

**INFORME**

**ABP DESARROLLO 1 Examenes**

**INTEGRANTE:**

* Miller vargas mola (87223).

**PRESENTADO A**: ING. John Carlos Arrieta Arrieta.

**INGENIRA EN SISTEMAS TERCER SEMESTRE**

**CARTAGENA DE INDIAS D, T Y C**

**03/ 13/ 2022**

**Términos**

**Lenguaje de programación:** En informática, se conoce como lenguaje de programación a un programa destinado a la construcción de otros programas informáticos. Su nombre se debe a que comprende un lenguaje formal que está diseñado para organizar algoritmos y procesos lógicos que serán luego llevados a cabo por un ordenador o sistema informático, permitiendo controlar así su comportamiento físico, lógico y su comunicación con el usuario humano**.**

**File:** Este nos permite buscar un archivo dentro del pc, según la ruta asignada.

**Arraylist:** La clase ArrayList en Java, es una clase que permite almacenar datos en memoria de forma similar a los Arrays, con la ventaja de que el número de elementos que almacena, lo hace de forma dinámica, es decir, que no es necesario declarar su tamaño como pasa con los Arrays.

**Programación orientada a objetos:** La programación orientada a objetos (POO) es un paradigma de programación que usa objetos para crear aplicaciones. Está basada en tres pilares fundamentales: herencia, polimorfismo, encapsulación. Su uso se popularizó a principios de la década de 1990.

**FileReader**: permite abrir el archivo que busca **File.**

**FileWriter(archivo,true):** gracias a el true nos permite escribir en el archivo .txt al final del contenido.

**Definición del problema**

Los profesores de la asignatura de Computación II de la Universidad de Chile deciden crear una base de datos que contenga la información de los resultados de los controles realizadas a los alumnos. Para realizar el diseño se sabe que:

● Los alumnos están definidos por su n° de matrícula, nombre y el grupo al que asisten a clase. Dichos alumnos realizan dos tipos de controles a lo largo del curso académico:

● Controles escritos: cada alumno realiza varios a lo largo del curso, y se definen por el n° de control, el n° de preguntas de que consta y la fecha de realización (la misma para todos los alumnos que realizan el mismo control). Evidentemente, es importante almacenar la nota de cada alumno por control.

● Prácticas: se realiza un n° indeterminado de ellas durante el curso académico, algunas serán en grupo y otras individuales. Se definen por un código de práctica, título y el grado de dificultad. En este caso los alumnos pueden examinarse de cualquier práctica cuando lo deseen, debiéndose almacenar la fecha y nota obtenida.

● En cuanto a los profesores, únicamente interesa conocer (además de sus datos personales: DNI y nombre), quien es el qué ha diseñado cada práctica, sabiendo que en el diseño de una práctica puede colaborar más de uno, y que un profesor puede diseñar más de una práctica. Interesa, además, la fecha en que ha sido diseñada cada práctica por el profesor correspondiente.

**Análisis del problema**

* Se debe realizar la creación de un programa que pueda crear exámenes y practicas, calificar dichos exámenes y practicas, también debe verificar que solo se pueda hacer dos examenes en el en semestre y que las practicas no tengan limite para hacerlas

**Organización de idea**

Se debe realizar la creación de un algoritmo el cual debemos hacer usos de librerías y para así poder registrar, consultar y mostrar los datos de las personas, además almacenar estos datos para así tener toda la información requerida.

**Objetivos**

* **Genérales**
* Crear un Aplicativo con interfaz que permita almacenar y mostrar practicas, exámenes y notas
* **Específicos**
* Crear un Archivo .json para poder almacenar la información y consultarla de manera dinámica.
* Crear un HashMap para poder almacenar la información y consultarla de manera dinámica.
* Identificar el problema
* Investigar las soluciones posibles.

**Entradas y Salidas de datos (Presentación de los datos)**

* Usuario,
* contraseña.
* Preguntas.
* Respuestas.
* Fecha de inicio.

