



Instituto Politécnico Nacional
Escuela Superior de Cómputo



REPORTE DE PRÁCTICA 1

Introducción al entorno de trabajo

Alumno: González Cárdenas Ángel Aquilez

Grupo: 1CM8

Realizada el 6 de septiembre del 2022

Objetivos

1. Observar y practicar la edición, compilación y ejecución de programas en el lenguaje de programación C.
2. Introducción al entorno de trabajo Dev C++® para la edición y compilación de programas.

Introducción

El lenguaje de programación C fue creado en durante los años 70, entre 1972 y 1973, como un lenguaje para la implementación del sistema operativo Unix, cuyo desarrollo se encontraba en paralelo. Creado por Dennis Ritchie en los laboratorios Bell de AT&T, su desarrollo se encuentra profundamente ligado a los lenguajes B (desarrollado por Ken Thompson) y BCPL (desarrollado por Martin Richards), así como al sistema operativo Multics. Derivado de las limitaciones y problemáticas que presentaba Multics, Unix y C cubren la necesidad de un sistema operativo que superen las limitaciones de sus antecesores, e introducen nuevas características que fueron posible gracias a la reimplementación de Unix en el propio lenguaje C. Gracias a la estandarización del lenguaje (conocido como *ANSI C*) y la posterior portabilidad conseguida a finales de 1970, es como C se posiciona como uno de los lenguajes dominantes para la creación de toda clase de proyectos en toda clase de arquitecturas. (Ritchie, 1993).

En 1983, el Instituto Nacional para la Estandarización (*ANSI*, por sus siglas en inglés) de los Estados Unidos, estableció un comité para conseguir una definición moderna y comprensiva del lenguaje C dando como resultado el estándar ANSI, o *ANSI C*, completado en 1988 (Kernighan, Ritchie, 1988).

Para convertir cualquier programa escrito en el lenguaje C, es necesario un *compilador* que traduzca las instrucciones escritas a instrucciones entendibles para la computadora, por lo cual utilizaremos el compilador GCC® para verificar que cualquiera de los programas a realizar funcione adecuadamente.

GCC (*GNU Compiler Collection*, por sus siglas en inglés), es un compilador producido por el Proyecto GNU, presentado por primera vez por Richard Stallman en 1987 con su versión 1.0 que sólo admitía lenguaje C.

Años más tarde, se agregaría compatibilidad con más lenguajes de programación y se extendería a C++.

Desarrollo

La práctica se dividió en tres partes, cada una con un código correspondiente para compilar, ejecutar y realizar observaciones sobre su comportamiento durante la ejecución.

Parte 1

En principio, tenemos el siguiente código:

```
#include <stdio.h>
int main(){
    printf("ViVa Mexico!\n");
    printf("ViVa Mexico!\t");
    printf("Prueba");
    printf("\n");
    printf("\r ViVa Mexico! ");
    printf("Prueba");
    printf("\n");
    printf("ViVa Mexico! \r");
    printf("Prueba");
    printf("\n");
    printf("ViVa Mexico! \a");
    printf("Prueba");
    printf("\n");
    printf("ViVa Mexico! \\");
    printf("\n");
    printf("\\ViVa Mexico! \\n");
    printf("\'ViVa Mexico!'\n");
    printf("\"ViVa Mexico!\"\n");
    return 0;
}
```

El cual produce las siguientes salidas al ejecutarlo:

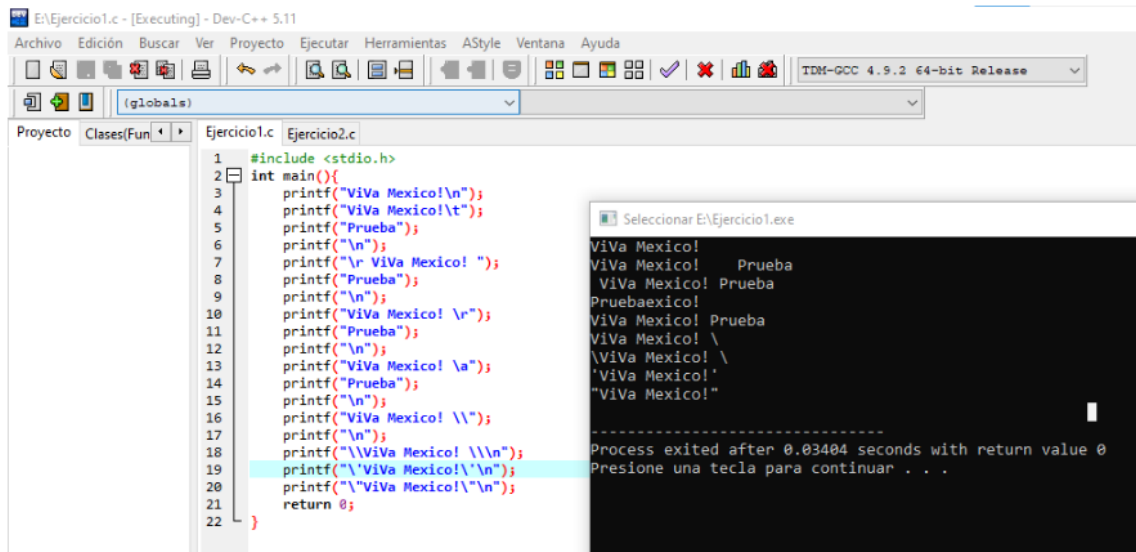


Figura 1: Ejercicio1.c produjo Ejercicio1.exe para su ejecución

Al ejecutar el programa, el texto generado presenta el siguiente comportamiento: En la instrucción de la línea 3, después del mensaje, se salta a la siguiente línea, por lo cual el mensaje de la línea 4 se imprime debajo de él. En la línea 4, al colocar un texto **Prueba**, se aprecia que después del mensaje, se imprime una tabulación, por lo cual el texto **Prueba** se imprime con los espacios equivalentes a la tabulación.

En la instrucción de la línea 7, la opción `\r` regresa el cursor de la impresión al principio de la línea, y como se tienen espacios en el mensaje, se aprecia el espacio sin interactuar de otra forma con el texto **Prueba**, sin embargo, en la línea 10, como el `\r` se encuentra al final del mensaje, el texto **Prueba** se imprime al principio de la línea, reemplazando el mensaje sobre lo que previamente se imprimió.

En la línea 13, se encuentra el operador `\a` al final del texto, por lo cual se apreció un sonido de alerta. Este operador no interactúa con los siguientes mensajes por lo cual el mensaje **Prueba** de la línea 14 se imprimió inmediatamente después sin alteraciones. Finalmente, las interacciones del carácter `\` y las comillas simples y dobles con la impresión de mensajes se aprecia en las siguientes líneas.

En la línea 16, se imprime solo un carácter `\`, ya que el operador `\\` lo imprimió. De igual forma, en la línea 18, el operador se encuentra al principio y al final del mensaje, imprimiendo el mismo carácter.

De forma similar, como la impresión de mensajes se controla mediante comillas simples o dobles, para la impresión de estas, es necesario colocar los

operadores correspondientes \' y \", volviéndose evidente su comportamiento en la impresión de mensajes.

Parte 2

Para la segunda parte, se tiene el siguiente código:

```
#include <stdlib.h>
#include <stdio.h>
int main(){
    int bol;
    float cal1, cal2, cal3, cal4, cal5, cal6, Prom;
    printf("Introduce el numero de Boleta: ");
    scanf("%d", &bol);
    printf("Introduce la calificacion de Matematicas: ");
    scanf("%f", &cal1);
    printf("Introduce la calificacion Espanol: ");
    scanf("%f", &cal2);
    printf("Introduce la calificacion de Biologia: ");
    scanf("%f", &cal3);
    printf("Introduce la calificacion de Historia: ");
    scanf("%f", &cal4);
    printf("Introduce la calificacion de taller de LR: ");
    scanf("%f", &cal5);
    printf("Introduce la calificacion de Geografia: ");
    scanf("%f", &cal6);
    Prom = (cal1 + cal2 + cal3 + cal4 + cal5) / 6;
    printf("Numero de Boleta: %d\n", bol);
    printf("El promedio es: %f.\n", Prom);
    system("pause");
    return 0;
}
```

Como este programa nos solicita la introducción de información para su ejecución, se introdujo lo siguiente: 10, 8, 9, 7, 8, 9.

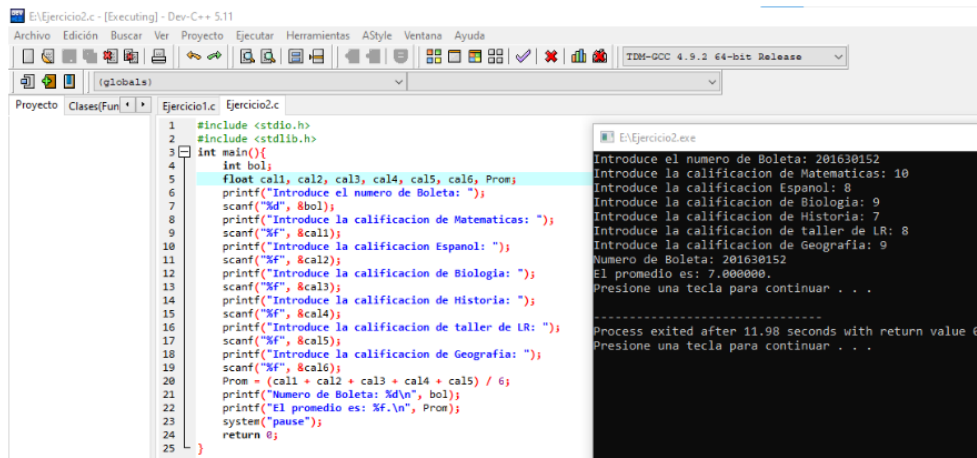


Figura 2: Programa que nos solicita calificaciones para calcular el promedio

Para este segundo programa, se analizó su comportamiento al eliminar la librería `stdlib.h`, provocando la siguiente salida al ejecutarse en una terminal de un sistema operativo tipo *GNU/Linux*:

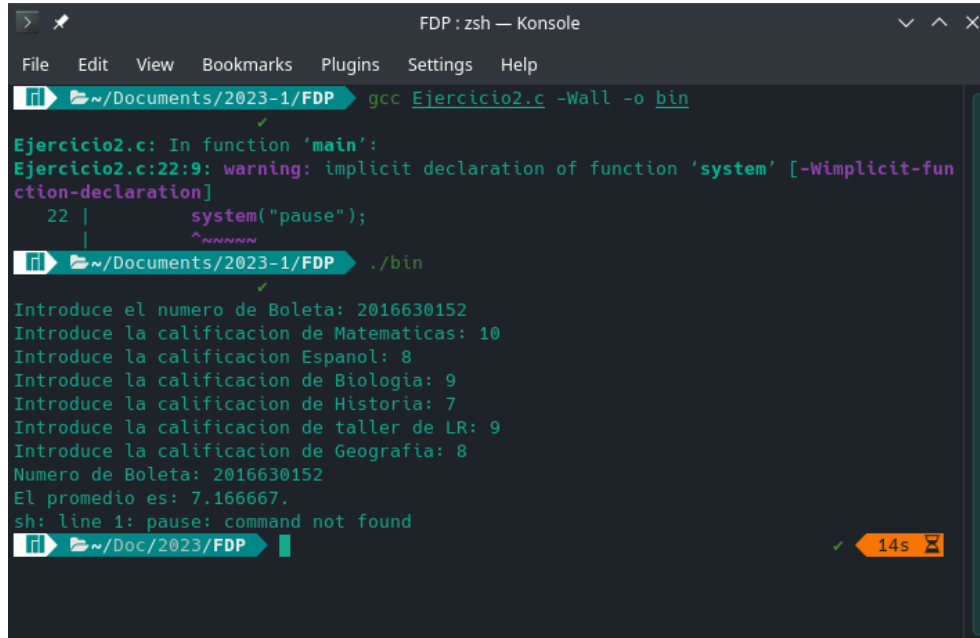


Figura 3: La instrucción no puede ser ejecutada al ser una arquitectura diferente

La ejecución del programa genera el mensaje

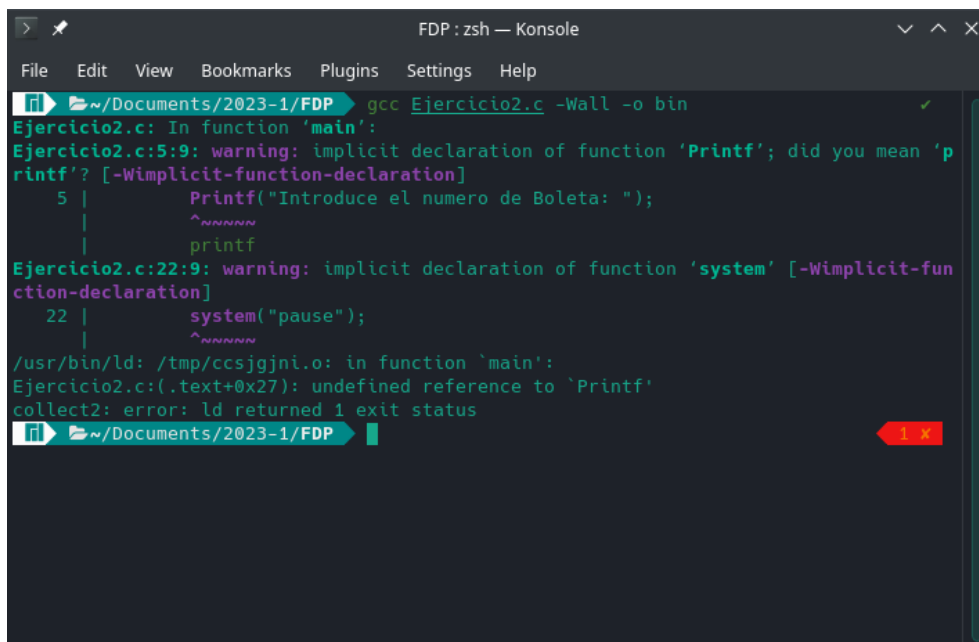
```
sh: line 1: pause: command not found
```

debido a que la instrucción `pause` es exclusiva de sistemas operativos tipo Windows®.

