Proyecto Final - Programación en C

José Gerardo González Suárez, Alizon Pamela Lagos Tlahuice

27 de noviembre de 2025

Índice general

Introducción

El proyecto surge de la necesidad de ayudarnos a realizar los cálculos de un proyecto de la materia de Ecuaciones Diferenciales. El proyecto en el que basamos nuestra solucion consiste en realizar composta bajo dos condiciones de temperatura. Se realiza en 6 vasos para cada ambiente, cada uno diferente cantidad de materia orgánica. Durante una cantidad de días que no se conoce se va registrar el comportamiento de la composta en cada vaso. Se registrará el día en el que se deja de observar materia orgánica en cada vaso para poder obtener la constante de descomposición de la materia orgánica en cada ambiente. Aquí es en donde nuestra solución entra en acción ya que se encarga de procesar los datos recabados y obtenern las constantes de descomposición para cada ambiente.

Objetivo General

Desarrollar un software funcional en C que permita dar solución a un problema planteado en una de las asginaturas que cursamos, reforzando las competencias de programación y aplicando conceptos del ámbito mecatrónico. Se busca que el estudiante haga uso el conocimiento adquirido, el pensamiento lógico y la creatividad para proponer una solución eficiente y efectiva.

Objetivos Específicos

Se busca que el alumno aplique los siguientes conceptos aprendidos durante el curso:

- Estructura secuencial
- Estructura condicional: if, switch
- Estructuras cíclicas: for, while, do...while
- Estructura de datos: arreglos
- Funciones
- Cadenas de caracteres
- Estructuras
- Uniones
- Manejo de archivos(lectura/escritura)

- Graficación básica(ASCII o con librería)

Alcance del proyecto

El proyecto consiste en el desarrollo de una aplicación de consola para calcular las constantes de descomposición k en un sistema de composta. Además, permitirá establecer condiciones iniciales distintas a las utilizadas físicamente, con el fin de simular el comportamiento del sistema bajo factores diferentes a los explorados. El programa será capaz de registrar los datos en un archivo .CSV, leerlos posteriormente y generar gráficas cuando sea requerido. Dichas gráficas se guardarán en formato .png junto con el archivo de datos en una carpeta de resultados creada por la aplicación. Para lograr esto, se implementarán funcionalidades de lectura y escritura de datos proporcionados por el usuario, manejo de archivos, operaciones matemáticas, graficación básica y control de las operaciones del sistema. Se contempla la portabilidad de la solución, permitiendo su ejecución en sistemas Windows, Mac y Linux. La solución se desarrollará en el lenguaje C, utilizando librerías estándar y algunas librerías externas para la graficación y manejo de errores.

No se contempla la creación de una aplicación móvil ni el uso de software avanzado de graficación. El alcance se limita a la funcionalidad básica, la interfaz de consola y los archivos generados por el programa. Por otro lado, la estructira del código está pensada de forma que partes de él puedan ser utilizadas en proyectos posteriores.

Desarrollo del proyecto

Iniciamos con la identifiacion del problema, posterior a eso se hizo el análisis de la posible solución. Era claro que se debían calcular las constantes k para cada vaso, así que primero se solicitan los valores con los que se va a calcular k. Posteriormente, las constantes se promedian y se escribe en pantalla el promedio para ambiente caliente y el promedio para ambiente frío. Después, se pide ingresar datos para la simulación, es en esta parte en donde puedes saber como se va a comportar la materia orgánica en más días de los que hiciste físicmamente. Los datos de la simulación son guardados en un archivo .CSV que a su vez se guarda en una carpeta de resultados que el programa crea en caso de que no exista o guarda en ella los datos si es que ya existe. La última cosa que hace el programa es leer los datos del archivo .CSV y poner en una gráfica en la que podemos ver tanto el comportamiento de la materia en el ambiente cálido como el frío y comparar.

