|  |  |  |
| --- | --- | --- |
| escudofi_color_m2008_jpg | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorios de docencia |

Laboratorio de computación . Salas A Y B

|  |  |
| --- | --- |
| Profesor: | Claudia Rodríguez Espinoza |
| Asignatura: | Fundamentos de programación |
| Grupo: | 1104 |
| No de Práctica(s): | N° 11 |
| Integrante(s): | Romero Bernal Rocío Fabiola |
|  |  |
|  |  |
|  |  |
| No. de Equipo de cómputo empleado: | N° 42 |
| Semestre: | 2019-1 |
| Fecha de entrega: | 29/10/18 |
| Observaciones: |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

OBJETIVO: Reconocer la importancia y utilidad de los arreglos, en la elaboración de programas que resuelvan problemas que requieran agrupar datos del mismo tipo, así como trabajar con arreglos tanto unidimensionales como multidimensionales.

Anteriormente habíamos visto como mediante ciertas funciones como for, while o do while, podíamos elaborar ciclos repetitivos (limitados mediante parámetros preestablecidos) que nos permitieran obtener una cierta cantidad de datos, con la finalidad de usarlos en un determinado proceso operativo la cantidad de veces que fuera necesario. Sin embargo la complejidad aumentaba a medida que se introducían más procesos y valores a las funciones y en consecuencia incrementaba la pérdida del control de las funciones dentro del código, además de que los resultados obtenidos eran poco satisfactorios y muchas veces ni siquiera se acercaban a los resultados esperados.

De aquí se deriva la importancia de los arreglos dentro de la programación, conjuntos ordenados de variables que nos permiten guardar diferentes valores bajo el nombre de una misma variable, asignándoles a cada valor, una posición dentro del arreglo (una dirección de memoria) para identificarlo y usarlo posteriormente con un fin determinado. Invariablemente, los arreglos van a necesitar que se les asigne un tamaño y una dimensión para así establecer la cantidad de valores que podrán contener.

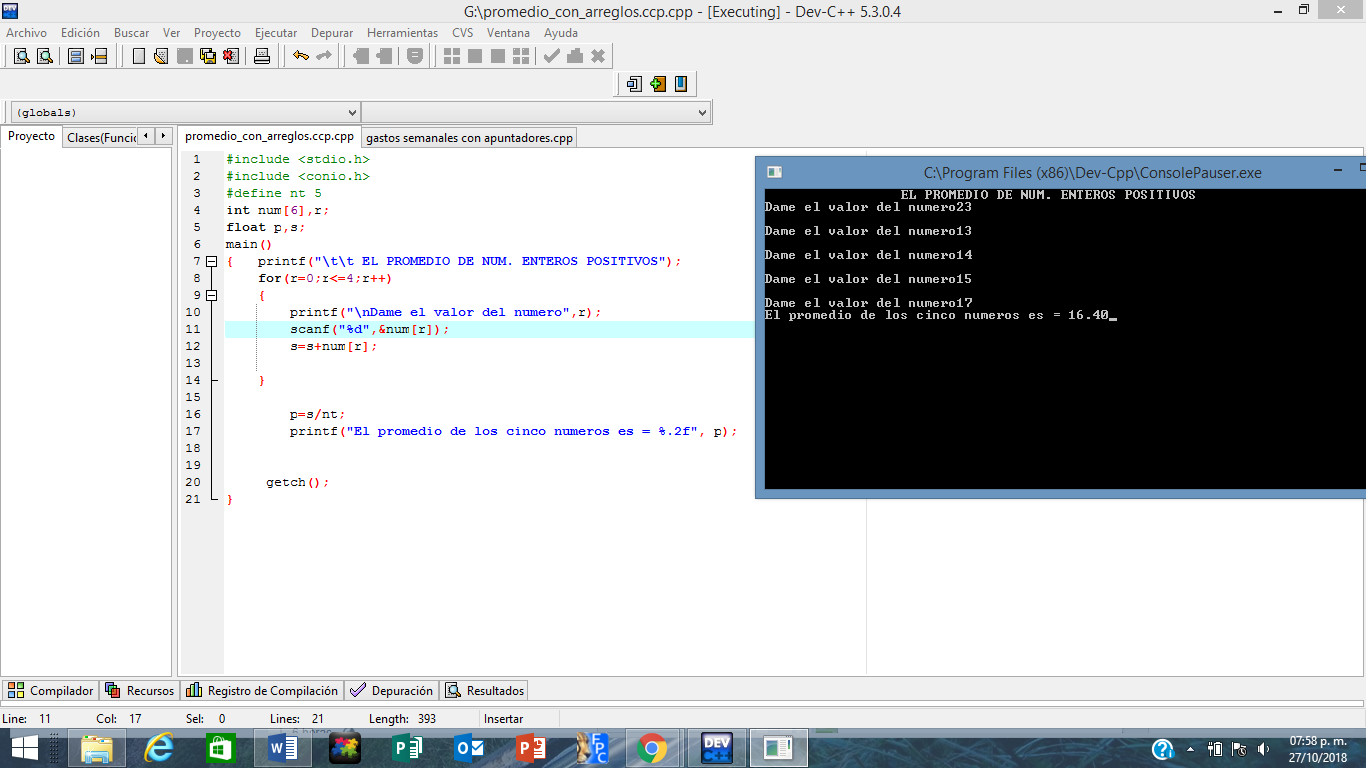
¿Pero que sucede cuando tenemos valores que no son numéricos?, pues las cadenas de caracteres que son consideradas arrays de caracteres (arreglos), guardan valores que no son del tipo numérico, bajo el nombre de una misma variable cuyo último elemento es nulo. De esta manera si nuestra cadena tiene un tamaño igual a 8 unidades, podrá contener 8 elementos (contando desde la posición 0) menos el valor nulo que se toma como el último elemento.

