FACULTAD DE INGENIERÍA Y CIENCIAS

UNIVERSIDAD ADOLFO IBÁÑEZ

Lenguajes y Paradigmas de la programación.

Tarea 1 pizzería en C.

Profesora:
María Loreto Arriagada

Integrantes:
Juan Pablo Aedo
Pedro Hodar
Joaquín Salazar
Matías Salinas

Sección: 1

Grupo: 4

Fecha: 03 - 04 - 2025

1) Informe de diseño y justificación de la solución:

El código para la tarea está estructurado en 5 archivos de código en C, cada uno cumpliendo una función específica, facilitando así la organización del programa. Estos archivos corresponden a Main.c, Metrics.c, Metrics.h, Utils.c y Utils.h, los cuales fueron compilados para un correcto funcionamiento en conjunto.

Main.c

En el archivo main.c se inicia verificando los argumentos ingresados por el usuario para luego realizar una lectura del archivo .csv con la función *"read_csv"* y se almacenan los datos en el arreglo *"Order"*. Luego, las métricas dadas por el usuario se analizan y se mapean a las funciones correspondientes para procesarlas y almacenar su resultado. Por último se imprimen los resultados de las métricas solicitadas.

Metrics.c

En esta parte se diseñan y guardan las funciones que permitirán procesar los datos del archivo csv y obtener de este la información solicitada en la tarea, como la pizza más vendida, ingrediente más vendido, etc.

Utils.c

El archivo utils.c implementa funciones auxiliares para el programa, específicamente la función read_csv. Es una parte clave del programa, ya que permite cargar los datos necesarios para calcular las métricas. También incluye manejo de errores, como problemas al abrir el archivo o al asignar memoria.

Metrics.h

Este archivo de apoyo declara un conjunto de funciones asociadas a las métricas del programa. Define un tipo de puntero a función (MetricFunction) y las firmas de las funciones asociadas a calcular métricas de la venta de pizzas.
Adicionalmente este archivo permite que el resto de códigos puedan utilizar dichas funciones siendo fundamental para organizar el código.

Utils.h

El archivo de cabecera utils.h define la estructura Order, que representa los datos de una orden de pizza (nombre, cantidad, fecha, precio, etc.), y declara la función read_csv, encargada de leer un archivo CSV y cargar los datos en un arreglo de estructuras Order. Su propósito es facilitar la modularidad y reutilización del código en otros archivos fuente como main.c y metrics.c.

Makefile

El archivo Makefile es un script que automatiza la compilación del proyecto. Define las reglas necesarias para compilar los archivos fuente (main.c, metrics.c, utils.c) en archivos objeto (main.o, metrics.o, utils.o) y enlazarlos para generar el ejecutable app1

2) Sección de reflexiones finales y autoevaluación:

Juan Pablo Aedo: Para mí aprender C fue un gran desafío, superado gracias a videos de YouTube y la asistencia de Copilot. Aunque logré comprender el lenguaje y leer código con mayor facilidad, no desarrollé tanto mi fluidez al programar, ya que no programé todo yo y usé Copilot para cosas como algunos esqueletos de código, centrándome más en la detección de errores. Aunque podría desarrollar programas sin problema, mi falta de práctica escribiendo código limita mi agilidad. Gracias a este trabajo aprendí la importancia de escribir el código por mí mismo al 100% para evitar errores y acostumbrarme a la sintaxis, aprendizaje que utilizaré para los próximos lenguajes y paradigmas que estudiemos.

Pedro Hodar: Trabajar con C fue un desafío, ya que mi conocimiento era limitado, lo que hizo que el proceso fuera lento y basado en prueba y error. Sin embargo, apoyarme en inteligencia artificial, vídeos de YouTube e investigación propia me ayudó a comprender mejor el lenguaje y armar código.

Joaquín Salazar: Este trabajo de programación fue un desafío personal, ya que desconocía el lenguaje “C”, por lo cual, me apoye en inteligencia artificial e información digital extra para poder aprender y comprender cómo crear los códigos necesarios para resolver la tarea y, al mismo tiempo, entender cómo se ligaban los diferentes códigos unos con los otros, lo cual me pareció interesante y un desafío grato de resolver.

Matías Salinas: Para mí lo más complejo de esta tarea fue comprender cómo utilizar C, debido a que nunca antes había trabajado con este lenguaje de programación. Para esto fue de gran ayuda investigar por mi cuenta en videos de YouTube junto con el uso de la herramienta Microsoft Copilot para revisar los errores de código que fuimos teniendo en el proceso. Una lección que obtuve de esta tarea fue la importancia de tener un código bien estructurado según las diversas funciones de cada archivo, dado a que esto facilita el orden al momento de programar.

3) Explicación de cómo se usó IA:

Para la realización de esta tarea, se utilizó principalmente la herramienta Microsoft Copilot como asistente de programación. Su uso fue fundamental para optimizar el tiempo y facilitar la escritura del código en C. Inicialmente, se empleó para generar esqueletos de código que sirvieran como base para el desarrollo del programa. Por ejemplo, uno de los prompts utilizados fue:

"¿Puedes generar un código en C que lea un archivo CSV?"

A partir de estas sugerencias, se adaptaron y modificaron los fragmentos generados según las necesidades específicas del proyecto.

Además, Microsoft Copilot resultó de gran ayuda en la detección y corrección de errores. Problemas menores, como la omisión de un punto y coma (“;”) o errores de sintaxis básicos, fueron fáciles de identificar y corregir. Sin embargo, cuando los errores se volvieron más complejos y difíciles de resolver manualmente, la herramienta fue clave para comprender el problema y encontrar soluciones adecuadas. En estos casos, enviar el código a Copilot y solicitar una explicación detallada junto con una posible corrección permitió superar obstáculos que, de otro modo, habrían tomado más tiempo en resolver.

En general, el uso de Copilot no solo agiliza el proceso de desarrollo, sino que también permitió un mejor entendimiento de los errores y una mayor eficiencia en la programación.