**Obtención de información**

**Se tomo dicha información gracias al estudio dedicado a https://profile.w3schools.com/**

**Programa en java**

**Clases:**

* Menu.java
* Registro.java
* VeriFicacion.java
* Principal.java
* BuscarPersona.java
* FinFormulario.java
* DatosPersona.java
* EliminarPersona.java

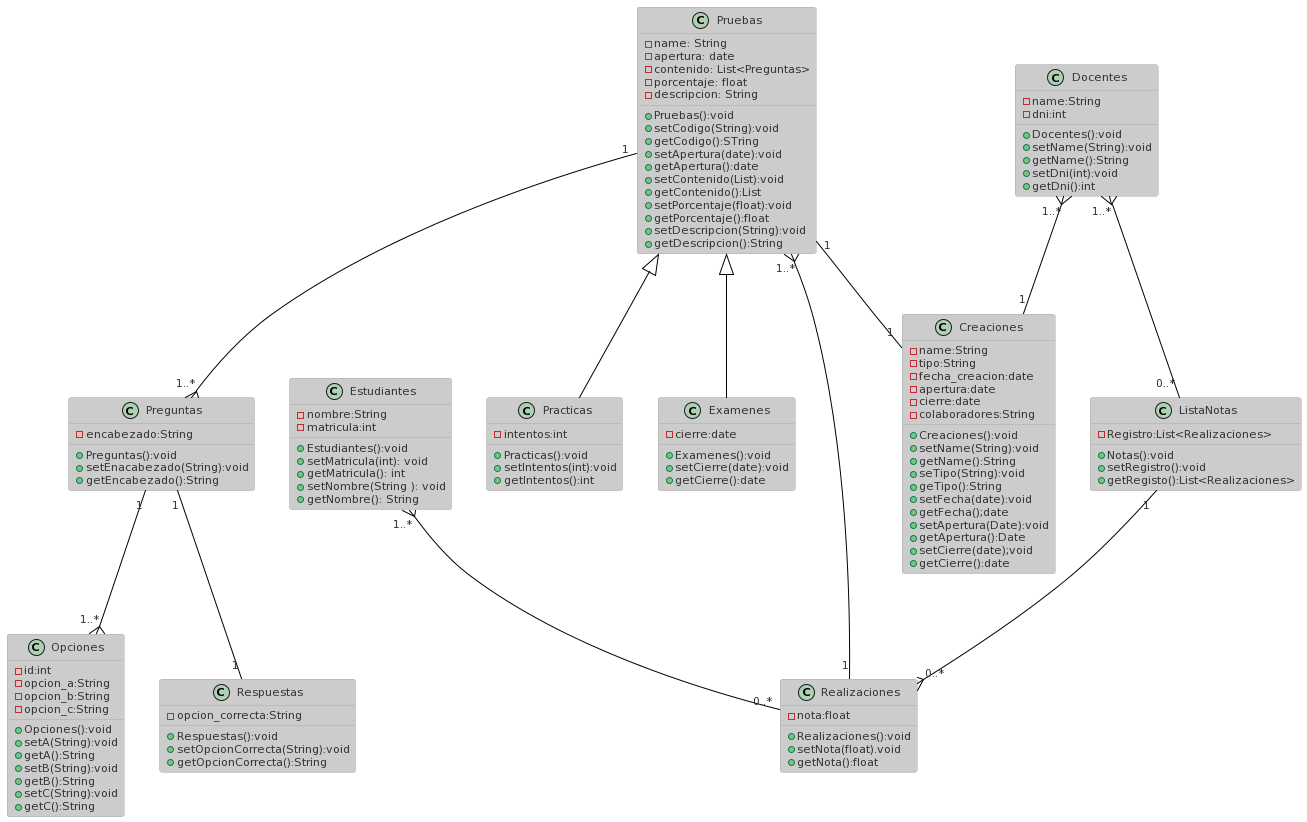
**Funciones:**

* Pedir los datos de las personas.
* Almacenarlos en un archivo .txt.
* Extraer los datos del archivo y almacenarlos en Arraylist.
* Buscar si una persona está registrada.
* Eliminar los datos de una persona.