Ciertamente a partir de la reducción de la cantidad de variables que vamos a utilizar, podemos tener una mejor deducción y control de los resultados que deseamos obtener. No obstante, aun es posible facilitar el uso de los arreglos, mediante el apoyo de los apuntadores, los cuales son variables que nos permiten accesar directamente a las direcciones de memoria es decir, a la información que se encuentra almacenada en la variable a la que se está apuntando.

Una vez entendido la utilidad y función de los arreglos y apuntadores se presentaran los programas que fueron realizados durante la sesión de laboratorio que buscan representar visualmente la explicación anteriormente presentada:

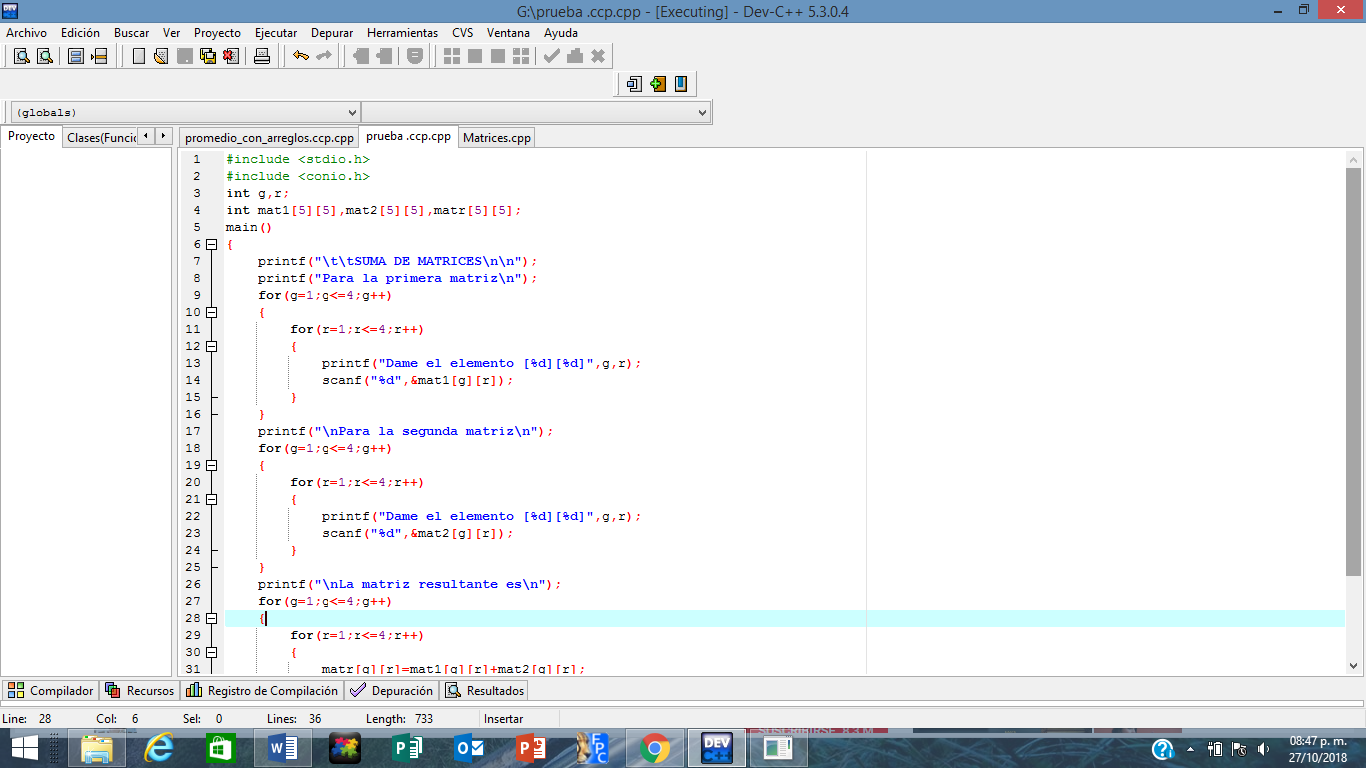
ACTIVIDAD 1: El promedio de 5 números

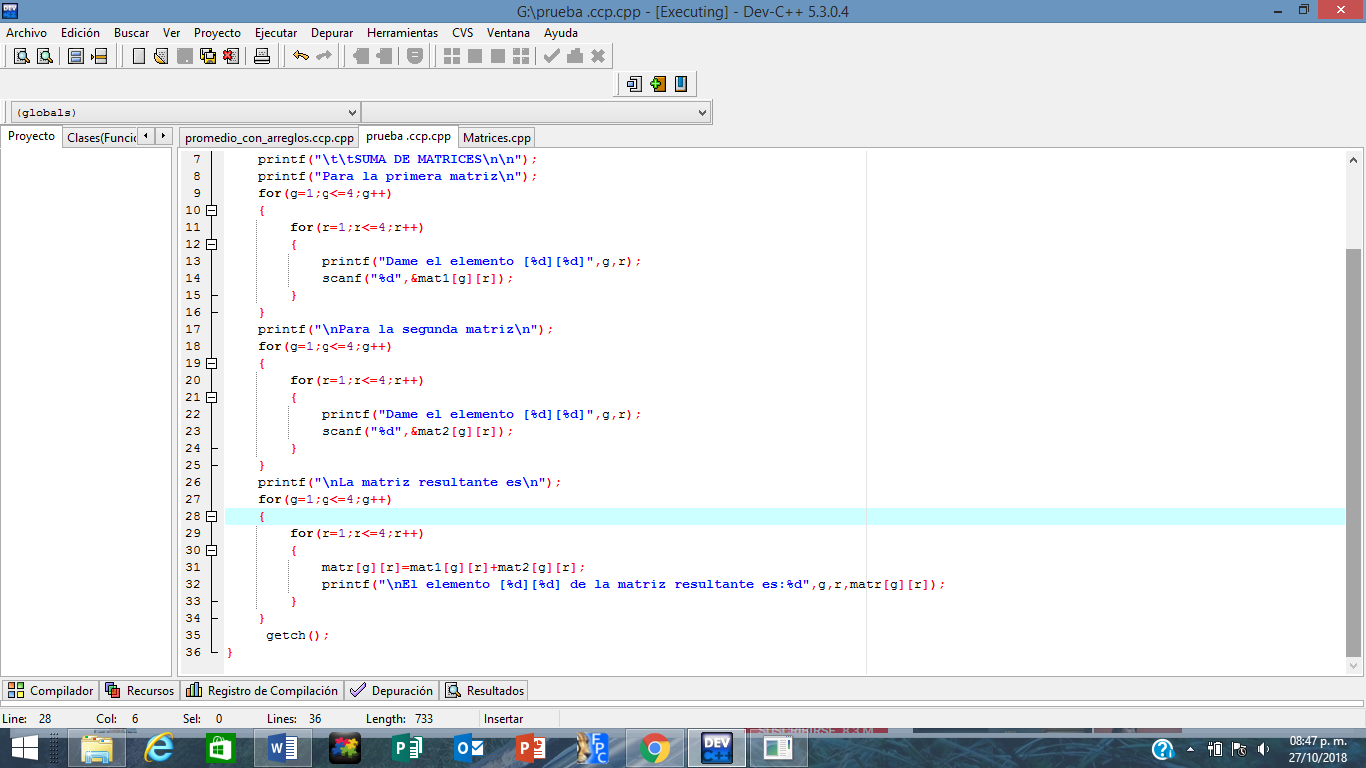
El primer paso para la elaboración de un programa con arreglos es declarar la variable y el arreglo que contendrá a la variable, bajo el tipo int (variable entera) y posteriormente definir el tamaño y la dimensión que se le vaya a asignar al arreglo. En esta ocasión fue designado como un arreglo unidimensional con un tamaño para almacenar 7 valores bajo el nombre de la variable r. Posteriormente se colocó una función for que nos permitiera pedir el valor correspondiente a una posición de la variable y ya que solo se necesito obtener el promedio de 5 números, se establecieron los parámetros dentro de la función para pedir únicamente el valor de 5 números enteros. Luego el programa realizo la suma de dichos números que se guardaban en la variable flotante s cuya magnitud se dividió entre 5(cantidad de valores designados) y el resultado se imprimió bajo el nombre de la variable c también del tipo flotante. Gracias a esto se pudo reducir la cantidad de variables que fueron usadas y la estructura del programa se volvió más sencilla

