# Universidad Autónoma de Baja California

## Facultad de Ingeniería Arquitectura y Diseño

## Ingeniero en computación
## Ingeniero en Software y tecnologías emergentes

**Materia:** Programación Estructurada / Clave **36276**

**Alumno:** Kevin Alejandro Gonzalez Torres

**Matrícula:** 372354

**Maestro:** Pedro Núñez Yépiz

**Actividad Numero:** 12

**Tema - Unidad:** Estructuras

**Ensenada Baja California a 15 de octubre del 2022**

# Universidad Autónoma de Baja California

## Facultad de Ingeniería Arquitectura y Diseño

### 1. INTRODUCCIÓN

En el lenguaje de programación C, las estructuras son una característica fundamental que permite a los programadores definir tipos de datos personalizados para organizar y almacenar múltiples variables relacionadas bajo un solo nombre. Estas estructuras son una forma eficiente de representar objetos complejos en un programa, ya que permiten agrupar datos de diferentes tipos en una sola entidad. En esencia, una estructura en C es una colección de variables de diferentes tipos de datos (como enteros, flotantes, caracteres, punteros, etc.) que se agrupan bajo un nombre común. Cada variable en una estructura se denomina "miembro" o "campo". Estos campos pueden tener nombres significativos que facilitan la manipulación de los datos.

La palabra clave typedef crea un alias "Punto" para la estructura "struct Punto", lo que facilita la declaración y uso de variables de tipo "Punto" en el código. Esta práctica implica la creación de un programa en el lenguaje de programación C que tenga la capacidad de generar el Identificador Único de Registro de Población (CURP) de cualquier individuo al ingresar únicamente los datos esenciales, tales como nombre, apellidos, fecha de nacimiento, entidad federativa y sexo. Este programa debe estar equipado para manejar excepciones y verificar la validez de los datos ingresados, asegurando así la integridad y precisión de la CURP generada. En resumen, las estructuras en C son una forma esencial de organizar y manipular datos complejos, y typedef es una característica útil para simplificar la declaración y uso de estructuras en el código.

### 2. COMPETENCIA

- Programación en C
- Diseño de estructuras de datos
- Manipulación de datos complejos
- Uso de typedef
- Validación y manejo de excepciones
- Desarrollo de software que cumple con requerimientos específicos
- Integridad y precisión de datos

## 3. FUNDAMENTOS

Cadenas:
https://www.programiz.com/c-programming/c-strings

Structs:
https://www.geeksforgeeks.org/structures-c/

Funciones:
https://www.w3schools.com/c/c_functions.php

Typedef:
https://www.geeksforgeeks.org/typedef-in-c/
https://learn.microsoft.com/es-es/cpp/c-language/typedef-declarations?view=msvc-170

Struct
https://www.w3schools.com/c/c_structs.php
https://www.geeksforgeeks.org/structures-c/

## 4. PROCEDIMIENTO

# PRÁCTICA 11

**1.- Cargar Archivo : El programa deberá cargar el vector de registros desde el archivo de texto (solo podrá cargarse una sola vez el archivo)**
**2.- Agregar : El programa deberá ser capaz de agregar un 10 registros al arreglo y al final del archivo de texto. (Generar automáticamente los datos).**
**3.- Eliminar : El programa deberá buscar una matrícula en el vector por medio del método de búsqueda más óptimo. Utilizar banderas para escoger el método más adecuado., imprimir el registro y preguntar si se quiere eliminar el registro, (al cerrar el programa se deberar agregar al archivo borrados el registro o registros eliminados, asi se debera mantener dos archivos uno con datos validos y otro con los datos que se borraron)**
**4.- Buscar : El programa deberá buscar una matrícula en el vector por medio del método de búsqueda más óptimo. Utilizar banderas para escoger el método más adecuado. Mostrar los datos en forma de registro**
**5.- Ordenar : El programa deberá ordenar el vector por medio del método de ordenación más óptimo. Utilizar banderas para escoger el método más adecuado se ordenará por el campo llave (matrícula)**
**6.- Mostrar Todo: El programa deberá mostrar todos los registros del vector tal y como están en ese momento ordenado o desordenado. (mostrar en forma de tabla, y de 40 en 40)**
**7.- Generar Archivo : El programa deberá preguntar al usuario el nombre del archivo, solo nombre sin extensión, el programa generará un archivo con el nombre proporcionado por el usuario con extensión .txt los datos que pondrá en el archivo de texto serán idénticos a los contenidos en el Vector de registros. (ordenado o desordenado). El programa podrá generar múltiples archivos para comprobar las salidas.**
**8.- Cantidad de registros en archivo : El programa deberá llamar a un archivo externo, donde mande ejecutar el archivo y como parametros el nombre del archivo que se desea evaluar, el programa externo deberar ser capaz de retornar un valor enero que sea la cantida de registros que contiene el archivo en cuestion**
**9.- Mostrar Borrados: El programa deberá mostrar el archivo de texto tal y como se visualiza con la cantidad de registros que se eliminaron del archivo original y que fueron marcados en su momnto como registros eliminados**

## 5. RESULTADOS Y CONCLUSIONES

En esta práctica hicimos un programa de gestión de registros de estudiantes con las siguientes características:

Estructura del Programa:
Está organizado en funciones para tareas específicas.
Utiliza la estructura Tstudents para representar la información de los estudiantes.
Funciones Específicas:

- autoDataReg(): Genera datos de registro automáticamente.
- linearSearch(): Búsqueda lineal en el array de estudiantes.
- binarySearch(): Búsqueda binaria en el array (asume orden).
- selectionSort(): Ordena el array con selección.
- quicksort(): Ordena el array con ordenación rápida.
- readTextFile(): Lee datos desde un archivo de texto.
- counterRegisters(): Cuenta registros en un archivo.
- printRegister(): Imprime registros en páginas de 40.
- writeTextFile() y writeDeletedTextFile(): Escribe registros en archivos de texto.

Interacción con el Usuario:
Menú interactivo para seleccionar operaciones.
Uso de validate() para entrada de datos válidos.
Persistencia de Datos:

Carga datos desde un archivo y los guarda al final.
Áreas de Mejora:

El menú podría permitir recargar archivos.
autoDataReg() podría modularizarse.
Se pueden agregar comentarios para mejorar la comprensión.

## 6. ANEXOS

GTKA_RP12_PE_ANEXOS

## 7. REFERENCIAS

**Diseño de algoritmos y su codificación en lenguaje C**
Corona, M.A. y Ancona, M.A. (2011)..
España: McGraw-Hill.
ISBN: 9786071505712

**Programación estructurada a fondo:implementación de algoritmos en C**
:Pearson Educación.Sznajdleder, P. A. (2017)..
Buenos Aires,Argentina: Alfaomega

**Como programar en C/C++**
H.M. Deitel/ P.J. Deitel
Segunda edición
Editorial: Prentice Hall.
ISBN:9688804711

**Programación en C.Metodología, estructura de datos y objetos**
Joyanes, L. y Zahonero, I. (2001)..
España:McGraw-Hill.
ISBN: 8448130138