



UNIVERSIDAD ABIERTA Y A DISTANCIA DE MÉXICO

División de Ciencias Exactas, Ingeniería y Tecnología

Desarrollo de software

**Semestre:** Primer semestre.

**Unidad didáctica:** 01 – Fundamentos de programación.

**Unidad de aprendizaje:** Unidad 3. Funciones y estructuras de datos.

**Actividad:** 2. Representación de módulos en diagramas de flujo y pseudocódigo.

**Nombre del estudiante:** Orlando Antonio Maturano Pizaña

**Matrícula:** ES251107915

**Grupo:** DS-DFPR-2501-B1-013

**Figura académica:** JOSÉ MANUEL NORIEGA BARRERA

**Fecha de entrega:** 5 de marzo de 2025

Estado de México, a 5 de marzo del 2025.

Diseño: DL-CPL

## ÍNDICE

<b>INTRODUCCIÓN</b>	1
<b>DESARROLLO DE LA ACTIVIDAD</b>	2
<b>Análisis del problema</b>	2
<b>Pseudocódigo</b>	4
<b>Diagrama de flujo</b>	7
<b>Prueba de escritorio</b>	8
<b>Desarrollo del código en lenguaje C</b>	11
<b>CONCLUSIÓN</b>	17
<b>CITAS DE AUTOR</b>	18
<b>REFERENCIAS BIBLIOGRÁFICAS</b>	18

## INTRODUCCIÓN

La representación de módulos en diagramas de flujo y pseudocódigo surge como respuesta a la necesidad de estructurar soluciones algorítmicas antes de su implementación en código. Históricamente, estas herramientas han sido fundamentales en la programación estructurada, permitiendo visualizar procesos complejos de manera intuitiva y facilitando la comunicación entre desarrolladores. Su importancia radica en que descomponen problemas en tareas específicas, promueven la claridad lógica y reducen errores en etapas tempranas.

El objetivo de esta actividad es fortalecer la capacidad de análisis y diseño algorítmico mediante la traducción de problemas cotidianos a representaciones visuales (diagramas de flujo) y descripciones textuales estructuradas (pseudocódigo), para luego codificarlas en C. Se delimitan aspectos como:

1. Identificación de patrones comunes en problemas simples.
2. Diseño jerárquico de módulos (funciones).
3. Uso de estructuras de control y datos.

En esta actividad se busca integrar el aprendizaje basado en problemas y casos, se busca dominar metodologías de programación modular, esenciales para resolver desafíos complejos con eficiencia y rigor técnico.

## DESARROLLO DE LA ACTIVIDAD

**Descripción de aspectos:**

## i. Problemática planteada

El programa debe calcular la ingesta calórica diaria y sugerir actividades físicas según el estado de salud del usuario. Esta necesidad surge de la creciente demanda de herramientas digitales para la promoción de hábitos saludables, como señala Joyanes (2005), quien enfatiza que: «la programación modular es esencial para desarrollar aplicaciones adaptables a diversos perfiles de usuarios» (p.45).

## ii. Información de entrada requerida

- Datos del usuario: Nombre, estado de salud (bueno, malo, lesiones, afecciones), sexo (H,M), edad, peso (kg), altura (cm).
- Requerimientos técnicos: UTF-8, portabilidad multiplataforma y gestión de buffers.

## iii. Módulos a implementar

1. Registro de datos: Captura la información del usuario.
2. Cálculo de calorías : Aplica la fórmula Harris-Benedict ajustada por sexo.
3. Generación de sugerencias. Ofrece recomendaciones basadas en el estado de salud. Según Kernighan y Ritchie (1988). «la modularización facilita el mantenimiento y la escalabilidad del código» (p.67), lo que valida este enfoque.

## iv. Resultados esperados

- Salida principal: Calorías diarias recomendadas y sugerencias de actividad física.

Ejemplo:

Hola, José Valenzuela. Aquí están tus resultados:

Calorías diarias recomendadas: 1410.25

Se recomiendan las siguientes sugerencias de actividad física:

- 30 minutos de caminata.
- 15 minutos de ejercicios de fuerza.
- 10 minutos de estiramientos.

Cómo señala Norman Donald A. (2013) en The Design of Everyday Things, «la retroalimentación efectiva debe ser inmediata y comprensible para guiar al usuario hacia acciones significativas» (p.78). Esto se traduce en la salida numérica precisa, las sugerencias de actividad física y una interfaz intuitiva, cuyos mensajes deben estar estructurados y libres de ambigüedades.

## Elementos técnicos para la solución

### Librerías

```
#include <stdio.h>
#include <string.h>
#include <locale.h>
#include <ctype.h>
```

```
#include <stdlib.h>
```

### Configuración para Windows

```
#ifdef _WIN32  
#include <windows.h>  
#endif
```

### Variables

```
char nombre[50], estadoSalud,[50] sexo;  
int edad;  
float peso, altura, calorias;
```

### Constantes

```
#define C_HOMBRE 5  
#define C_MUJER -161
```

### Estructuras de control

- Bucle do-while: Para múltiples cálculos.
- Condicionales if-else: Para determinar sugerencias.

### Módulos y funciones

```
void registroUsuario();  
float calcularCalorias();  
void generarSugerencia();  
void mostrarDatosEstudiante();
```

### Configuración UTF-8 y portabilidad

```
setlocale(LC_ALL, "en_US.UTF-8");  
fflush(stdin); // Limpia el buffer después de scanf.
```

### Pseudocódigo:

Se utilizaron las siguientes abreviaturas: N E (nombre del estudiante), M (Matrícula), G (Grupo), LI M (Llamar Módulo), N U (nombre del usuario).