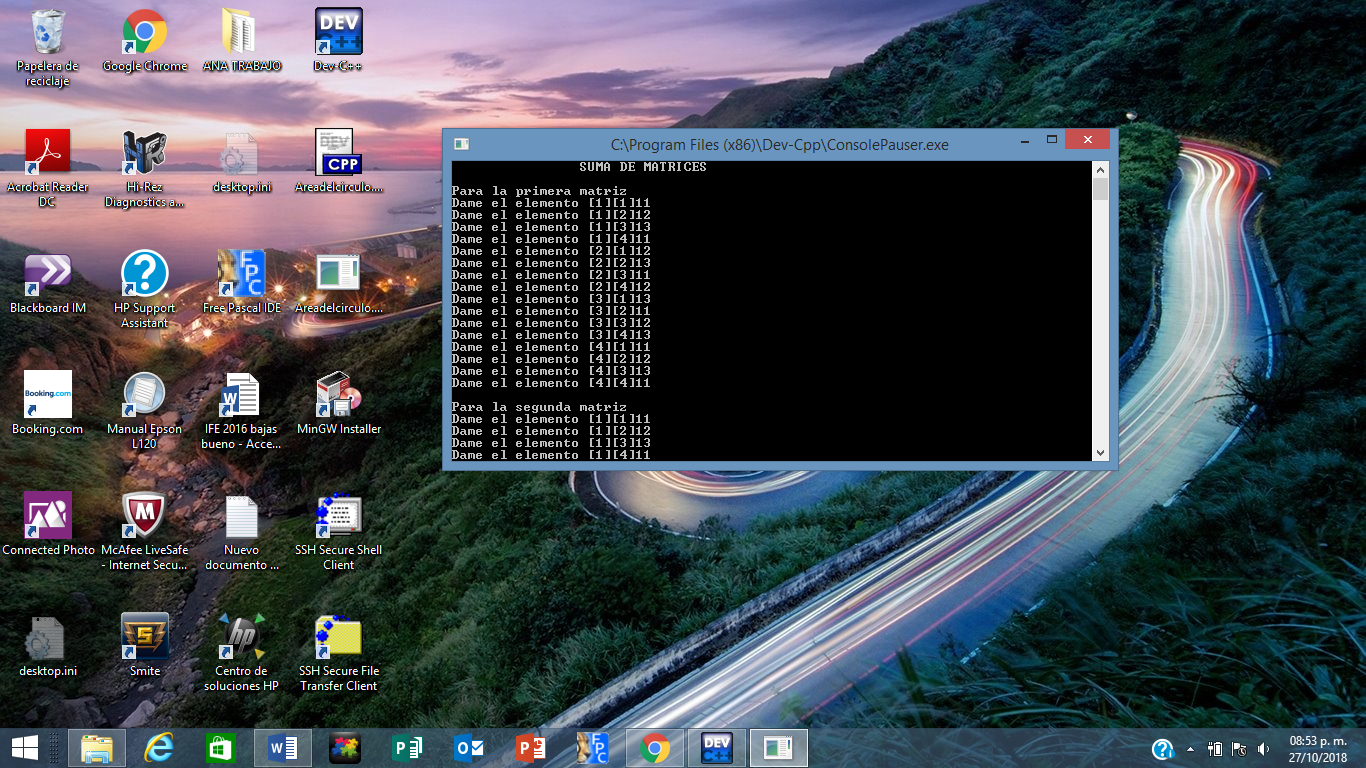


