conforme la información dentro de la sociedad fue aumentando de una manera exponencial, pero la información no fue lo único que aumentó de manera exponencial con el pasar del tiempo, sino también la tecnología y nuestra comprensión de los sistemas de información también; En este caso se implementó una solución a los sistemas arcaicos de librerías físicas que albergan registros en libros físicos. La solución soluciona problemas frecuentes en sistemas de información físicos tales como la pérdida de datos por la degradación del papel y la posible pérdida de información por accidentes como inundaciones o incendios, debido a que la información pasa de ser física a ser algo digital, más fácil de almacenar y manejar, y ni se diga lo considerablemente segura que es su almacenación. Se implementó una aplicación la cual por medio de archivos csv se guardó y cargo información de estudiantes, docentes, empleado (Personas en general), la cual fue utilizada para implementar herramientas tales como el agregar algún ente a cualquiera de los registros, obtener la información de cualquier persona dentro de los datos en cuestión de milisegundo, acceder a su modificación, eliminación, etc. Si pensamos esto más a profundidad nos podemos dar cuenta del potencial de las tecnologías de la información y como con la ayuda de la ingeniería en informática y sistemas pueden generarse sistemas complejos de información los cuales pueden ser manipulados de una manera sencilla por un usuario sin conocimientos avanzados sobre computación, informático o ingeniería.

Conclusión No.2 Trabajo en equipo, y la importancia de la comunicación

Mientras los problemas de la sociedad a nivel, industrial, laboral, social, etc. Van aumentando de complejidad, las capacidades necesarias para solucionarlos también van aumentando, lo que incentiva a personas a trabajar en equipos para desarrollar una solución especializada en cada área de la implementación; Una parte importante dentro de los equipos desarrollados para implementar una solución es la comunicación, dicha comunicación es la que permite al equipo avanzar en un proyecto de manera sincronizada y efectiva, de no ser eficiente la comunicación, el desarrollo de dicho proyecto se vuelve menos eficiente lo cual implica retraso en la incorporación de las partes que conforman la solución e inclusive conflictos dentro de la propia organización de equipo debido a la poca sincronía y entendimiento dentro del mismo; Pero de lograr llegar a una comunicación efectiva, esto representa una mejora significativa en los resultados positivos de la implementación de dicha solución.

Conclusión No.3 Correcta utilización de las herramientas informáticas

Una parte importante al momento de trabajar en equipo es el establecer el conjunto de herramientas básicas que se utilizarán en el desarrollo de la solución; Desde el IDE que se utilizará para desarrollar dicha solución, hasta el entorno del manejo del versionamiento con el que se estará trabajando; Debido a que cada una de las herramientas básicas establecidas permiten que la soluciones de desarrolle en un mismo entorno, lo que permite menores errores de compilación y una sincronización más efectiva que permite un desarrollo más eficiente.

Conclusión No.4 Conclusiones técnicas

- La herencia implementada de una manera correcta nos permite utilizar funciones y/o datos de una clase principal en otras clases derivadas; Dicha implementación nos permitó implementar soluciones basadas en polimorfismo que hace de la reestructuración de clases más específicas (Clases creadas para solucionar problemas específicos de una rama en general) pueda ser un proceso más fluido y simple.
- La creación de clases nos permite crear estructuras de datos o entidades adaptadas a problemas específicos, lo cual nos permite una mayor versatilidad en la implementación de soluciones con problemas puntuales y específicos ya que las clases pueden contener desde tipos de datos, hasta estructuras de datos complejas.
- No todos los tipos de estructuras son útiles y efectivos para todo tipo de problemas, una parte importante del desarrollo de una solución informática es el diseño de la misma, en dicho proceso es donde se decide que tipo de estructura es la indicada para el manejo de los datos a utilizar y los problemas a resolver; De no ser establecida de una manera clara y con justificación puede llevar a la generación de problemas innecesarios que perjudicarán la implementación de la solución desarrollada.