Orlando Antonio Maturano Pizáñ  
ES 251107975

//Librerías

Incluir <stdio.h>, <string.h>, <locale.h>, <cctype.h>,  
<stdlib.h>

//Configuración para Windows

Si el sistema operativo actual es Windows

Incluir <Windows.h>

Finalizar si es // Fin de configuración.

//Módulo principal ---

Inicio

Imprimir: N; E; // Imprime mis datos de  
M; estudiante  
G;

Repetir

Llamar a Módulo Registrar Usuario

LIM Calcular Calorías

LIM Generar Sugerencia

Imprimir "¿Desea calcular para otro usuario? (S/N)"

Leer respuesta

Limpiar buffer de entrada

Mientras respuesta sea "S"

Fin

Orlando Antonio Maturano Pizaña  
E5251107915

//Submódulo 1:

Procedimiento Registrar Usuario

Imprimir "==== Registro de Usuario ===";  
"Nombre:"

Leer nombre

Imprimir "Estado de salud (bueno, malo, lesiones,  
afecciones):"

Leer estado de salud

Imprimir "Sexo (H/M):"

Leer sexo

Imprimir "Edad:"

Leer edad

Imprimir "Peso (Kg):"

Leer peso

Imprimir "Altura (cm):"

Leer altura

Fin de procedimiento

Orlando Antonio Maturano Pizana  
ES251107915

### II Submódulo 2:

Función CalcularCalorias (sexo, edad, peso, altura)

Constante C\_Hombre = 5

Constante C\_Mujer = -167

Si sexo es "H" o "h" Entonces

$$C = C_{\text{Hombre}}$$

Sino

$$C = C_{\text{Mujer}}$$

Fin Si

$$\text{calorías} = (70 \times \text{peso}) + (6.25 \times \text{altura})$$

$$- (5 \times \text{edad}) + C$$

Devolver calorías

Fin de Función

### II Submódulo 3:

Procedimiento GenerarSugerencias (estadoSalud)

Imprimir "Hola, (NU). Aquí están tus resultados:

Si estado de salud es "bueno", "Sin lesiones",

"Sin afecciones" Entonces.

Imprimir "- 30 minutos de caminata."

Imprimir "- 15 minutos de ejercicios de fuerza!"

Orlando Antonio Maturano Pizáña  
ES251107915

Imprimir "10 minutos de estiramientos."

