**18/08/2021**

**Taller #1.**

**Comprensión de código, lenguajes compilados, lenguaje C, compilador Gcc**

**Nombre: Sebastián Gómez Valencia**

1. **Preguntas generales**

Compile como le fue presentado en clase el programa *Taller1.c.* Luego conteste las siguientes preguntas:

* 1. ¿Para qué sirve la instrucción **#include**?

**R/** La instrucción de **#include** sirve para incluir los ficheros o archivos los cuales serán utilizados en el programa, a la vez, estos archivos o ficheros pueden servir como librerías, desde las cuales se importan las funciones a utilizar. En este caso en específico, incluye la librería que permite la entrada y salida de datos.

* 1. ¿Cuál es el objetivo principal de ese programa?

**R/** El objetivo principal del programa es resolver dados dos números (ingresados por el usuario), las cuatro operaciones básicas de la matemática (suma, resta, multiplicación y división). Cumpliendo la función de una calculadora básica de operaciones básicas.

* 1. ¿Cuántas variables tiene en total?

**R/** 5 variables en total. (*result, a,b,c,d*)

* 1. ¿Cuáles de esas variables son variables locales y cuáles son variables globales? ¿Por qué?

**R/** En el programa *taller1.c* no se presentan variables globales, dado que no hay variables por fuera de las funciones definidas por el usuario, se presentan variables locales, como es el caso de la variable *result* en la *funcionCuatro* #4, las variables *a,b,c* y d son variables locales de la función *main*(). Se menciona que son variables locales pues únicamente son reconocidas y en “rango” dentro del bloque de código delimitado por las llaves de las funciones.

* 1. ¿Cuántas funciones definidas por el usuario tiene ese programa?

**R/** Hay 4 funciones definidas por el usuario (*funcionUno, funcionDos, funcionTres, funcionCuatro*).

* 1. ¿Cuál es el nombre de cada una de esas funciones?

**R/** Las funciones son: 1) int funcionUno 2) int funcionDos 3) int funcionTres 4) float funcionCuatro.

* 1. ¿Para qué cree usted que se convierten los números enteros en flotantes en la función cuatro? Explique

**R/** En la función cuatro donde se realiza la división de dos números, se pasan los datos a tipo float para brindar una mayor exactitud al momento de realizar la operación y a la vez, para tener concordancia con el tipo de dato en el cual va a ser almacenado el resultado, siendo este de tipo float, generando la concordancia en los tipos de datos.

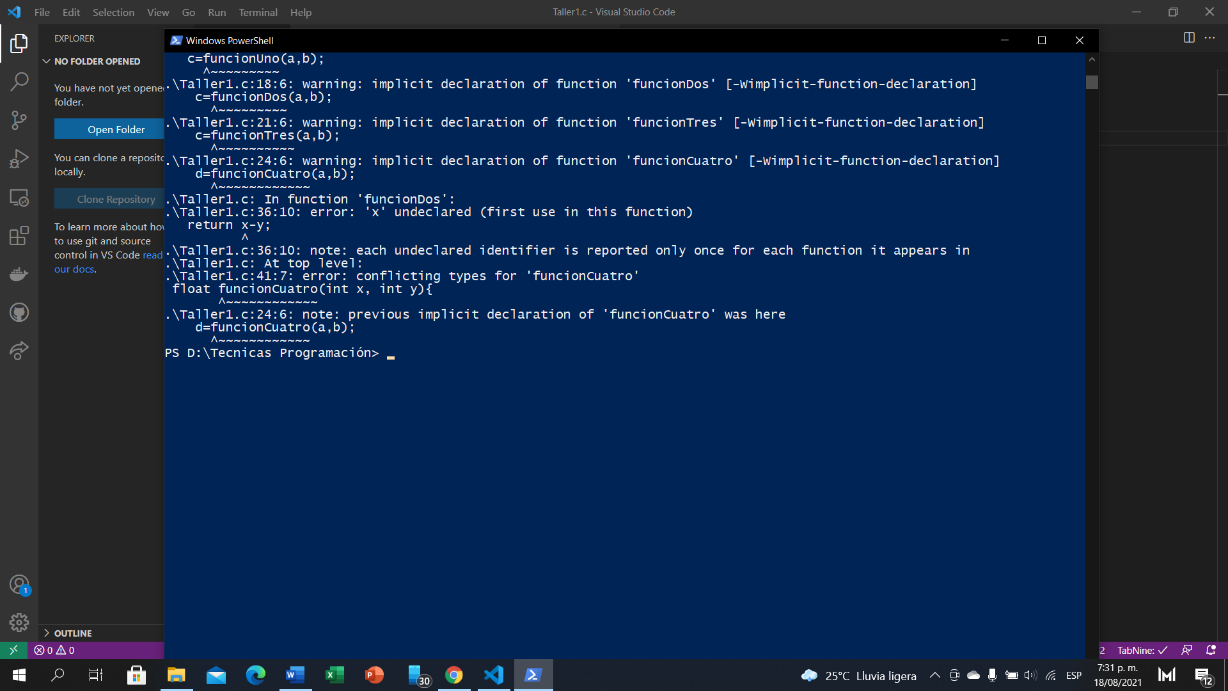
* 1. ¿Explique qué es la función main y por qué es importante?

**R/** La función main() es fundamental e indispensable en todos los programas de C, dado que este es el que marca el punto de inicio o la estructura de algoritmos que serán ejecutados, esto genera la estructura secuencial a seguir tras la ejecución.

1. **Modificar y contestar**
   1. Cambie la función **main** de posición. Córtela y péguela luego de la declaración de las librerías. Compile y ejecute nuevamente el programa. ¿Qué resultado obtuvo? ¿Por qué cree usted que se obtuvo ese resultado?