VII. RECOMENDACIONES

Diseñar antes de programar: El diseño ahorra muchos problemas de lógica debido a que en el proceso de diseño podemos darnos cuenta de ciertos procesos o errores que podemos tener incorrectos antes de empezar con el código, lo cual hace del proceso de codificación una tarea agobiante y repetitiva debido a que no tenemos claro que debe hacer cada una de las partes del código; Por lo que establecer que estructuras se utilizarán para el desarrollo de la solución es importante debido a que cada una satisface necesidades distintas.

La implementación de interfaces que permitan al usuario utilizar la aplicación con mayor facilidad hace que la utilización de la misma sea más amigable y simple de usar, lo cual aumenta el índice de efectividad de la misma.

Aunque la herencia es un proceso útil que nos permite ahorrarnos líneas de código y problema de compilación, no siempre es la mejor opción, en algunas ocasiones la implementación de la herencia puede ocasionarnos más problemas que soluciones, por eso es importante entender el concepto de herencia para saber en que momento es más recomendable su implementación.

VIII. REFERENCIAS

IOSTREAM: Declara objetos que controlan la lectura y la escritura en los flujos estándar. Esta inclusión es a menudo el único encabezado que necesita para hacer entrada y salida de un programa C++.

<https://docs.microsoft.com/en-us/cpp/standard-library/iostream?view=msvc-170>

STRING: Define muchas plantillas de strings básicos.

<https://docs.microsoft.com/en-us/cpp/standard-library/string?view=msvc-160>

LOCALE: Define plantilla y funciones que los programas en C++ pueden usar para encapsular y manipular diferentes convenciones culturales con respecto a la representación y el formato de datos numéricos, monetarios y calendáricos, incluido el soporte de internacionalización para la clasificación de caracteres y la intercalación de cadenas.

<https://docs.microsoft.com/en-us/cpp/standard-library/locale?view=msvc-160>

FSTREAM: Nos sirvió para la lectura y escritura del archivo csv.

<https://docs.microsoft.com/en-us/cpp/standard-library/fstream?view=msvc-160>

MSCLR\MARSHAL_CPPSTD.H: MSCLR es una librería estándar, pero la utilización de su header MARSHAL_CPPSTD.H fue para la utilización de funciones marshaling.

<https://docs.microsoft.com/en-us/cpp/dotnet/overview-of-marshaling-in-cpp?view=msvc-170>

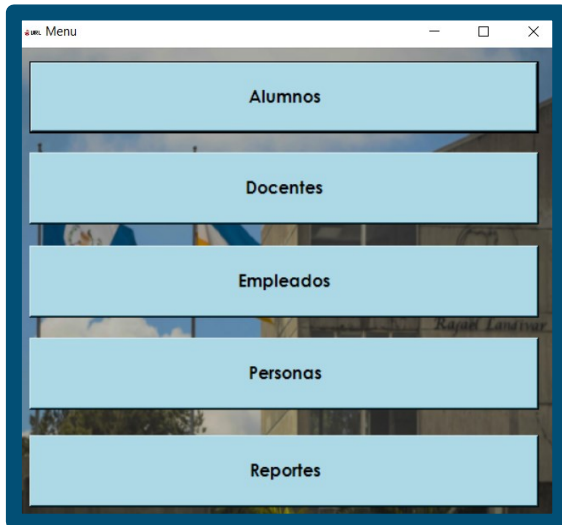
Conversión de C++ / CLI de System::String^ a std::string: Esta conversión nos permitió la conversión de un System::String^ a un std::string debido a que hubieron muchas entradas de cadenas por medio de textboxes lo cuales representan a un System::String^ mientras las listas utilizan un std::string

<https://stackoverflow.com/questions/946813/c-cli-converting-fromsystemstring-to-stdstring>

IX. ANEXOS

X. MANUAL DE USUARIO

MENU:



En el menú podremos acceder a cualquiera de los apartados de la aplicación

ALUMNOS:

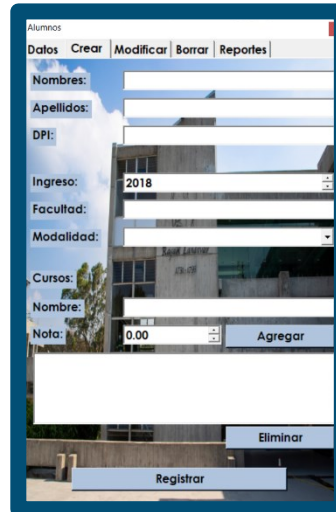
DATOS



En el apartado de *Datos* podremos tener acceso a la información de los alumnos y encontraremos el botón “Exportar” que exporta al archivo los cambios y el

botón “Abrir” que nos permitirá seleccionar un archivo para cargar los datos.