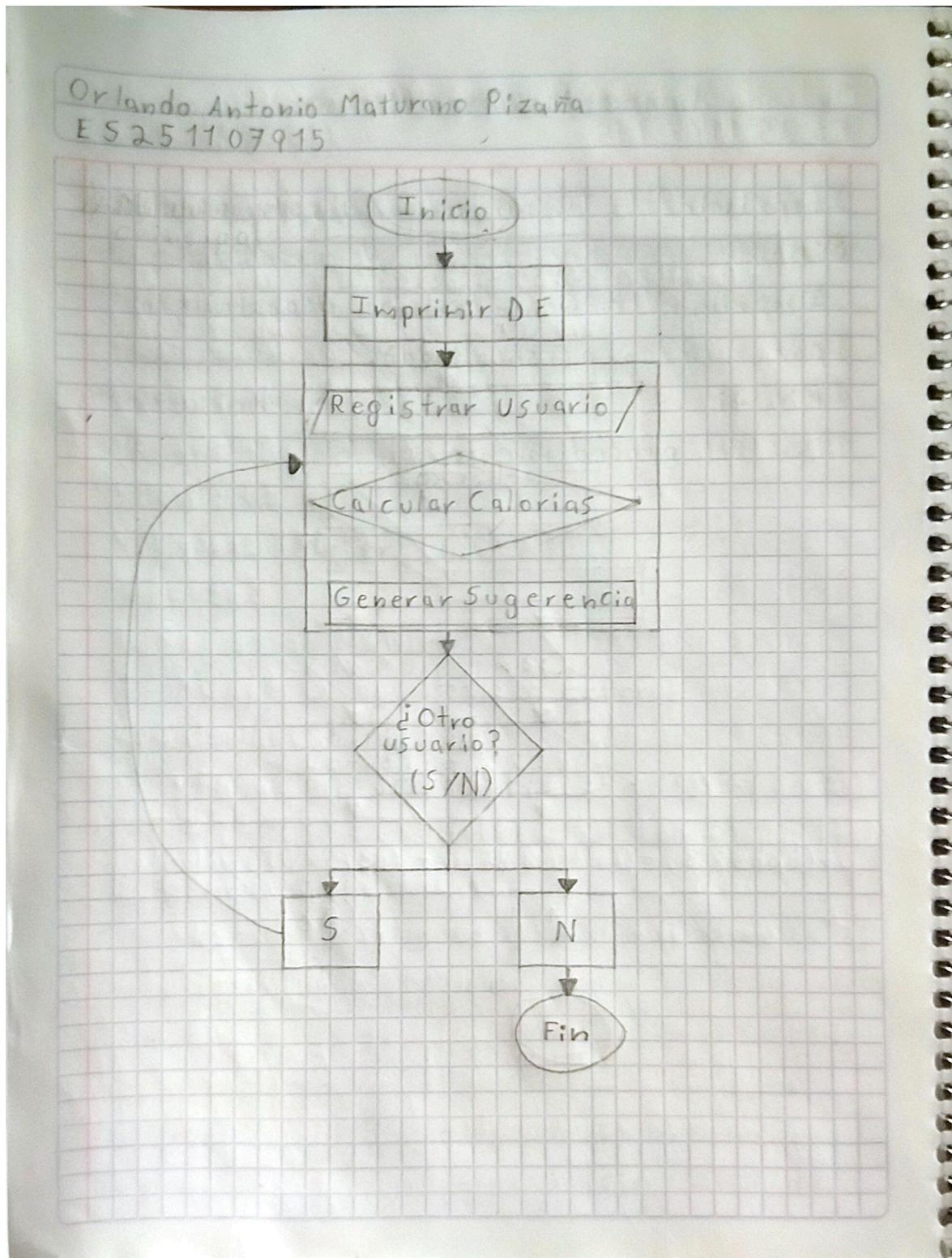
Sino

Imprimir "Consultar a un especialista  
para un plan personalizado."