Estos procesos que realiza el programa están divididos en un menú de 3 opciones. La opción 1 es para calcular k y simular, la opción 2 es para graficar y la opción 3 es para salir del programa. Cuando termina el proceso 1 o 2, el programa lo notifica con un mensaje de éxito o de error si lo hubiera y regres al menú para poder seleccionar otra opción. El programa está dividido en 4 archivos, de esta forma el código es más digerible y algunas funcionalidades de pueden reutiliar en otro proyecto. Los archivos usados son app.h, main.c, solve_equation.c y graficar.c. En la siguiente sección se explicara la función de cada archivo.

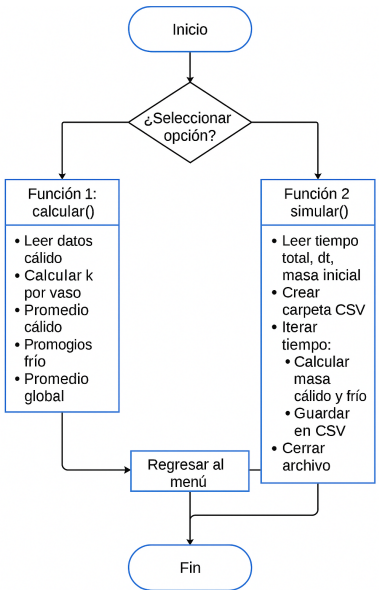

Figura 1: Lógica general del programa.

Fragmentos de código con comentarios

APP.h

```
        // -- Header File --
#ifndef APP_H //Si no esta definido APP_H
#define APP_H //Define APP_H

void calcular(); //Declaramos funcion para calcular k
void simular();  //Declaramos funcion que hara la simualcion
void graficar(const char*input_csv const char*output_png);
//Definimos funcion para graficar

#endif
```

Este es el primer archivo que se crea, es un archivo de encabezado (header file) en el que se implementa un include guard que evita que el archivo se incluya mas de una vez en la compilación. Además, se declaran las funciones que usará nuestro proyecto.

main.c

```
#include <stdio.h>
#include "app.h"
int main() {
    int opcion;
    do
    {
        printf("\n=== Aplicacion Decaimiento Exponencial ===\n");
        printf("1. Calcular y simular\n");
        printf("2. Graficar datos\n");
        printf("3. Salir\n");
        printf("Seleccione una opcion: ");
        scanf("%d", &opcion);

        switch(opcion) {
            case 1:
                calcular();
                                simular();
                break;
            case 2:
                graficar_csv("Resultados/simulacion.csv",
                                "Resultados/grafica.png");
                break;
            case 3:
                printf("Saliendo...\n");
                break;
            default:
                printf("Opcion invalida.\n");
        }
```

```
    } while(opcion != 3);

    return 0;
}
```

En este archivo se encuentra la función main que es el menú para elegir que queremos realizar. Se hace uso del ciclo do...while además de switch para las opciones del menú.

solve_equation.c

```
//Para Windows
#ifdef _WIN32
    #include <direct.h>   // Para _mkdir en Windows
#else
    #include <sys/stat.h> // Para mkdir en Linux/macOS
    #include <sys/types.h>
#endif

#include <errno.h>
#include <string.h>

//Para Linux y macOS
#include <sys/stat.h>
#include <sys/types.h>

#include <stdio.h>
#include <math.h>
#include "app.h"
}
```

Este es el comienzo del archivo solve_equation.c, en él se llaman las librerías necesarias para ejecutar el programa. Podemos notar que hay dos opciones de librerías, esto es para utilizar las librerías necesarias según el sistema operativo en el que se esté ejecutando el programa.

```
        int i;
    double M0, Mf, t_final;
        //Declaramos variables para Masa inicial, Masa final y
        //tiempo final
    double k_calido[6], k_frio[6];
        //Declaramos arreglos para giardar los valores de k en
        //cada ambiente
    double promedio_calido = 0.0, promedio_frio = 0.0,
        promedio_global;
    //Declaramos variables para guardar los promedios de k en
        //cada ambiente
        double tiempo_total, dt;
        //Declaramos variables para el tiempo total de simualcion
        //y el incremento del tiempo
```

```
    double M;
        //Declaramos variable para la masa total de simulacion
    FILE *archivo;
        //Declaramos puntero de archivo para manejar el .CSV
```