**Archivo:**

* Usuarios.txt

**Diagrama de clases**



**Codigo del diagrama de clases**

|  |
| --- |
| @startuml !theme reddress-lightgreen class **Estudiantes** {  - nombre:**String** - matricula:int  + **Estudiantes()**:void + setMatricula(int): void + getMatricula**()**: int + setNombre(**String** ): void + getNombre**()**: **String** }  class **Pruebas**{  - name: **String** - apertura: date - contenido: **List**<**Preguntas**> - porcentaje: float - descripcion: **String** + **Pruebas()**:void + setCodigo(**String**):void + getCodigo**()**:**STring** + setApertura(date):void + getApertura**()**:date + setContenido(**List**):void + getContenido**()**:**List** + setPorcentaje(float):void + getPorcentaje**()**:float + setDescripcion(**String**):void + getDescripcion**()**:**String** }  class **Practicas**{  - intentos:int  + **Practicas()**:void + setIntentos(int):void + getIntentos**()**:int }  class **Examenes**{  - cierre:date  + **Examenes()**:void + setCierre(date):void + getCierre**()**:date  }  class **Docentes**{  - name:**String** - dni:int  + **Docentes()**:void + setName(**String**):void + getName**()**:**String** + setDni(int):void + getDni**()**:int  }  class **Creaciones**{  - name:**String** - tipo:**String** - fecha\_creacion:date - apertura:date - cierre:date - colaboradores:**String** + **Creaciones()**:void + setName(**String**):void + getName**()**:**String** + seTipo(**String**):void + geTipo**()**:**String** + setFecha(date):void +getFecha**()**;date + setApertura(**Date**):void + getApertura**()**:**Date** + setCierre(date);void + getCierre**()**:date  }  class **ListaNotas**{  - **Registro**:**List**<**Realizaciones**>  + **Notas()**:void + setRegistro**()**:void + getRegisto**()**:**List**<**Realizaciones**> }  class **Realizaciones**{ - nota:float  + **Realizaciones()**:void + setNota(float).void + getNota**()**:float }   class **Preguntas**{ - encabezado:**String** + **Preguntas()**:void + setEnacabezado(**String**):void + getEncabezado**()**:**String** }  class **Opciones**{ -id:int - opcion\_a:**String** - opcion\_b:**String** - opcion\_c:**String** + **Opciones()**:void + setA(**String**):void + getA**()**:**String** + setB(**String**):void + getB**()**:**String** + setC(**String**):void + getC**()**:**String** }  class **Respuestas**{ - opcion\_correcta:**String** + **Respuestas()**:void + setOpcionCorrecta(**String**):void + getOpcionCorrecta**()**:**String** }  **Estudiantes** "1..\*" }-- "0..\*" **Realizaciones  Docentes** "1..\*" }-- "0..\*" **ListaNotas Docentes** "1..\*" }-- "1" **Creaciones  ListaNotas** "1" --{ "0..\*" **Realizaciones  Pruebas** "1" -- "1" **Creaciones Pruebas** "1..\*" }-- "1" **Realizaciones Pruebas** "1" --{ "1..\*" **Preguntas Pruebas** <|--  **Examenes Pruebas** <|--  **Practicas  Preguntas** "1" --{ "1..\*" **Opciones Preguntas** "1" -- "1" **Respuestas** @enduml |

**Manual del usuario o persona encargada**

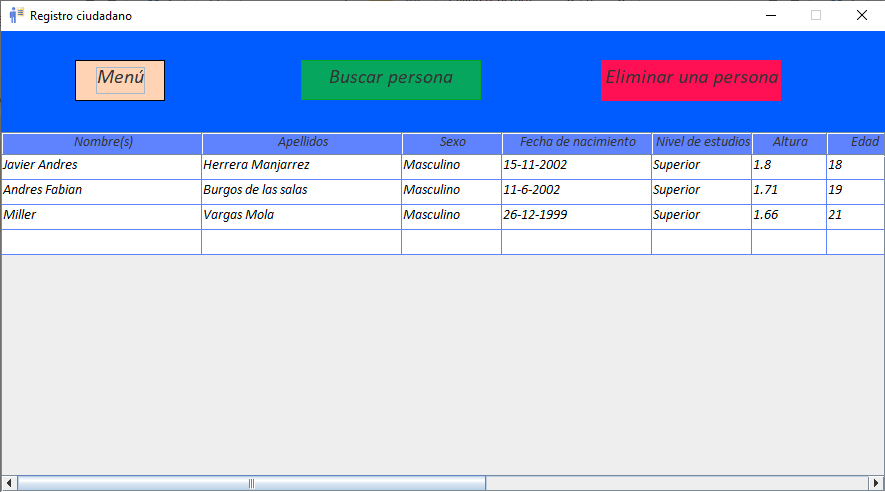
1. Al ejecutar el programa le se muestra el menú de inicio junto con las funciones principales del programa.