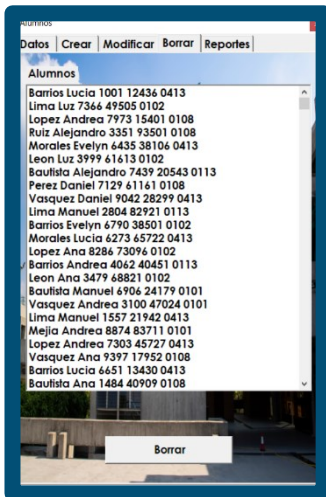
CREAR



En el apartado de *Crear* podremos crear un nuevo registro de alumno llenando cada uno de los campos.

Para agregar los cursos del alumno se deberá escribir el nombre del curso, la respectiva nota del curso y presionar el botón “Agregar” y así sucesivamente con cada uno de los cursos, los cuales podremos ir visualizando en una lista, en la cual podremos seleccionar alguno para borrar de la misma manera que en el apartado de borrar.

BORRAR

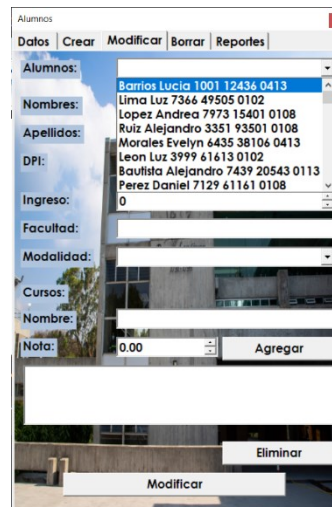


En el apartado de *Borrar*, seleccionando el alumno a borrar y presionando el botón “Borrar” podremos eliminar un alumno del registro.

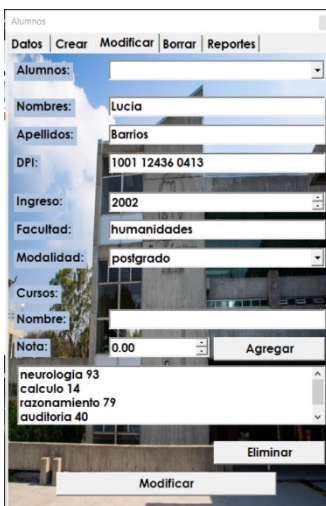
MODIFICAR



En el apartado de *Modificar* podremos modificar los datos de cualquiera de los alumnos.



Seleccionamos al alumno a modificar.



Los datos del mismo se cargarán automáticamente y podremos modificar cualquiera de los campos; Para aplicar los cambios

debemos presionar el botón “Modificar”.

REPORTES

En el apartado de *Reportes* tenemos la cantidad de alumnos en Postgrado; Alumnos por facultad seguido del mejor promedio;

También podemos seleccionar un alumno y conocer su promedio o su mejor nota.

DOCENTES:

DATOS

Codigo	Apellido	Nombre	D.
2197985	Rodriguez	Alvaro	43
1788717	Sanchez	Jacobo	43
1653898	Cerezo	Jacobo	43
3279588	Castillo	Alvaro	43
1726691	Valdez	Patricia	43
2350913	Sanabria	Alfonso	43
3455717	Rodriguez	Laura	43
1231796	Arzu	Alvaro	43
2813503	Portillo	Laura	43
1259318	Castillo	Alvaro	43
2646690	Valdez	Gabriela	43
3295118	Argueta	Laura	43
1517415	Colom	Guadalupe	43
1251218	Cerezo	Alvaro	43
1955111	Sanabria	Gabriela	43
1269512	Castillo	Patricia	43
3181705	Argueta	Laura	43
3349205	Colom	Patricia	43
3129217	Castillo	Nahomi	43
3753084	Argueta	Jimmy	43
3171087	Valdez	Alfonso	43