Declaramos variables de tipo double porque trabajaremos con números decimales pequeños de varios digitos

```
void calcular() {
// Calculo de k para ambiente calido
printf("=== Ambiente calido ===\n");
for (i = 0; i < 6; i++) {
        printf("\nVaso %d:\n", i + 1);
        printf("Ingrese masa inicial (M0): ");
        scanf("%lf", &M0);
        do{
                printf("Ingrese masa final (Mf): ");
                scanf("%lf", &Mf);
                if (Mf <= 0){
                        printf("Error: la masa final deber ser
                                mayor que 0\n");
                }
        } while(Mf <= 0);
        printf("Ingrese tiempo total (t): ");
        scanf("%lf", &t_final);

        // Formula analitica para k
        k_calido[i] = (1.0 / t_final) * log(Mf / M0);
        promedio_calido += k_calido[i];
        printf("k calculada: %lf\n", k_calido[i]);
}
promedio_calido /= 6.0;

// Calculo de k para ambiente frio
printf("\n=== Ambiente frio ===\n");
for (i = 0; i < 6; i++) {
        printf("\nVaso %d:\n", i + 1);
        printf("Ingrese masa inicial (M0): ");
        scanf("%lf", &M0);
        do{
                printf("Ingrese masa final (Mf): ");
                scanf("%lf", &Mf);
                //Commprobacion para que la masa final sea > 0
                if (Mf <= 0){
                        printf("Error: la masa final debe ser
                                mayor que 0.\n");
                }
        } while(Mf <= 0);
        printf("Ingrese tiempo total (t): ");
        scanf("%lf", &t_final);
```

```
43
44          k_frio[i] = (1.0 / t_final) * log(Mf / M0);
45          promedio_frio += k_frio[i];
46          printf("k calculada: %lf\n", k_frio[i]);
47  }
48  promedio_frio /= 6.0;
49
50  // Calculo del promedio global
51  promedio_global = (promedio_calido + promedio_frio) / 2.0;
52
53  printf("\nPromedio k calido: %lf\n", promedio_calido);
54  printf("Promedio k frio: %lf\n", promedio_frio);
55
56  }
```