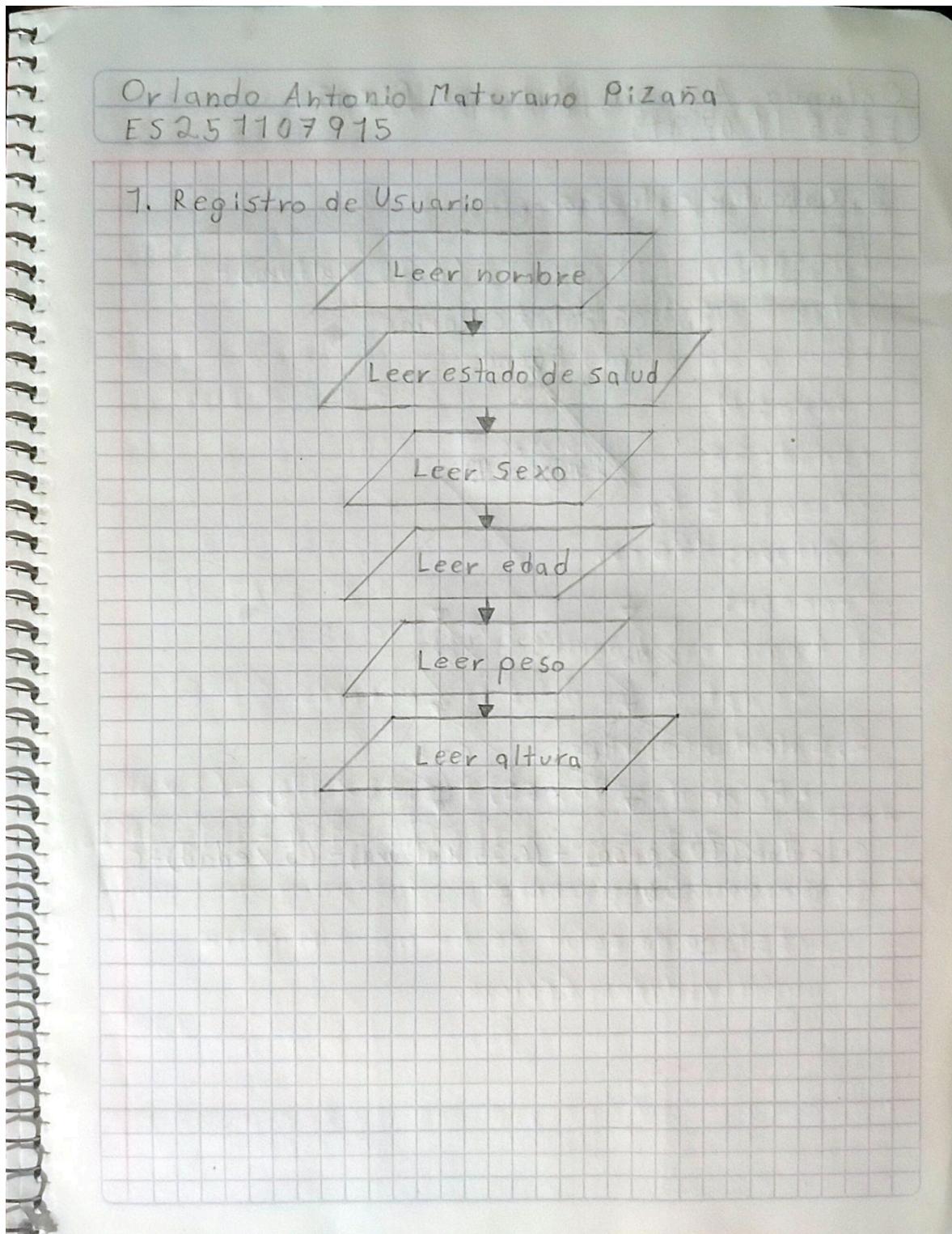
Fin Si

Fin del procedimiento

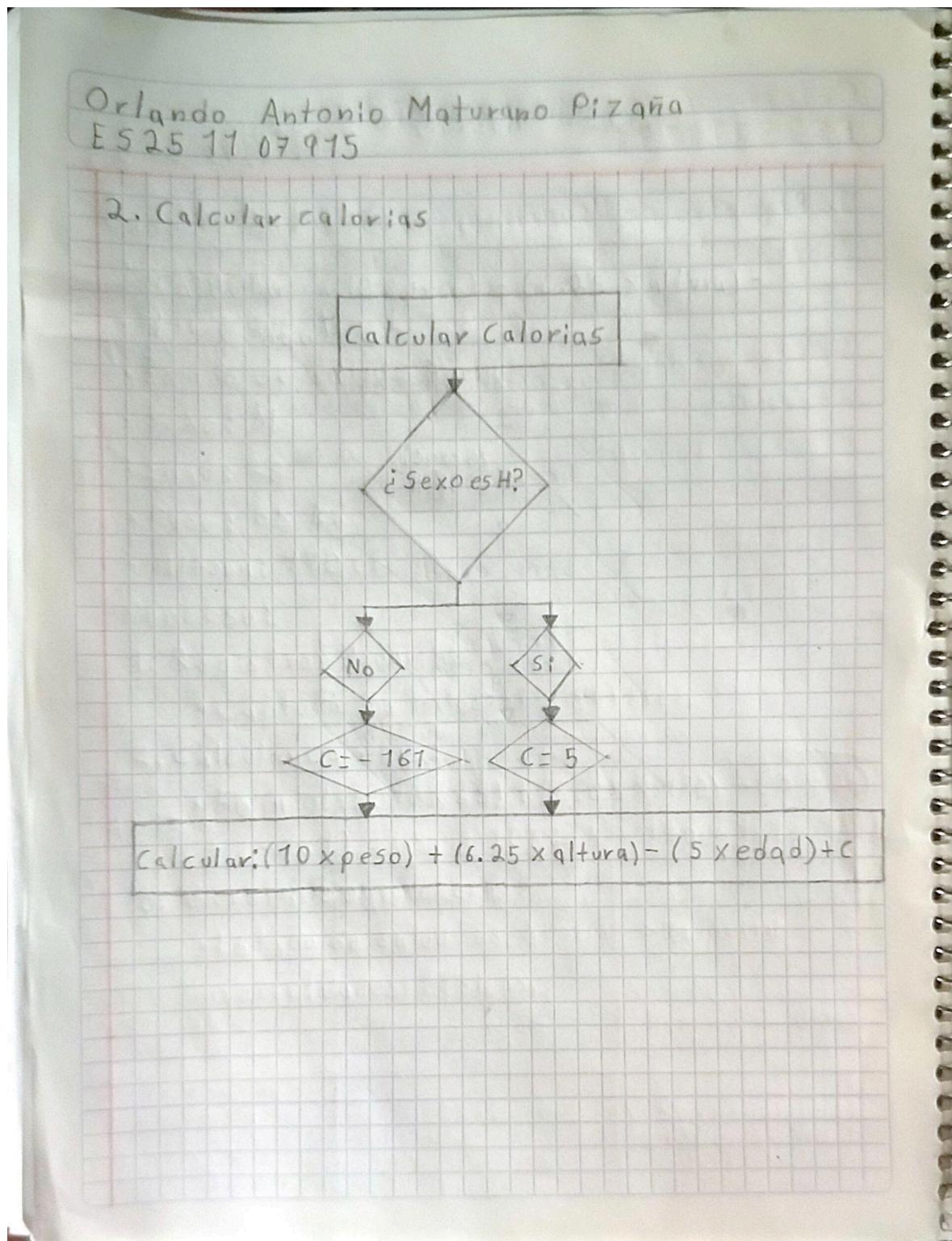
## Diagrama de flujo general



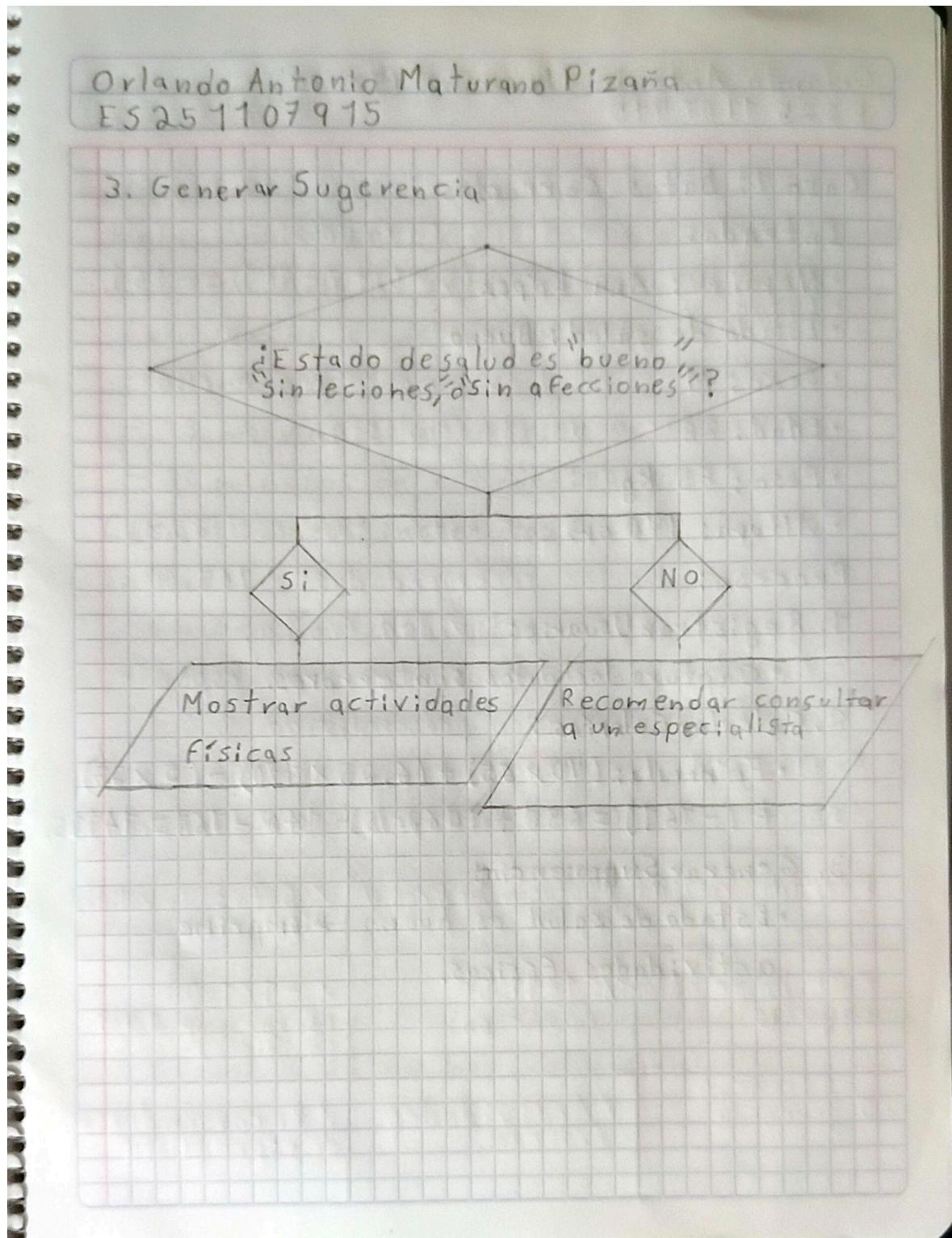
## Diagrama de flujo del módulo 1



## Diagrama de flujo del módulo 2



### Diagrama de flujo del módulo 3



## Prueba de escritorio

Orlando Antonio Maturano Pizana  
ES25 1107915

### Caso 1: Datos Correctos

#### Entrada:

- Nombre: Ana López
- Estado de salud: Bueno
- Sexo: M
- Edad: 28
- Peso: 65 kg
- Altura: 170 cm

#### Proceso:

##### 1. Registrar Usuario:

- Captura los datos sin errores.

##### 2. Calcular Calorías:

$$\begin{aligned} \text{Fórmula: } & (10 \times 65) + (6.25 \times 170) - (5 \times 28) \\ & + (-167) = 650 + 1062.5 - 140 - 167 = 1411.5. \end{aligned}$$

##### 3. Generar Sugerencia:

- Estado de salud es bueno → imprime actividades Físicas.

Orlando Antonio Maturano Pizanía  
ES251107915

Salida esperada:

Hola, Ana López. Aquí estan tus resultados:

Calorías diarias recomendadas: 1411,50

Sugerencia de actividad física:

- 30 minutos de caminata.