`system` es una función que se utiliza para ejecutar comandos en la terminal del sistema, por lo cual debe considerarse las diferentes instrucciones correspondientes a cada sistema operativo.

Nótese que al eliminar la librería `stdlib.h`, la compilación no se detiene, pero la ejecución sí.

Por último, al colocar la instrucción `Printf` en lugar de `printf` en alguna instrucción, se obtuvo la siguiente salida:



```
FDP : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
~ / Documents / 2023-1 / FDP gcc Ejercicio2.c -Wall -o bin ✓
Ejercicio2.c: In function 'main':
Ejercicio2.c:5:9: warning: implicit declaration of function 'Printf'; did you mean 'printf'? [-Wimplicit-function-declaration]
   5 |         Printf("Introduce el numero de Boleta: ");
     |         ^~~~~~
     |         printf
Ejercicio2.c:22:9: warning: implicit declaration of function 'system' [-Wimplicit-function-declaration]
   22 |         system("pause");
     |         ^~~~~~
/usr/bin/ld: /tmp/ccsjgjni.o: in function `main':
Ejercicio2.c:(.text+0x27): undefined reference to `Printf'
collect2: error: ld returned 1 exit status
~ / Documents / 2023-1 / FDP
```

Figura 4: La compilación no es posible

Esto se debe a que la referencia a la instrucción `printf` no es correcta, ya que esta se encuentra en la librería `stdio.h`, por lo cual es obligatorio que se encuentre correctamente escrita para ser utilizada.

Parte 3

Finalmente, realizaremos un programa que genere la serie de Fibonacci, donde cada número de la serie es producto de los dos que le preceden. El código generado es el siguiente:

```
#include <stdio.h>
int main() {
    int fn, f1, f2, r;
    f1 = 0;
    f2 = 1;
    printf("Fibonacci: ");
    scanf("%d", &fn);
    for (int i=0; i<=fn; i++){
        printf("%d, ", f1);
        r = f1 + f2;
        f1 = f2;
        f2 = r;
    }
    printf("\n");
}
```

El código nos solicita el número de ciclos hasta los cuales la serie de Fibonacci se generará, su comportamiento se aprecia en la salida que se genera:

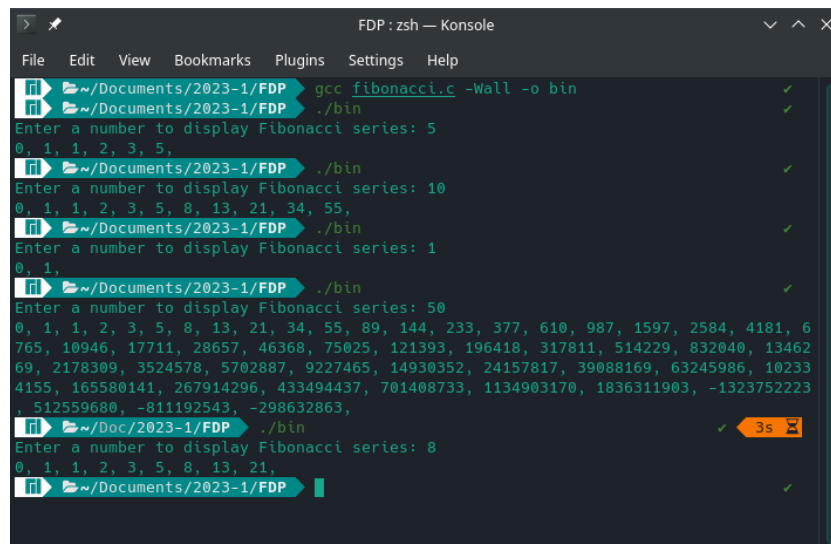


Figura 5: Diferentes salidas de la serie de Fibonacci

Conclusiones

Tras finalizar la práctica, podemos concluir lo siguiente:

1. Se lograron los objetivos de la práctica satisfactoriamente.
2. El comportamiento de la ejecución de los programas compilados en diferentes sistemas operativos puede producir incompatibilidades que generan la detención de la ejecución de manera súbita.
3. Las instrucciones provenientes de librerías deben escribirse correctamente para ser utilizadas.

Bibliografía

1. Kernighan, B. W., & Ritchie, D. M. (1988). C Programming Language (2a ed.). Prentice Hall.
2. Stallman, Richard (s/f). About the GNU project - GNU project - free software foundation. Gnu.org. Recuperado el 11 de septiembre de 2022, de <https://www.gnu.org/gnu/thegnuproject.html>
3. Wikipedia contributors. (2022, julio 24). GNU Compiler Collection. Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=GNU_Compiler_Collection&oldid=1100096189