En esta sección se encuentran los calculos de k para cada vaso así como el promedio de k para cada ambiente con un ciclo for. Además, se agregó una comprobación para quue la masa final ingresada sea mayor que 0 y evitar errores ya que no se puede calcular el logaritmo de cero o numeros negativos.

```
1   void simular() {
2   // Simulacion con formula analitca que se obtiene de resolver la
3   //ec dif usando promedio calido y frio
4       printf("\nIngrese tiempo total de simulacion: ");
5       scanf("%lf", &tiempo_total);
6       printf("Ingrese paso de tiempo (dt): ");
7       scanf("%lf", &dt);
8       printf("Ingrese masa inicial para simulacion: ");
9       scanf("%lf", &M0);
10
11  // Crear carpeta Resultados si no existe pra Windows
12  #ifdef _WIN32
13      if (_mkdir("Resultados") == -1 && errno != EEXIST) {
14          printf("Error al crear carpeta Resultados: %s\n",
15                  strerror(errno));
16          return;
17      }
18  #else
19      if (mkdir("Resultados", 0777) == -1 && errno != EEXIST) {
20          printf("Error al crear carpeta Resultados: %s\n",
21                  strerror(errno));
22          return;
23      }
24  #endif
25
26  // Crear carpeta Resultados si no existe para Linux y MacOS
27  #ifdef _WIN32
28      if (_mkdir("Resultados") == -1) {
29          if (errno != EEXIST) {
```

```
            printf("Error al crear carpeta Resultados: %s\n",
                        strerror(errno));
            return;
        }
    }
#else
    if (mkdir("Resultados", 0777) == -1) {
        if (errno != EEXIST) {
            printf("Error al crear carpeta Resultados: %s\n",
                        strerror(errno));
            return;
        }
    }
#endif

// Abrir archivo CSV
    archivo = fopen("Resultados/simulacion.csv", "w");
    if (!archivo) {
        printf("Error al abrir archivo simulacion.csv: %s\n",
                strerror(errno));
        return;
    }
    fprintf(archivo, "Tiempo,Masa_Calido,Masa_Frio\n");

    for (double tiempo = 0; tiempo <= tiempo_total;
        tiempo += dt) {

        // Metodo analitico
        double M_calido = M0 * exp(promedio_calido*tiempo);
        double M_frio = M0 * exp(promedio_frio*tiempo);

                fprintf(archivo, "%.2lf,%.6lf,%.6lf\n", tiempo,
                M_calido, M_frio);
        printf("Tiempo: %.2lf\tCalido: %.6lf\tFrio: %.6lf\n",
                tiempo, M_calido, M_frio);
    }

    fclose(archivo);
    printf("\nDatos guardados en simulacion.csv\n");
    return ;
}
```

En esta sección se realiza la simulación del comportamiento de la materia en ambos ambientes. Tambien se crea la carpeta de resultados, agregando una comprobación para evitar errores en caso de la carpeta ya exista. También, se agregaron dos opciones para crear la carpeta dependiendo del sistema operativo en el que se ejecute.

graficacion.c

```c
#include <cairo/cairo.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>

#define WIDTH 800
#define HEIGHT 600
#define MARGIN 50

void graficar_csv(const char *input_csv,
const char *output_png) {
    FILE *archivo = fopen(input_csv, "r");
    if (!archivo) {
        fprintf(stderr, "Error al abrir %s: %s\n", input_csv,
                strerror(errno));
        return;
    }

    char linea[256];
    int capacidad = 1000;
    double *tiempo = malloc(capacidad * sizeof(double));
    double *masa_calido = malloc(capacidad * sizeof(double));
    double *masa_frio = malloc(capacidad * sizeof(double));
    if (!tiempo || !masa_calido || !masa_frio) {
        fprintf(stderr, "Error de memoria\n");
        fclose(archivo);
        return;
    }

    int n = 0;
    fgets(linea, sizeof(linea), archivo); // Saltar encabezado
    while (fgets(linea, sizeof(linea), archivo)) {
        if (n >= capacidad) {
            capacidad *= 2;
            tiempo = realloc(tiempo, capacidad *
                        sizeof(double));
            masa_calido = realloc(masa\_calido, capacidad *
                        sizeof(double));
            masa_frio = realloc(masa\_frio, capacidad *
                        sizeof(double));
        }
        sscanf(linea, "%lf,%lf,%lf", &tiempo[n], &masa\_calido[n],
                &masa_frio[n]);
        n++;
    }
    fclose(archivo);