En el apartado de *Datos* podremos tener acceso a la información de los docentes y encontraremos el botón “Exportar” que exporta al archivo los cambios y el

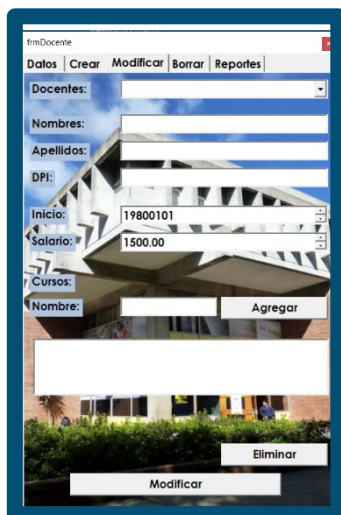
botón “Abrir” que nos permitirá seleccionar un archivo para cargar los datos.

CREAR

En el apartado de *Crear* podremos crear un nuevo registro de docente llenando cada uno de los campos. El formato de la fecha es el siguiente: AñoMesDía

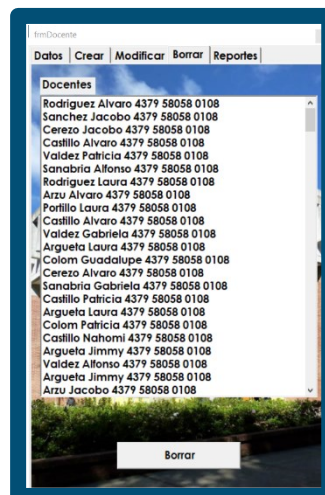
Para agregar los cursos es de la misma manera que en el apartado de Alumnos, solo que en este caso no hay nota.

MODIFICAR



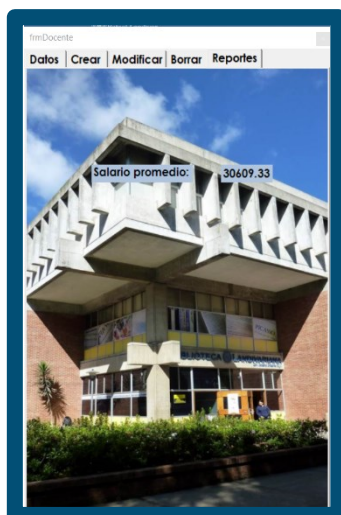
El apartado de modificar de docentes funciona de manera similar que en el de Alumnos, la única diferencia son los campos y sus valores.

BORRAR



En el apartado de *Borrar*, seleccionando el docente a borrar y presionando el botón “Borrar” podremos eliminar un docente del registro.

REPORTES



En el apartado de *Reportes* podemos encontrar el salario promedio.

EMPLEADOS:

DATOS



En el apartado de *Datos* podremos tener acceso a la información de los docentes y encontraremos el botón “Exportar” que exporta al archivo los cambios y el

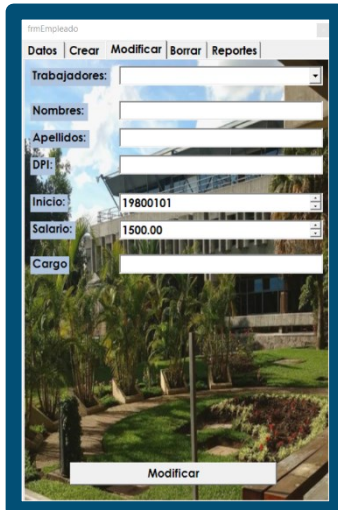
CREAR



En el apartado de *Crear* podremos crear un nuevo registro de empleado llenando cada uno de los campos. El formato de la fecha es el siguiente:
AñoMesDía, la

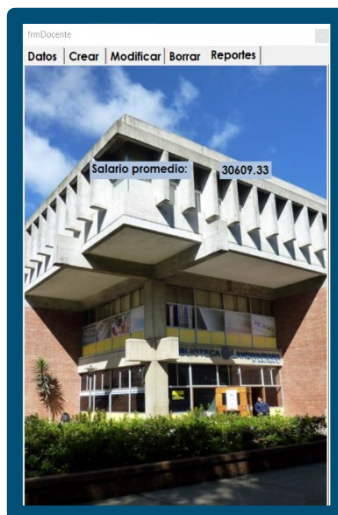
botón “Abrir” que nos permite seleccionar un archivo de datos.

