UTN – UA Mar del Plata – TSP ***TSP - Laboratorio 1*** ***Trabajo Práctico Final*** *Junio 2017*	***Integrantes del grupo***	***Nota***

Introducción

Con el propósito principal de integrar contenidos de otras asignaturas y además integrar todo lo aprendido en la materia laboratorio 1 hemos planteado la siguiente problemática:

- Codificar un sistema de Logueo que gestione la estructura cliente, guarda, recupera y encripta la contraseña del mismo en un archivo binario.

Encriptar es la forma más efectiva de alcanzar una auténtica seguridad en los datos electrónicos. Para leer un archivo encriptado, es necesario tener acceso a la contraseña que está guardada en el mismo para poder desencriptar la información. Para lograr este objetivo deberán pensar en desarrollar un algoritmo de encriptación, que mediante el uso de matrices y cálculos matemáticos, que logre ocultar a los ojos curiosos la información de nuestro cliente.
Aplicando los conocimientos que ya adquirimos como alumnos de la cátedra “Matemática”, desarrollaremos diversos algoritmos que trabajen con cálculos matriciales y vectores para lograr encriptar información.

Fundamentación

El valor pedagógico de la propuesta se apoya en el aprendizaje colaborativo (se formarán grupos de 2 o 3 alumnos) y la integración de contenidos de otras asignaturas a partir del desarrollo de un proyecto de software. Para que este tipo de proyectos sea más exitoso, deben llevarse a cabo desde un enfoque que facilite alcanzar los Objetivos de Aprendizaje propuestos.
Una de las ideas centrales es desarrollar competencias profesionales y preparar al futuro programador para el mundo laboral y el trabajo en equipo.
En un ambiente de aprendizaje colaborativo, los estudiantes:

- Construyen conocimiento y en lugar de recibirlos en forma pasiva;
- Se involucran y comprometen directamente con el descubrimiento de nuevo conocimiento;
- Se exponen a puntos de vista alternativos e ideas contrapuestas, de forma tal que pueden sacar sus propias conclusiones y así transformar conocimientos y experiencias previas y de esta manera comprender con mayor profundidad;
- Transfieren conocimientos y habilidades nuevos a nuevas situaciones o circunstancias;
- Se responsabilizan y apropian tanto de su aprendizaje continuo de contenidos curriculares como del desarrollo propio de competencias;
- Los estudiantes colaboran para el aprendizaje del grupo y el grupo colabora en el aprendizaje individual de estos.

Objetivos

De aprendizaje:

- Incorporar Arreglos, Matrices, Modularización, Estructuras de Datos complejas y Archivos Binarios.
- Incorporar cálculos matriciales aplicados a la programación.
- Trabajar en forma colaborativa.

Metodológicos:

- Ser capaces de trabajar en un proyecto complejo, aplicando técnicas de desarrollo de software.
- Lograr integrar contenidos de otras asignaturas.
- El grupo deberá ir mostrando el avance sobre el trabajo en clase.

Modo de Evaluación del Trabajo Práctico

- Se establece el desarrollo de un trabajo práctico final, brindando una fecha límite de entrega del mismo 23/06
- La aprobación del trabajo práctico estará sujeta a los puntajes considerados en la tabla debajo.
- Es obligatorio la presentación de este trabajo para aprobar la materia.

Apartado	***Puntaje***	***Obtenido***
Funciones para ABMCL de Clientes y Usuarios ● Alta, Baja, Modificación, Consulta y Listados de Clientes y Usuarios. ● Validación en el ingreso de los Datos. ○ Validación de los tipos de datos. ○ Validación de la longitud de la contraseña (máximo 10 char). ● Buscar Usuario.	25	
Funciones para manejo de archivos binarios ● Persistencia de datos en Archivos. ● Listar clientes. ● Validar si el Usuario ya existe.	25	
Funciones para cálculos matriciales, encriptación y desencriptación de password. **<u>Registro de Usuario:</u>** ● Codificación del Pass ○ Cargar matriz con el pass ○ Multiplicación de la matriz de codificación/testigo x Matriz mensaje. ○ Guardar en archivo **<u>Login</u>** ● Decodificación del Pass ○ Leer Archivo ○ Cargar la Matriz codificada (Password) ○ Calcular inversa de Matriz de codificación/testigo ○ Multiplicar la inversa con el Password = Password decodificado.	25	
Función main () y funciones de manejo de vistas del sistema Orden y prolijidad del código entregado. Explicación presencial del sistema ● Correcta modularización de las funciones. ● Correcto uso de parámetros. ● Prolijidad general del código. ● Reutilización de las funciones. ● Comentarios del código.	25	