```

```
49      // Calcular maximos
50      double max_t = tiempo[n - 1];
51      double max_m = masa_calido[0];
52      for (int i = 0; i < n; i++) {
53          if (masa_calido[i] > max_m) max_m = masa_calido[i];
54          if (masa_frio[i] > max_m) max_m = masa_frio[i];
55      }
56      max_m *= 1.1; // margen extra
57
58      double scale_x = (WIDTH - 2 * MARGIN) / max_t;
59      double scale_y = (HEIGHT - 2 * MARGIN) / max_m;
60
61      cairo_surface_t *surface = cairo_image_surface_create(CAIRO_
62          FORMAT_ARGB32, WIDTH, HEIGHT);
63      cairo\_t *cr = cairo_create(surface);
64
65      if (cairo_surface_status(surface) != CAIRO_STATUS_SUCCESS) {
66          fprintf(stderr, "Error␣creando␣superficie␣Cairo\n");
67          free(tiempo); free(masa_calido); free(masa_frio);
68          return;
69      }
70
71      // Fondo blanco
72      cairo_set_source_rgb(cr, 1, 1, 1);
73      cairo_paint(cr);
74
75      // Ejes
76      cairo_set_source_rgb(cr, 0, 0, 0);
77      cairo_set_line_width(cr, 2);
78      cairo_move_to(cr, MARGIN, HEIGHT - MARGIN);
79      cairo_line_to(cr, WIDTH - MARGIN, HEIGHT - MARGIN);
80      cairo_move_to(cr, MARGIN, HEIGHT - MARGIN);
81      cairo_line_to(cr, MARGIN, MARGIN);
82      cairo_stroke(cr);
83
84      // Etiquetas
85      cairo_select_font_face(cr, "Sans", CAIRO_FONT_SLANT_NORMAL,
86          CAIRO_FONT_WEIGHT_NORMAL);
87      cairo_set_font_size(cr, 14);
88      cairo_move_to(cr, WIDTH / 2, HEIGHT - 10);
89      cairo_show_text(cr, "Tiempo");
90      cairo_move_to(cr, 10, HEIGHT / 2);
91      cairo_show_text(cr, "Masa");
92
93      // Ticks en X
94      int ticks = 10;
95      for (int i = 0; i <= ticks; i++) {
96          double t = max_t * i / ticks;
97          double x = MARGIN + t * scale_x;
98          cairo_move_to(cr, x, HEIGHT - MARGIN);
```

```
99          cairo_line_to(cr, x, HEIGHT - MARGIN + 5);
100         cairo_stroke(cr);
101         char etiqueta[32];
102         snprintf(etiqueta, sizeof(etiqueta), "%.1f", t);
103         cairo_move_to(cr, x - 10, HEIGHT - MARGIN + 20);
104         cairo_show_text(cr, etiqueta);
105     }
106
107     // Ticks en Y
108     for (int i = 0; i <= ticks; i++) {
109         double m = max_m * i / ticks;
110         double y = HEIGHT - MARGIN - m * scale_y;
111         cairo_move_to(cr, MARGIN, y);
112         cairo_line_to(cr, MARGIN - 5, y);
113         cairo_stroke(cr);
114         char etiqueta[32];
115         snprintf(etiqueta, sizeof(etiqueta), "%.1f", m);
116         cairo_move_to(cr, 10, y + 5);
117         cairo_show_text(cr, etiqueta);
118     }
119
120     // Curva calido (rojo)
121     cairo_set_source_rgb(cr, 1, 0, 0);
122     cairo_set_line_width(cr, 2);
123     cairo_move_to(cr, MARGIN, HEIGHT - MARGIN
124         - masa_calido[0] * scale_y);
125     for (int i = 1; i < n; i++) {
126         cairo_line_to(cr, MARGIN + tiempo[i] * scale_x,
127                 HEIGHT - MARGIN - masa_calido[i] * scale_y);
128     }
129     cairo_stroke(cr);
130
131     // Curva frio (azul)
132     cairo_set_source_rgb(cr, 0, 0, 1);
133     cairo_move_to(cr, MARGIN, HEIGHT - MARGIN - masa_frio[0] *
134         scale_y);
135     for (int i = 1; i < n; i++) {
136         cairo_line_to(cr, MARGIN + tiempo[i] * scale_x, HEIGHT
137                 - MARGIN - masa_frio[i] * scale_y);
138     }
139     cairo_stroke(cr);
140
141     // Leyenda
142     cairo_set_source_rgb(cr, 0, 0, 0);
143     cairo_move_to(cr, WIDTH - 150, MARGIN);
144     cairo_show_text(cr, "Calido␣(rojo)");
145     cairo_move_to(cr, WIDTH - 150, MARGIN + 20);
146     cairo_show_text(cr, "Frio␣(azul)");
147
148     cairo_surface_write_to_png(surface, output_png);
```

```
149     cairo_destroy(cr);
150     cairo_surface_destroy(surface);
151 
152     free(tiempo);
153     free(masa_calido);
154     free(masa_frio);
155 
156     printf("Grafica␣exportada␣como␣%s\n", output_png);
157 }
158 
159 void ejecutar_graficacion() {
160     graficar_csv("Resultados/simulacion.csv",
161         "Resultados/grafica.png");
162     return;
163 }
```