MODIFICAR



El apartado de modificar de empleados funciona de manera similar que los apartados anteriores, la única diferencia son los campos y sus valores.

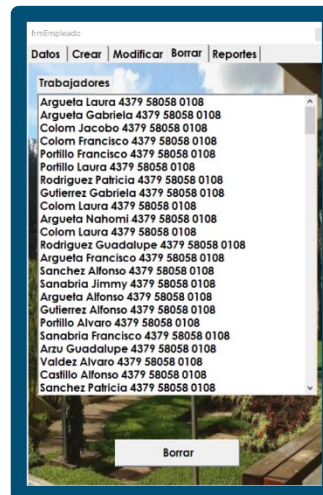
REPORTES



En el apartado de *Reportes* podemos encontrar el salario promedio.

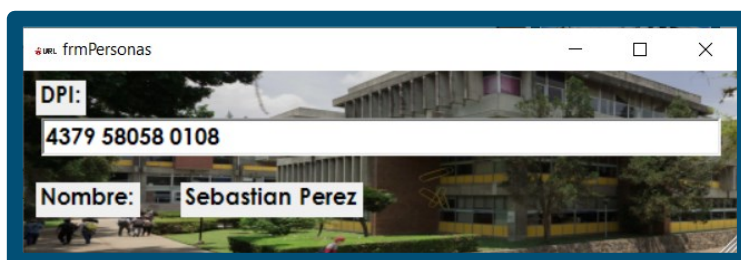
manera de agregar es la misma que en los apartados anteriores.

BORRAR



En el apartado de *Borrar*, seleccionando el empleado a borrar y presionando el botón “Borrar” podremos eliminar un empleado del registro.

PERSONAS:



aparecera en pantalla.

En el apartado de personas podemos buscar a una persona por medio del número de DPI; Escribimos en DPI en el cuadro de texto y automáticamente el nombre

REPORTES:

The screenshot shows a report window titled 'frmReportes'. At the top, there is a search bar labeled 'Curso:' with the value 'calculo'. Below this, there are two main sections: 'Alumnos:' and 'Profesores:'. The 'Alumnos:' section shows a count of 246 and a list of 16 students with their IDs and names. The 'Profesores:' section shows a count of 45 and a list of 16 teachers with their names. At the bottom, there are two buttons labeled 'Nombre' and 'Nota'.

Alumnos:	Profesores:
14 Lucia Barrios	Laura Portillo
14 Luz Leon	Jimmy Argueta
63 Manuel Lima	Alfonso Valdez
47 Andrea Lopez	Jacobo Arzu
69 Lucia Barrios	Alvaro Castillo
51 Luz Perez	Jacobo Valdez
98 Luz Lima	Alvaro Portillo
98 Evelyn Perez	Francisco Cerezo
6 Sebastian Morales	Laura Valdez
8 Luz Barillas	Guadalupe Rodriguez
75 Kevin Ruiz	Guadalupe Rodriguez
38 Luz Lima	Laura Portillo
37 Lucia Lopez	Gabriela Argueta
100 Alejandro Ruiz	Jimmy Rodriguez
77 Alejandro Barrios	Jimmy Colom
74 Manuel Vasquez	Patricia Castillo
96 Luz Lima	Nahomi Portillo
1 Kevin Morales	Francisco Gutierrez
64 Lucia Mejia	Alfonso Valdez
	Laura Portillo

En el apartado de reportes con el nombre del curso podemos visualizar la cantidad de alumnos que lo reciben, seguido de una lista de los nombres y el punteo, lista la cual podemos ordenar alfabéticamente o por nota; Y también la cantidad de docentes que imparten el curso seguido de una lista con los nombres