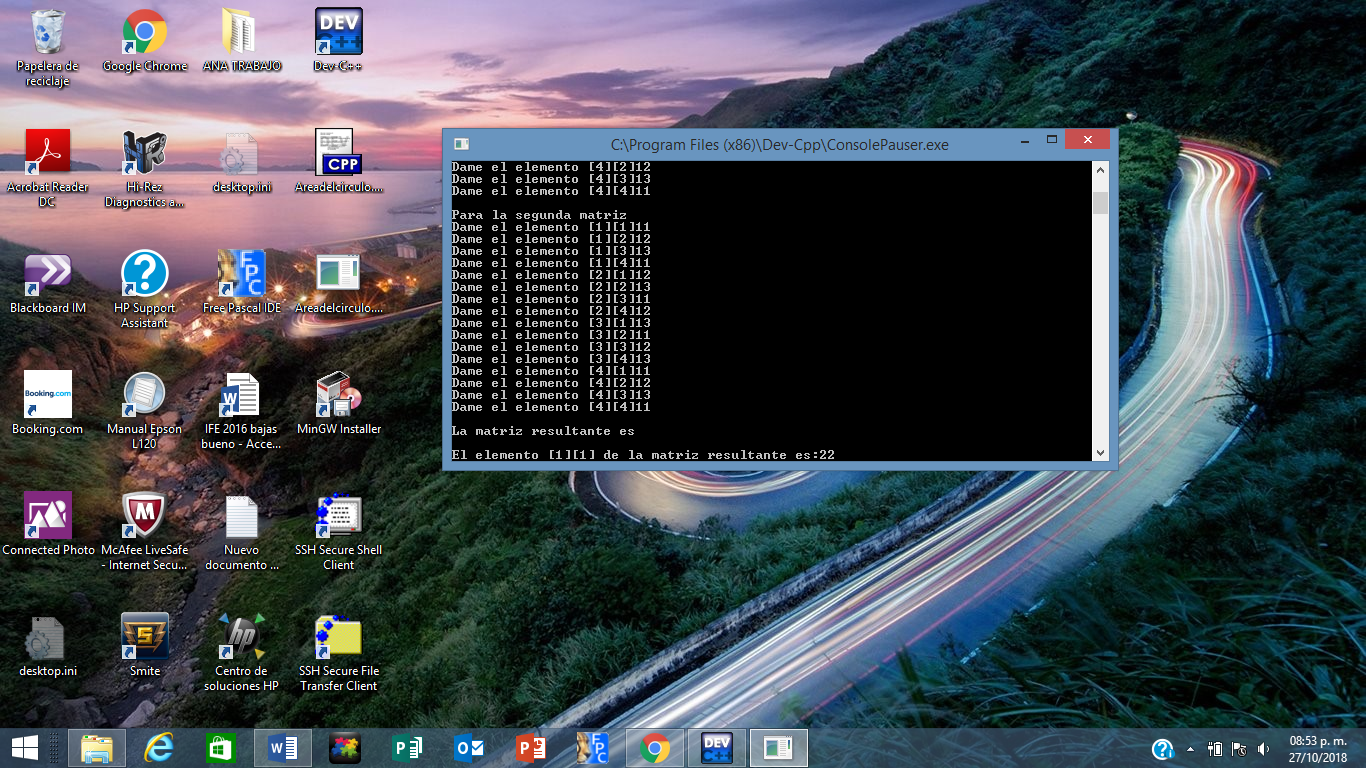
ACTIVIDAD 2: “Suma de matrices 4x4”