1. Al presionar el botón principal se actualizará la ventana dando a conocer un formulario que será correctamente diligenciado para registrar una persona.



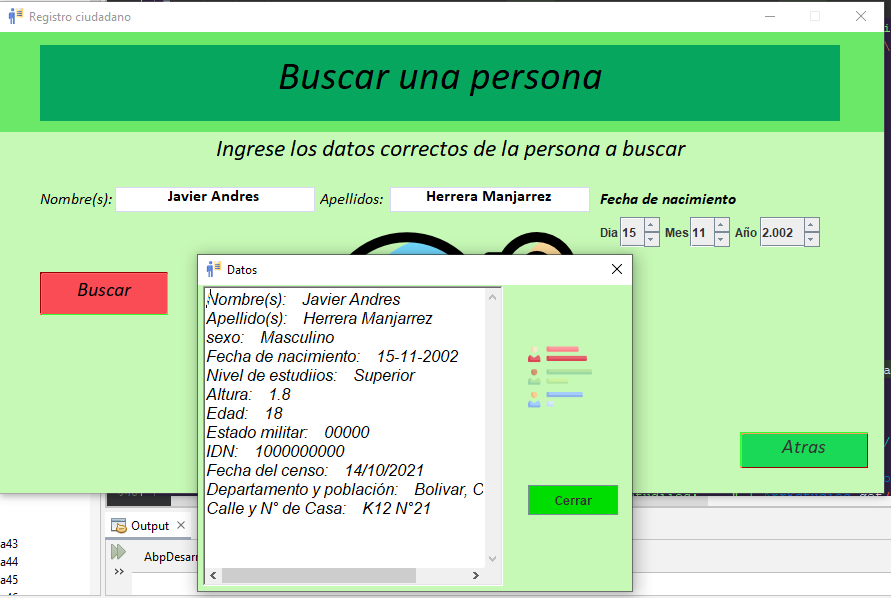
1. Si presiona el botón archivos vera una cuadricula donde esta la información de las personas registradas en el programa y se dan a conocer dos opciones más del programa: eliminar y Buscar.



1. En el botón buscar persona vera lo siguiente.

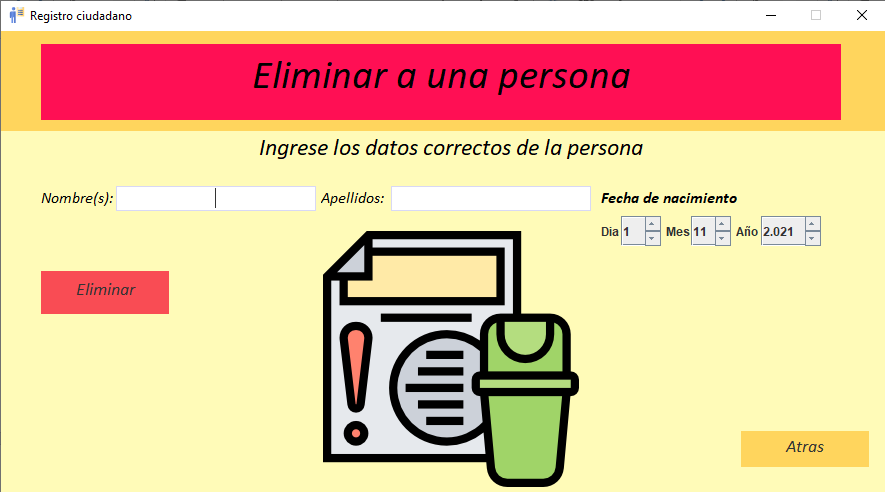


debe digitar correctamente los capos y si la persona esta el los archivos del programa se mostrara una ventana emergente con los datos de la persona o las posibles coincidencias sino observará un mensaje diciendo que no hay una persona con estas características.



1. Al presionar el botón Eliminar persona visualizara la siguiente ventana.

Aquí observará que le piden digitar los datos correctos de la persona la cual se eliminaran sus datos.



**Conclusión**

Después de ver los términos de Software, Librerías, manejo de Archivos; Podemos inferir que aplicando todas estas tecnologías podremos solventar el problema.

**Bibliografía**

[W3school](https://profile.w3schools.com/)

[sololearn](https://www.sololearn.com/profile/18005154)

[**Programmer click**](https://programmerclick.com/article/6901279413/)