**R/** Como resultado generó errores de nombre *“warning implicit declaration of function…”* esto se genero para cada una de las funciones, este error opino se dio por usar una función la cual no había sido vista o considerada por el programa.

* 1. Modifique la función *funcionDos.* Borre la declaración de la variable *x*. Compile y ejecute nuevamente el programa. ¿Qué resultado obtuvo? ¿Por qué cree usted que se obtuvo ese resultado?

**R/** Tras eliminar la declaración de la variable *x* se generan dos errores principales, en la función dos se dio el error sobre *undeclared* variable por lo cual no se pueden retornar valores no declarados o el resultado de la operación entre una variable inexistente o no declarada y una declarada. Asimismo, se presenta los errores mencionados en el punto 2.1, con respecto a los errores de la declaración implícita de funciones, también probé restableciendo el orden de la función main(), en donde para arrojar el resultado de la operación de resta, es decir segunda función se generó el error donde se dieron más valores de los esperados en la ejecución o llamado a la función. 

1. **Mejorar**

**Nota**: Para desarrollar este punto ubique la función **main** nuevamente al final del archivo y declare nuevamente la variable *x* en la *funciónDos* para que el programa compile y se ejecute sin errores.

El programa *taller1.c* no se encuentra correctamente documentado. El nombre del programa no se relaciona con la funcionalidad principal, los nombres de las funciones no tienen relación con su objetivo, los nombres de las variables tampoco tienen relación con los elementos que almacenan, ni tiene documentación en las funciones que facilite la comprensión del código.

Modifique el programa actual y:

* 1. Renombre el archivo fuente de acuerdo con el objetivo principal del programa que fue especificado en el numeral 1.2.
  2. Renombre cada función del programa, de acuerdo con la ocupación principal que realiza cada función.
  3. Renombre cada variable para que el nombre tenga relación con su utilidad dentro del programa.
  4. Modifique los printf después de invocar a cada función para que indique a qué operación corresponde cada resultado.
  5. Pruebe que el nuevo programa compile y se ejecute correctamente y guárdelo con el nombre que usted crea que deba tener, más las iniciales de su nombre completo. Por ejemplo, si el programa sirviera para obtener las tablas de multiplicar, el programa se llamaría: tablasMultiplicarLGNA.c . La parte en verde es el nombre del programa, la parte en roja corresponde a sus iniciales, que en mi caso es LGNA. (por si las moscas, no le tiene que quedar el nombre con colores)

Para la documentación del programa tenga en cuenta el siguiente estándar de nombramiento:

Use nombres en lo posibles cortos y con un significado claro. La primera letra debe ser minúscula, si son más de 2 palabras se pone la primera letra de la primera palabra en minúscula y las iniciales de las demás palabras en mayúsculas. Además, para las funciones, el nombre debe comenzar por un verbo en infinitivo. Esta notación se llama ***lowerCamelCase.***

***Ejemplos de funciones:*** *quitarBoton, calcularCredito, sumarNumeros*

***Ejemplos de variables****: sumaGeneral, promedio, nroHabitantes.*

1. **Crear**

Para esta parte del taller debe consultar:

Como leer y escribir números enteros y como escribir cadenas en C.

Uso de condicionales en C.

Cómo calcular el módulo entre dos números para saber si una división es exacta.

* 1. Adicione al programa que modificó en el numeral 3, un procedimiento que le ingresa por parámetro un número entero de nueve (9) dígitos y debe indicar por pantalla si este número es o no palíndromo, es decir, que el número se puede leer igual de izquierda a derecha que de derecha a izquierda. (pista: se podría ayudar de uso de la división y el módulo)

Restricciones:

* No puede hacer uso de arreglos (cadenas, vectores, listas, matrices, etc.)
* Si desea, puede hacer uso de ciclos, aunque no es obligatorio.
* En la función main debe pedir el número de 9 dígitos al usuario, y hacer el llamado al procedimiento. El procedimiento debe estar creado afuera del main.
* Recuerde el uso de lowerCamelCase y la documentación de su código.
  1. Adicione al programa que modificó anteriormente, una función que calcule si un año es o no bisiesto y retorne 1 si el año es bisiesto y 0 si el año no es bisiesto.
  2. En la función main adicione: una línea que solicite el ingreso del año a analizar y la invocación a la función que calcula si el año es bisiesto.
  3. Si la función que calcula si el año es bisiesto retorna un 1, entonces imprima en pantalla “El año AAAA es bisiesto” y “mi nombre es XXXXX”. Donde, AAAA corresponde al año ingresado por el usuario y XXXXX corresponderá a su nombre. Si la función que calcula si el año es bisiesto retorna 0, entonces imprima en pantalla: “El año AAAA no es bisiesto y tengo YY hermano(s)”. Donde YY corresponde a la cantidad de hermanos que tenga. Si no tiene hermanos ponga 0. (También por si las moscas, tenga en cuenta que su nombre y su cantidad de hermanos es un valor arbitrario que usted digita, no necesita hacer cálculos ni nada parecido)

La lógica de este punto la podrá incluir en la función **main**, o en otra función creada por usted que sea llamada en el método **main**.

Tenga en cuenta que un año es bisiesto en dos casos posibles:

* Es divisible por 4 y no divisible por 100.
* Es divisible por 400. Por ejemplo, los años 1800 y 1900 no fueron bisiestos, pero sí lo fueron el 2000 y el 2012.

Un ejemplo del resultado de la ejecución del programa sería:



**ADJUNTO ARCHIVO .C RESPECTIVO CON LOS CAMBIOS DEL PUNTO 3 Y 4.**