- 15 minutos de ejercicios de fuerza.
- 10 minutos de estiramientos.

Caso 2 : Datos incorrectos

Entrada:

- Nombre: Carlos
- Estado de salud: malo
- Sexo: X (inválido)
- Edad: -30 (inválido)
- Peso: abc (inválido)
- Altura: 0 (inválida)

Proceso:

1. Registrar Usuario:

- Error al leer edad, peso, altura (el programa asume 0 si no hay validación).

Orlando Antonio Maturano Pizana  
ES251107915

## 2. Calcular Calorías:

- Fórmula con datos inválidos:

$$(70 \times 0) + (6.25 \times 0) - (5 \times 0) + (-761) = -161.$$

## 3. Generar Sugerencia:

- Estado de salud es malo → recomienda consultar a un especialista.

Salida esperada:

Hola, Carlos. Aquí están tus resultados:

Calorías diarias recomendadas: -161,00

Sugerencias de actividad física:

- Se recomienda consultar a un especialista para un plan personalizado.

ANÁLISIS DE RESULTADOS

Aspecto	Caso 1 (correcto)	Caso 2 (incorrecto)
Entrada válida	✓ Datos numéricos y lógicos	X Datos inválidos
Cálculo de calorías	✓ 1411.50 (preciso)	X -161,00 (Error por datos inválidos)
Sugerencia	✓ Actividades físicas	✓ Recomendación de especialista