Como condición de aprobación mínima la aplicación tiene que ser funcional y compilar. Además, tiene que cumplir con las siguientes funcionalidades:

- Organizar en Menús
- Registrar usuario y clientes.
- Persistir datos en el archivo; tanto de registro como de password encriptado por medio de cálculos matriciales.
- Login: desencriptar password para lograr logeo por medio de cálculos matriciales.

Tabla de puntuación:

Obtenido	10	20	30	40	50	60	70	75	80	90	100
Nota	1	2	2	3	4	5	6	7	8	9	10

PAUTAS GENERALES

Se nos pide desarrollar un sistema de gestión de clientes y usuarios que permita persistir la información en archivos binarios. Las claves de acceso de los usuarios deberán guardarse encriptadas en el archivo, para lo cual nos valdremos de los conocimientos adquiridos en la cátedra ***Matemática*** sobre matrices y sus aplicaciones en la criptografía.

Para la persistencia de datos en el archivo utilizaremos las siguientes estructuras de datos:

Estructura de Clientes
Id (auto incremental)
Nombre
Apellido
Domicilio
Localidad
	typedef struct {
	int id;
	char nombre[30];
	char domicilio[30];
	char localidad[30];
	int eliminado; // indica 1 o 0 si el cliente fue eliminado
	} stCliente;

Estructura de Usuarios
Id.cliente
Usuario
Contraseña **// se guardará en el archivo encriptada**
	typedef struct {
	int id_cliente;
	char usuario[20];
	int pass[2][5];
	int eliminado; // indica 1 o 0 si el cliente fue eliminado
	} stUsuario;

La carga de ambas estructuras se realiza en paralelo y deben persistir ambas en los archivos correspondientes. El campo id (stCliente) y id_cliente (stUsuario) están relacionados, para poder recuperar la información a partir de dicho campo.

También podrán utilizar la siguiente estructura de datos, a fin de organizar la información para mostrar el listado de usuarios (con la clave desencriptada).

Estructura de Usuarios para Listados Id.cliente Usuario Contraseña **// desencriptada**	typedef struct { int id_cliente; char usuario[20]; char pass[10]; } stUsuarioListado;

Como ya se ha dicho, toda la información administrada por el sistema se persistirá en 2 (dos) archivos binarios: “clientes.dat” y “usuarios.dat”.

El sistema tendrá que proporcionar el acceso a las diferentes funcionalidades:

Menú principal

1. Administración de Clientes
2. Ingreso con User y Pass

1- Sub-Menu de Administración de Clientes

- **Alta**: Una vez completado el formulario de alta se valida que no exista el usuario. Si no existe se encripta la contraseña (se convierte el pass en una matriz de 2xn se multiplica por la matriz testigo que la tengo como constante y el resultado queda en la matriz pass de la estructura) entonces se guardan los datos en los archivos correspondientes.
- **Baja**: En baja hay que tener la precaución de abrir los 2 archivos, modificar el campo eliminado en las dos estructuras y guardarlos.
- **Modificación**: En modificación se ingresa el nro de cliente. Si no existe, se muestra mensaje de error; si existe desencripta la contraseña y lo muestra; y se ve una forma de poder modificar los campos. Una forma sería mostrar los campos con un número de orden y pedir el ingreso del nro de camp a modificar.
- **Consulta**: Para la consulta la metodología sería similar a listados.
- **Listados**: Para los listados se abren los archivos y se cargan en los array correspondientes (por supuesto a medida que leo el archivo de usuarios voy desencriptando pass) y se trabajan en paralelo a la hora de ordenarlos según lo requerido y al mostrarlos.
    - **Sub-Menu de Listado de Clientes (ordenados)**
        1 Por Id (ordenado por selección)
        2 Por Apellido (ordenado por inserción)
        3 Por Domicilio (criterio a elejir)

