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: Pedro Carlos Balderrama Campos

Matrícula: 372267

Maestro: Pedro Núñez Yépiz

Actividad No. : 8

Tema - Unidad : Teoria arreglos y funciones

Ensenada Baja California a 24 de marzo del 2022

Universidad Autónoma de Baja California

Facultad de Ingeniería Arquitectura y Diseño

1. INTRODUCCIÓN

Durante esta práctica, nos hemos dedicado al desarrollo de una serie de funciones destinadas a manipular, leer, copiar y almacenar cadenas de texto en nuestro lenguaje de programación. Estas funciones tienen como objetivo realizar diversas funciones, tales como: crear una cadena a partir de dos cadenas externas, crear un vector bidimensional a partir de una cadena unidimensional, crear una cadena aleatoria a partir de ciclos y asegurarnos de que los datos no se repitan, etc. En este informe, presentaremos las implementaciones de las funciones solicitadas, comenzando con funciones simples que transforman la cadena de diferentes maneras.

2. COMPETENCIA

Esta actividad es una oportunidad ideal para afinar las habilidades en el manejo de cadenas de texto durante el aprendizaje de programación. A través de varios ejercicios prácticos, se descubre cómo efectuar diversas operaciones con las cadenas, tales como manipular las cadenas de la manera deseada.

3. FUNDAMENTOS

La habilidad para manipular estos conjuntos de datos, redistribuyéndolos entre diferentes estructuras (de vectores individuales a matrices), no solo refleja la versatilidad en el manejo de la memoria y el almacenamiento en programación, sino que también enfatiza la importancia de las operaciones de control de flujo y la lógica condicional en la creación de programas eficientes y dinámicos. Este enfoque práctico hacia las estructuras de datos como vectores y matrices ilustra conceptos fundamentales de la informática, tales como la iteración a través de colecciones de datos, la generación de números aleatorios sin repetición, y la organización de datos en estructuras multidimensionales para su manipulación y análisis.

4. PROCEDIMIENTO

La implementación se desarrolla en funciones dedicadas a tareas particulares como la generación de números aletarois para despues ser almacenados en una cadena, la combinación de datos, y la visualización de resultados. Cada función es cuidadosamente documentada y probada para asegurar su correcto funcionamiento. Este enfoque no solo facilita la comprensión y el mantenimiento del código, sino que también refuerza el aprendizaje de conceptos fundamentales en la programación y el manejo de estructuras de datos.

5. RESULTADOS Y CONCLUSIONES

La realizacion de este codigo fue bastante variada, ya que se utilzaron la mayoria de los conocimientos previos vistos en clase, incluyendo la definicion de variables globales, la declaracion de funciones, etc. La unica parte que tuve que cambiar del codigo fue el hecho de generar una cadena 4x5, el profesor pedia una matriz 4x4 pero al tener 20 numeros en mi vector, me sobrarian 4 valores, por lo que opte por generar una matriz mas grande, tal como mencione antes.
Fragmento del codigo en cuestion:

```
void mat_fill()
{
    clear();
    int l = 0;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if (l < te * 2)
            {
                mat[i][j] = vec3[l];
                l++;
            }
        }
    }
    printf("Matriz de 4x5 llenada con exito!");
    pause();
}
```

Cabe resaltar que al momento de desarrollar mis funciones me tome la libertad de realizar unas cuantas extra por mi cuenta, como por ejemplo “clear()” y “pause()”, las cuales me ayudan a ordenar un poco mas mi uitilizando cosas como limpiar pantalla y una “pausa” (La cual solo espera a que el usuario presione Enter para continuar).

```
fflush(stdin);
gets(ch);
vec_man[i] = atoi(ch);
```

Aqui utilice primero un vector para despues convertirlo a numero entero utilizando atoi(), asi, en caso de que el usuario utilice una letra en el valor por error, el programa no generara una fallo.

En conclusión, este proyecto fue una excelente oportunidad para aplicar conocimientos de programación previamente adquiridos, a la vez que se introdujeron innovaciones prácticas como ajustar la estructura de datos a una matriz 4x5 y desarrollar funciones auxiliares para mejorar la interacción del usuario. Estos cambios no solo cumplieron con los requisitos del proyecto, sino que también demostraron una comprensión profunda de cómo la flexibilidad y la atención al detalle pueden impactar positivamente en el desarrollo de software.

6. 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