Orlando Antonio Maturano Pizaña  
ES251107915

Manejo de  
errores

N/A

X No valida números  
negativos o caracteres

## Desarrollo del código en lenguaje C

```
1 #include <cslibs.h>
2 #include <string.h>
3 #include <locale.h>
4 #include <ctype.h>
5 #include <stdlib.h>
6
7 // Configuración especial para Windows (API para UTF-8)
8 #ifndef _WIN32
9 #include <windows.h>
10#endif
11
12 // Prototipos de funciones:
13 void configurar_encoding(); // Configuración regional UTF-8
14 void mostrarDatosEstudiante(); // Muestra datos del desarrollador
15 void limpiarBuffer(); // Limpia el buffer de entrada
16 void registrarUsuario(char nombre[], char estadoSalud[], char *sexo, int edad, float *peso, float *altura); // Captura y valida datos del usuario
17 float calcularCalorías(char sexo, int edad, float peso, float altura); // Aplica fórmula nutricional
18 void generarSugerencia(char estadoSalud[]); // Genera recomendaciones personalizadas
19
20 // ===== CONFIGURACIÓN UTF-8 MULTIPLATFORMA =====
21 void configurar_encoding() {
22     #ifdef _WIN32
23         // Windows: Cambia la codificación de la consola a UTF-8
24         SetConsoleOutputCP(CP_UTF8); // Salida
25         SetConsoleInputCP(CP_UTF8); // Entrada
26         system("chcp 65001 >nul"); // Forzar UTF-8 en Dev-C++
27     #else
28         // Linux/Mac: Configura localización para caracteres especiales
29         setlocale(LC_ALL, "UTF-8");
30     #endif
31 }
32
33 // ===== FUNCIÓN PRINCIPAL =====
34 int main() {
35     configurar_encoding(); // Asegura compatibilidad con tildes/ñ
36
37     // Variables para almacenar datos del usuario
38     char nombre[50], estadosalud[50], sexo, respuesta;
39     int edad;
40     float peso, altura;
41
42     mostrarDatosEstudiante(); // Muestra información del desarrollador
43
44     // Bucle principal: permite múltiples ejecuciones
45 }
```

The screenshot shows a C programming environment with the following details:

- Title Bar:** F: [Copia] Orlando\_UMADM [Asignaturas] [Modulo 1]Fundamentos de programación] [Unidad 3]DFPR\_U3\_A2 Programa de sugerencias de salud.c - Embarcadero Dev C++ 6.3
- Menu Bar:** Archivo, Edición, Buscar, Ver Proyecto, Ejecutar, Herramientas, ASyle, Ventana, Ayuda
- Toolbar:** Includes icons for file operations like Open, Save, Print, and a search bar.
- Project Explorer:** Shows a project named "Programa de sugerencias de salud.c" under "Proyecto".
- Code Editor:** Displays the C source code for the program. The code includes functions for calculating nutritional needs, displaying developer information, and reading user input. It uses standard C libraries and includes comments explaining the steps of the program.
- Compiler Status:** Shows "Compilador(1)" and "Recursos" in the bottom left.
- Bottom Status Bar:** Shows line numbers (Line: 7), column numbers (Col: 56), selected items (Sel: 0), lines (Lines: 160), length (Length: 5983 Insertar), and parsing status (Done parsing in 0.016 seconds).

F:\Copia\Orlando\UNADM\Asignaturas\Modulo 1\Fundamentos de programación\Unidad 3\DFPR\_U3\_A2\Programa de sugerencias de salud.c - Embarcadero Dev-C++ 6.3

```

    Archivo Edición Buscar Ver Proyecto Ejecutar Herramientas AStyle Ventana Ayuda
    (globales) TDM-GCC 9.2.0 64-bit Release
    Proyecto C:\> Programa de sugerencias de salud.c

    // Captura y valida datos del usuario con bucles de verificación
    void registrarUsuario(char nombre[], char estadoSalud[], char *sexo, int *edad, float *peso, float *altura) {
        printf("==> Registro de usuario ==>\n");
        // Validación 1: Nombre no vacío (usa fgets para espacios)
        do {
            printf("Nombre: ");
            fgets(nombre, 50, stdin);
            nombre[strcspn(nombre, "\n")] = '\0'; // Elimina el salto de línea
        } while(strlen(nombre) == 0); // Repite si está vacío
        // Validación 2: Estado de salud válido
        do {
            printf("Estado de salud [bueno/malo/lesiones/afecções]: ");
            fgets(estadoSalud, 50, stdin);
            estadosSalud[strcspn(estadoSalud, "\n")] = '\0';
        } while(strlen(estadoSalud) == 0);
        // Validación 3: Sexo solo H/M (convierte a mayúsculas)
        do {
            printf("Sexo (H/M): ");
            scanf("%c", sexo);
            limpiarBuffer();
            *sexo = toupper(*sexo); // Estandariza entrada
        } while(*sexo != 'H' && *sexo != 'M');
        // Validación 4: Edad entre 1 y 120 años
        do {
            printf("Edad: ");
            scanf("%d", edad);
            limpiarBuffer();
        } while(*edad < 0 || *edad > 120); // Rango realista
        // Validación 5: Peso positivo
        do {
            printf("Peso (kg): ");
            scanf("%f", peso);
            limpiarBuffer();
        } while(*peso <= 0);
        // Validación 6: Altura positiva
        do {
            printf("Altura (cm): ");
            scanf("%f", altura);
            limpiarBuffer();
        } while(*altura <= 0);
    }

    Compilador (1) Recursos Registro de Compilación Depuración Resultados Console
    Line: 7 Col: 56 Sel: 0 Lines: 160 Length: 5983 Insertar Done parsing in 0.016 seconds