2- Ingreso Con User y Pass: Esta pantalla pide que se ingrese Usuario y Contraseña, si el usuario existe comprueba que la contraseña sea correcta y muestra los datos del cliente. Si el usuario no existe << muestra mensaje >> y si el usuario existe pero la contraseña no es correcta << muestra mensaje >>

El desarrollo del sistema deberá ser ordenado, identificando con comentarios cada una de las funciones realizadas, explicando brevemente lo que realizan.
Se tendrá en cuenta,al momento de evaluar, la prolijidad del código y la organización de los módulos. Se recomienda agrupar los mismos por funcionalidad.

A continuación, explicaremos el mecanismo de encriptación y desencriptación de una contraseña, a partir del ejemplo planteado en la cátedra ***Matemática:***

Criptografía: el mundo de las telecomunicaciones y las nuevas tecnologías de la información se interesa cada vez más por la transmisión de mensajes encriptados que sean difíciles de desencriptar por otros, en caso de ser interceptados, pero que se decodifiquen con facilidad por quienes los reciben. Hay muchas formas interesantes de cifrar o encriptar mensajes ***(en nuestro caso, contraseñas)***, y en su mayor parte usan la teoría de números o el álgebra lineal. Describiremos aquí un método que es eficaz, en especial cuando se usa una matriz de gran tamaño.

Comenzaremos con una matriz M invertible, que sólo la conocen quienes encriptan y desencriptan las contraseñas.

Por ejemplo, $M = \begin{pmatrix} -3 & 4 \\ -1 & 2 \end{pmatrix}$

Supongamos que se desea encriptar la contraseña: ATTACK NOW. Reemplazamos cada letra por el número que le corresponde a su posición en el alfabeto (A B C D E F G H I J K L M N O P Q R S T U V W X Y Z) y representamos un espacio por 0, es decir:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

La contraseña anterior se ha convertido en la sucesión de números:

A	T	T	A	C	K	Espacio	N	O	W
1	20	20	1	3	11	0	14	15	23

Agrupamos estos números en una sucesión de vectores columna para formar una matriz de 2x5 (para poder multiplicar en este orden: M*N):

$$N = \begin{pmatrix} 1 & 20 & 3 & 0 & 15 \\ 20 & 1 & 11 & 14 & 23 \end{pmatrix}$$

$$M * N = \begin{pmatrix} -3 & 4 \\ -1 & 2 \end{pmatrix} * \begin{pmatrix} 1 & 20 & 3 & 0 & 15 \\ 20 & 1 & 11 & 14 & 23 \end{pmatrix} = \begin{pmatrix} 77 & -56 & 35 & 56 & 47 \\ 39 & -18 & 19 & 28 & 31 \end{pmatrix}$$

La contraseña cifrado que se persiste se lee por columnas: 77,39,-56,-18,35,19,56,29,47,31.

Para desencriptar la contraseña, quien la recibe debe calcular M^{-1} , dejemos a cargo del lector verificar que es:

$M^{-1} = \begin{pmatrix} -1 & 2 \\ -1/2 & 3/2 \end{pmatrix}$ y multiplicar por los números recibidos agrupados en una sucesión de vectores columna igual que antes, obtenemos así la contraseña original:

$$\begin{pmatrix} -1 & 2 \\ -1/2 & 3/2 \end{pmatrix} * \begin{pmatrix} 77 & -56 & 35 & 56 & 47 \\ 39 & -18 & 19 & 28 & 31 \end{pmatrix} = \begin{pmatrix} 1 & 20 & 3 & 0 & 15 \\ 20 & 1 & 11 & 14 & 23 \end{pmatrix}$$