Este archivo se encarga de graficar los datos guardados en el archivo .CSV. Se hace uso de la librería CAIRO para graficar y convertir a imagen .png. Para lograr ejecutarlo en otro equipo se debe instalar esta libraría. Se incluyeron instrucciones para hacerlos en Windows, MAC y Linux en el README del proyecto.

Resultados obtenidos

Los resultados que obtuvimos fueron satisfactorios, el programa realiza los cálculos correctamente y genera las gráficas esperadas. Es un programa además útil porque ayudo a agilizar los cálculos y la gráfica pare el proyecto de diferenciales. En un aspecto más técnico, el proeyecto representó un buen reto para aplicar y consolidar los conocimientos adquiridos, así como adquirir nuevos conocimientos y habilidades en el manejo de librerías y en la creación de soluciones. Aprendimos a llevar un flujo de desarrollo ordenado con archivos separadas y funciones lo que nos permitirá reutilizar partes del código en proyectos futuros. Hacerlo de esta forma también se facilita la lectura y comprensión del código. Se logró crear un proeycto portable capaz de ser utilizad en distintos equipos y sistemas operativos.

Conclusiones

En conclusión, es evidente que la programación puede satisfacer diferentes disciplinas, usándose como una herramienta esencial en diversos ámbitos. En este caso particular, la aplicación del lenguaje de programación C resultó importante para agilizar los cálculos para el análisis de datos en el experimento de ecuaciones diferenciales.

Es importante destacar que en el desarrollo se utilizaron metodologías de programación estructurada, lo que nos facilitó la organización del código. Esto es una buena práctica que se debe aplicar para el desarrollo de nuestros proyectos.

Se debe mencionar que fue un gran reto el desarrollo de esta solución sobre todo en la graficación y la portabilidad del proyecto, fue necesario investigar documentación y

apoyarnos de Copilot como un revisor de código para lograrlo. Aunque suele ser controversial entre la comunidad de desarrolladores, consideramos que es una herramienta poderosa para ayudarnos con la detección y solución de errores, así como para aprender sobre nuevas tecnologías.

Reconocemos el valor y dedicacion de este trabajo como una base sobre la cual podemos construir nuevos conocimientos y abordar nuevos proyectos.

Bibliografía

Corcuera, P. (n.d.). Programación en Lenguaje C. https://personales.unican.es/corcuerp/progcomp/

Lógica de programación. (n.d.). https://montealegreluis.com/logica-programacion/docs/lenguaje-c.html

Ricci, T. I. A., & Cervantes, M. A. N. (n.d.). Entorno de Programación. https://repositorio-uapa.cuaed.unam.mx/repositorio/moodle/pluginfile.php/3077/mod_resource/content/1/UAPA-Entorno-Programacion-Lenguaje-C/index.html

Val, P. B., Ayres, I. E., & Pardo, A. (2018, julio). Capítulo 1. Estructura de un programa en C. https://www.it.uc3m.es/pbasanta/asng/course_notes/program_structure_es.html

(Corcuera, n.d.) Val et al. (2018) ("Lógica de programación", n.d.) Val et al. (2018) (Ricci & Cervantes, n.d.) Val et al. (2018) (Val et al., 2018) Val et al. (2018)