```

F:\Copia\Orlando\UNADM\Asignaturas\Modulo 1\Fundamentos de programación\Unidad 3\DFPR\_U3\_A2\Programa de sugerencias de salud.c - Embarcadero Dev-C++ 6.3

```

    Archivo Edición Buscar Ver Proyecto Ejecutar Herramientas AStyle Ventana Ayuda
    (globales) TDM-GCC 9.2.0 64-bit Release
    Proyecto C:\> Programa de sugerencias de salud.c

    // Calcula calorías usando fórmula Harris-Benedict modificada
    float calcularCalorías(char sexo, int edad, float peso, float altura) {
        const float C_HOMBRE = 5.0f; // Constante para hombres
        const float C_MUJER = -161.0f; // Constante para mujeres
        float constante = (*ouper(sexo) == 'H') ? C_HOMBRE : C_MUJER; // Asigna valor según sexo
        return (10.0f * peso) + // Fórmula:
               (6.25f * altura) -
               (5.0f * edad) +
               constante; // 10*peso + 6.25*altura - 5*edad + C
    }

    // Genera recomendaciones según estado de salud (comparación insensible)
    void generarSugerencia(char estadoSalud[]) {
        // Normaliza a minúsculas para comparar cualquier formato
        for(int i = 0; estadoSalud[i]; i++) {
            estadoSalud[i] = tolower(estadoSalud[i]);
        }
        printf("\nSugerencia de actividad física:\n");
        // Busca coincidencias en el texto ingresado
        if(strstr(estadoSalud, "bueno") || strstr(estadoSalud, "sin lesiones")) {
            printf("- 30 minutos de caminata\n- 15 minutos de ejercicios de fuerza\n- 10 minutos de estiramientos\n");
        } else {
            printf("- Consultar a un especialista para un plan personalizado\n"); // Caso por defecto
        }
    }

    Compilador (1) Recursos Registro de Compilación Depuración Resultados Console
    Line: 7 Col: 56 Sel: 0 Lines: 160 Length: 5983 Insertar Done parsing in 0.016 seconds

```

El programa se compiló sin errores y se ejecuta correctamente:

F:\Copia\Orlando\UNADM\Asignaturas\Modulo 1\Fundamentos de programación\Unidad 3\DFPR\_U3\_A2\Programa de sugerencias de salud.c - Embarcadero Dev-C++ 6.3

```

116     } while(edad <= 0 || edad > 120); // Rango realista
117     // Validación 5: Peso positivo
118     do {
119         printf("Peso (kg): ");
120         scanf("%f", &peso);
121         limpiarBuffer();
122     } while(peso <= 0);
123
124     // Validación 6: Altura positiva
125     do {
126         printf("Altura (cm): ");
127         scanf("%f", &altura);
128         limpiarBuffer();
129     } while(altura <= 0);
130 }
131
132 // Calcula calorías usando fórmula Harris-Benedict modificada
133 float calcularCalorias(char sexo, int edad, float peso, float altura) {
134     const float C_HOMBRE = 5.03f; // Constante para hombres
135     const float C_MUJER = -161.0f; // Constante para mujeres
136     float constante = toupper(sexo) == 'H' ? C_HOMBRE : C_MUJER; // Asigna valor según sexo
137
138     return (10.0f * peso) +
139            (6.25f * altura) -
140            (5.0f * edad) +
141            constante; // 10*peso + 6.25*altura -5*edad + C
142 }
143
144 // Genera recomendaciones según estado de salud (comparación insensible)
145 void generarSugerencias(char* estadoSalud[]) {
146     // Normalizar a minúsculas para comparar cualquier formato
147     for(int i = 0; i < strlen(estadoSalud[1]); i++) {
148         estadoSalud[i] = tolower(estadoSalud[i]);
149     }
150 }
151
152 printf("\nSugerencia de actividad física:\n");

```

Compilador (1) Registro de Compilación Depuración Resultados Console Cerrar

- Errors: 0  
- Warnings: 0  
- Output Filename: F:\Copia\Orlando\UNADM\Asignaturas\Modulo 1\Fundamentos de programación\Unidad 3\DFPR\_U3\_A2\Programa de sugerencias de salud.exe  
- Output Size: 19 Kib  
- Compilation Time: 1.94s

Line: 7 Col: 56 Sel: 0 Lines: 160 Length: 5983 Insertar Done parsing in 0.016 seconds

F:\Copia\Orlando\UNADM\As x + v

```

----- Datos del estudiante -----
Nombre: Orlando Antonio Maturano Pizáña
Matrícula: ES251107915
Grupo: DS-DFPR-2501-B1-013

*** Registro de usuario ***
Nombre: Orlando Maturano
Estado de salud [bueno/malo/lesiones/afecciones]: bueno
Sexo (H/M): H
Edad: 22
Peso (Kg): 75
Altura (cm): 185

Hola, Orlando Maturano. Aquí están tus resultados:
Calorías diarias recomendadas: 1801.25

Sugerencia de actividad física:
- 30 minutos de caminata
- 15 minutos de ejercicios de fuerza
- 10 minutos de estiramientos

¿Desea calcular nuevamente para otro usuario? (S/N): S

*** Registro de usuario ***
Nombre: James Bond
Estado de salud [bueno/malo/lesiones/afecciones]: lesiones
Sexo (H/M): H
Edad: 53
Peso (Kg): 75
Altura (cm): 177

Hola, James Bond. Aquí están tus resultados:
Calorías diarias recomendadas: 1596.25

Sugerencia de actividad física:
- Consultar a un especialista para un plan personalizado

¿Desea calcular nuevamente para otro usuario? (S/N): S