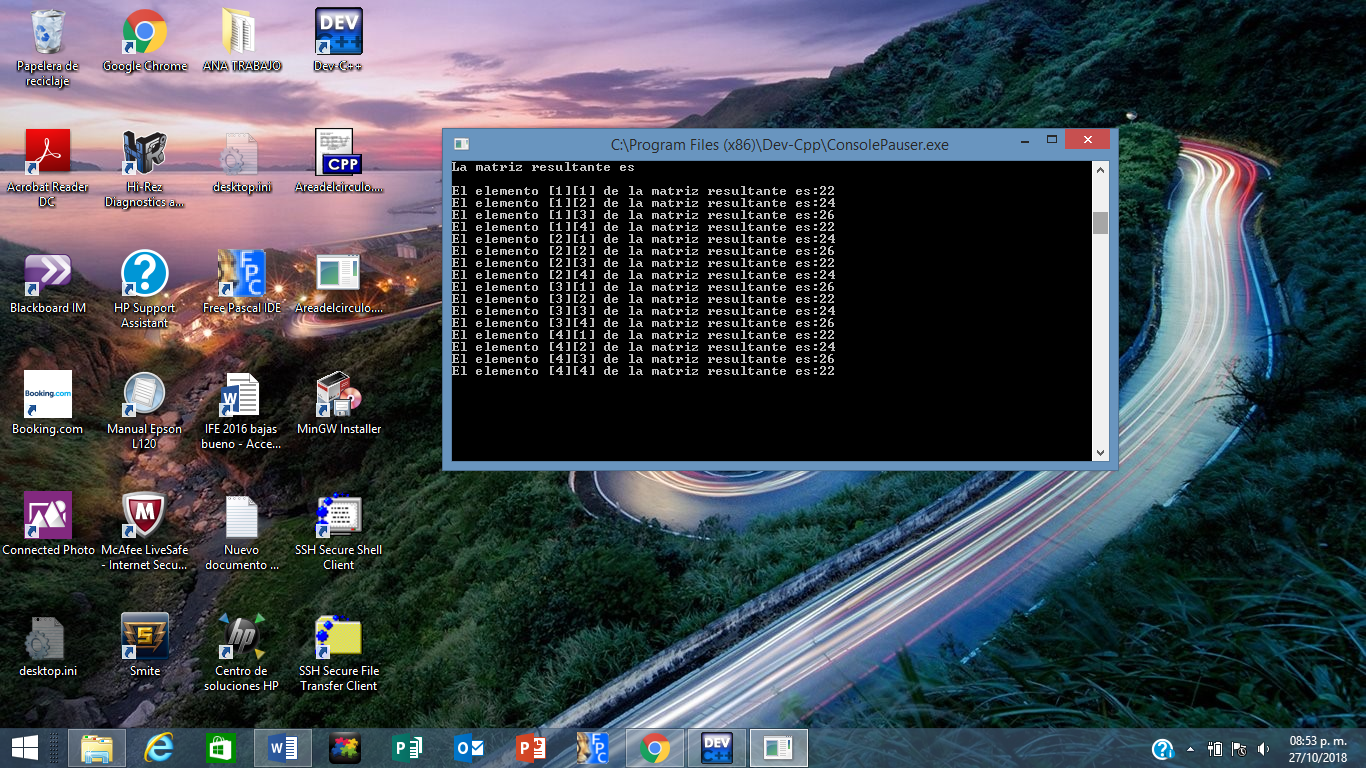
Los arreglos que utilizamos en este programa fueron de una dimension mayor a la anterior de 4x4, es decir para contener a 16 elementos diferentes, y el unico objetivo era sumar dos matrices bidimensionales e imprimir los valores obtenidos en una tercera matriz con las mismas dimensiones a las dos anteriores.