```

```

P:\Copia\Orlando\UNADMAs <--> + - X
Edad: 38
Peso (Kg): 55
Altura (cm): 165
Hola, Lisa Rogan. Aquí están tus resultados:
Calorías diarias recomendadas: 1230.25
Sugerencia de actividad física:
- 30 minutos de caminata
- 15 minutos de ejercicios de fuerza
- 10 minutos de estiramientos
¿Desea calcular nuevamente para otro usuario? (S/N): S
== Registro de usuario ==
Nombre: Carlos Méndez
Estado de salud [bueno/malo/lesiones/afecciones]: malo
Sexo (H/M): H
Edad: 39
Peso (Kg): 58
Altura (cm): 173
Hola, Carlos Méndez. Aquí están tus resultados:
Calorías diarias recomendadas: 1471.25
Sugerencia de actividad física:
- Consultar a un especialista para un plan personalizado
¿Desea calcular nuevamente para otro usuario? (S/N): S
== Registro de usuario ==
Nombre: Melanie Sanchez
Estado de salud [bueno/malo/lesiones/afecciones]: afecciones
Sexo (H/M): M
Edad: 34
Peso (Kg): 60
Altura (cm): 155
Hola, Melanie Sanchez. Aquí están tus resultados:
Calorías diarias recomendadas: 1237.75
Sugerencia de actividad física:
- Consultar a un especialista para un plan personalizado
¿Desea calcular nuevamente para otro usuario? (S/N): N
Presione una tecla para continuar . . .

```

## CONCLUSIÓN

Este trabajo abordó el desarrollo de un programa en C para calcular la ingesta calórica diaria y sugerir actividades físicas, aplicando principios de programación modular, validación robusta y portabilidad multiplataforma. El objetivo principal que fué crear una herramienta accesible que integre nutrición y salud se logró mediante funciones especializadas (`registroUsuario`, `calcularCalorias`, `generarSugerencia`), garantizando claridad y reutilización del código. La implementación de validaciones estrictas (rangos numéricos, entradas no vacías) evitó errores comunes, mientras que la configuración UTF-8 aseguró compatibilidad con caracteres especiales en distintos sistemas operativos.

Los aprendizajes clave incluyen la importancia de la planificación modular para simplificar problemas complejos y la necesidad de anticipar entradas incorrectas mediante bucles de verificación. Los desafíos técnicos como el manejo seguro de buffer y la adaptación a APIs específicas de Windows, reforzaron la necesidad de escribir código defensivo y documentado.

En un contexto en donde las herramientas digitales de salud son prioritarias, este proyecto subraya la relevancia de la programación estructurada para desarrollar soluciones confiables y escalables. No solo resuelve un problema práctico como personalizar recomendaciones basadas en datos, sino que también sirve como modelo para futuras aplicaciones que integren algoritmos nutricionales con interfaces accesibles. Así, se demuestra que la tecnología, cuando se diseña con rigor técnico y enfoque centrado en el usuario, puede ser un aliado clave en la promoción de hábitos saludables.

## REFERENCIAS BIBLIOGRÁFICAS

- Embarcadero. (2024). Product documentation. C++ Builder. De:  
[https://docwiki.embarcadero.com/RADStudio/Athens/en/Main\\_Page](https://docwiki.embarcadero.com/RADStudio/Athens/en/Main_Page)
- GCC Documentation. (2025). 3.12 Options That Control Optimization. De:  
<https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html>
- Y: <https://gcc.gnu.org/onlinedocs/gcc.pdf>
- ISO. (2024). Information technology — Programming languages — C, De:  
<https://www.open-std.org/jtc1/sc22/wg14/www/docs/n3220.pdf>
- Javier Ceballos Sierra. (2019). C(C++) Curso de programación 5 ed. De:  
<https://es.scribd.com/doc/9838927/Ceballos-C-C-Curso-de-programacion-5Ed>
- Joyanes, L. (2005). Programación en C. Algoritmos, estructuras de datos y objetos. McGraw-Hill,  
De:  
<https://intprog.wordpress.com/wp-content/uploads/2013/08/programacion-en-c-metodologia-algoritmos-y-estructura-de-datos-editorial-mcgraw-hill1.pdf>
- Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language (2nd ed.). De:
- Inglés:  
[https://www.cimat.mx/ciencia\\_para\\_jovenes/bachillerato/libros/%5BKernighan-Ritchie%5DThe\\_C\\_Programming\\_Language.pdf](https://www.cimat.mx/ciencia_para_jovenes/bachillerato/libros/%5BKernighan-Ritchie%5DThe_C_Programming_Language.pdf)
- Español:  
[https://frrq.cvg.utn.edu.ar/pluginfile.php/13741/mod\\_resource/content/0/El-lenguaje-de-programacion-C-2-ed-kernighan-amp-ritchie.pdf](https://frrq.cvg.utn.edu.ar/pluginfile.php/13741/mod_resource/content/0/El-lenguaje-de-programacion-C-2-ed-kernighan-amp-ritchie.pdf)
- Learn Microsoft. (2023). Función SetConsoleOutputCP. De:  
<https://learn.microsoft.com/es-es/windows/console/setconsoleoutputcp>
- Norman, Donald A. (2013). The Design of Everyday Things. De:  
<https://dl.icdst.org/pdfs/files4/4bb8d08a9b309df7d86e62ec4056ceef.pdf>
- UnADM. (s.f.) Unidad 2. Introducción al lenguaje C. De:  
[https://dmd.unadmexico.mx/contenidos/DCEIT/Compartidas/FPR/U2/descargables/FPR\\_U2\\_Contenido.pdf](https://dmd.unadmexico.mx/contenidos/DCEIT/Compartidas/FPR/U2/descargables/FPR_U2_Contenido.pdf)
- Unicode Consortium. (2022). Unicode 15.0.0. 2 General structure. 2.5 Encoding Forms. De:  
<https://www.unicode.org/versions/Unicode15.0.0/>