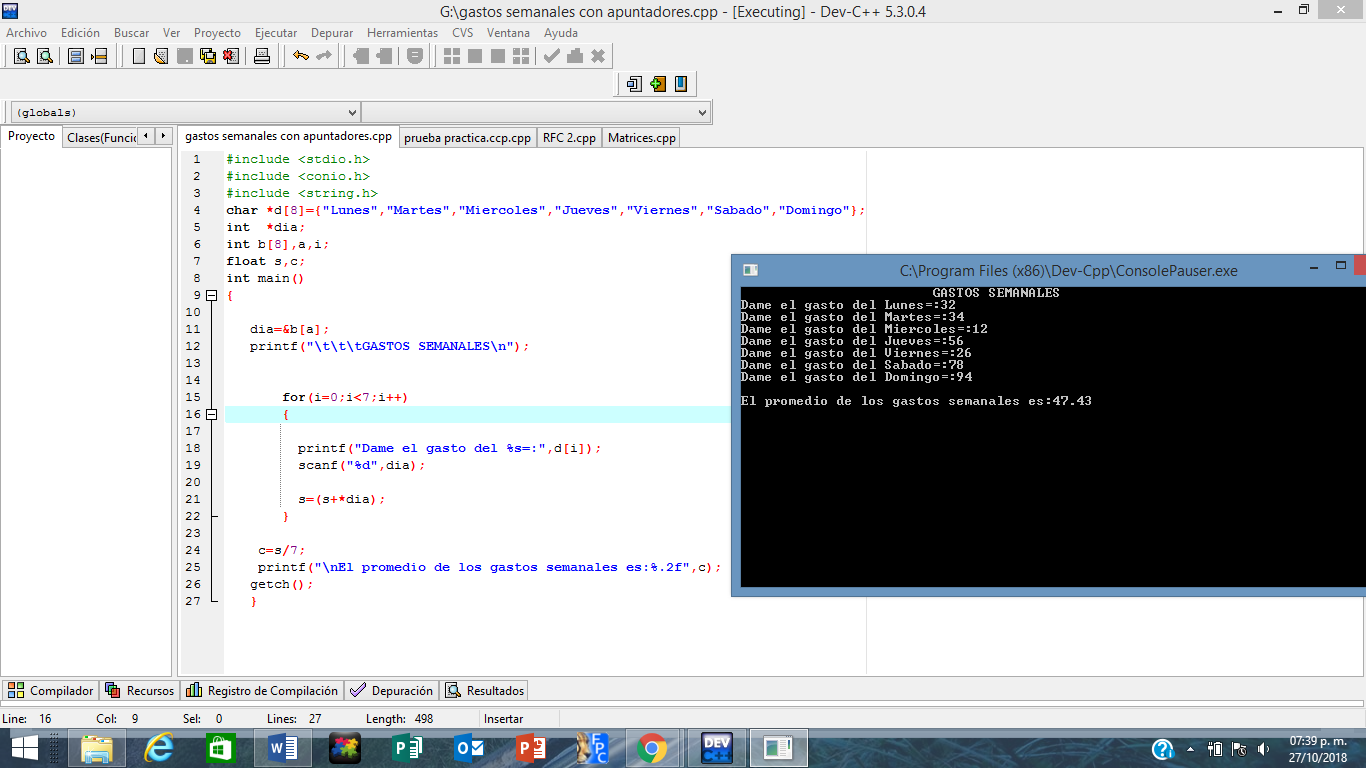






ACTIVIDAD 3:“Promedio de los gastos mensuales”

Aquí el procedimiento no fue complicado, sin embargo implico el uso de otro tipo de arrays bastante conocido, las cadenas de caracteres (que se pueden utilizar solo si se a añade la librería string al archivo), la cual se inicializo con los nombres de los días de la semana, y requirió el uso de un for para ser leído cada uno de sus elementos, en orden de los saldos que se fueran pidiendo al usuario. Al principio se me complico entender como adjuntar ambos arreglos que eran del tipo unidimensional, con un solo for, pues ambos tenían tipos de variables distintas, pero al final se cambió los parámetros de la función for, de tal manera que al contrario del programa anterior, aquí el for se encargara de las posiciones del arreglo de los nombres de los días de la semana y no del arreglo para los gastos que se fueran pidiendo al usuario



CONCLUSION: Los apuntadores facilitan el trabajo de los arreglos porque ya no necesitan depender de la dirección de un for para tener el acceso a cada uno de los elementos que integran a un arreglo, pues un apuntador trabaja con direcciones de memoria y es mucho más fácil llegar a la información que se encuentra almacenada en la variable, al reconocer su ubicación. Además es mucho más fácil relacionar datos del tipo int o float con caracteres, o en su defecto, cadenas de caracteres mediante arreglos, pues ambos son conjuntos de valores que pueden ser enlazados a partir de un solo for.