|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre de la práctica** | **Apuntadores** | | | **No.** | **5** |
| **Asignatura:** | **METODOS NUMERICOS** | **Carrera:** | **INGENIERIA EN SISTEMAS COMPUTACIONALES** | **Duración de la práctica (Hrs)** |  |

1. **Competencia(s) específica(s):**

**II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):**

Aula

**III. Material empleado:**

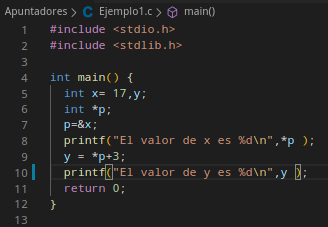
Visual studio code

Gcc compiler linux

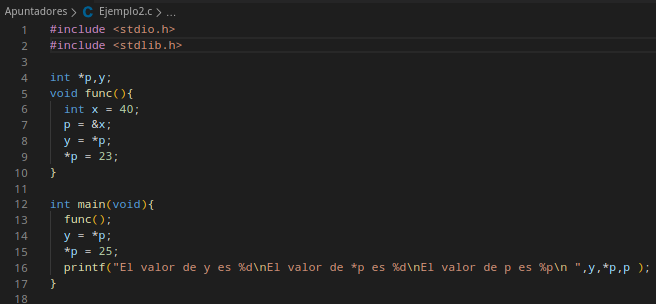
**IV. Desarrollo de la práctica:**

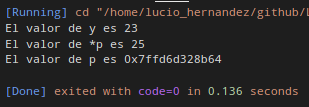
**Lamina 15**

En el primer ejemplo realizamos un codigo que ayuda a comprender como es que asignamos la referencia de memoria a un apuntador de otra variable.

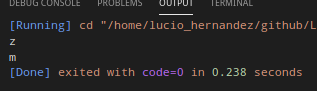
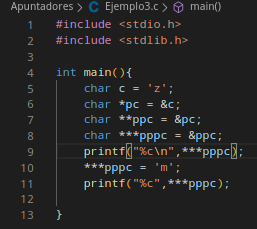


En el siguiente ejemplo podemos ver que una forma no valida de asignar valores a un apuntador puede ser con una variable local de una funcion ya que esta variable no continua con su vida util en otras funciones, esto hace que el apuntador tenga la direccion de un elemento que ya no es necesario para el programa.

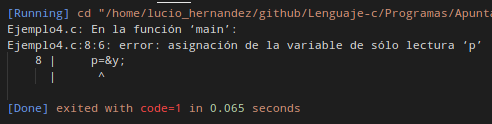
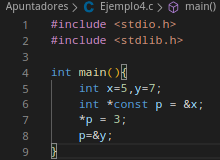




Como podemos deducir los apuntadores ocupan un espacio en memoria que puede ser referenciado desde otro apuntador, esto lo podemos ver en el siguiente programa.

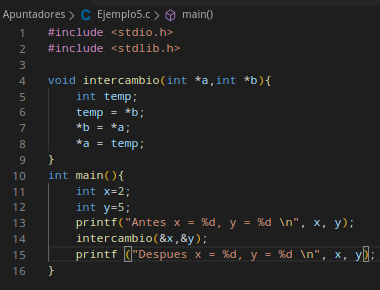


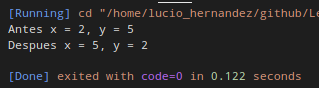
Un apuntador tambien puede ser dafinido como constante, esto lo veremos al tratrar de asignarle una direccion nueva a un apuntador y no al tratar de modificar el valor de la variable a la que apunta.



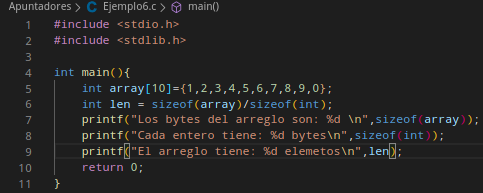
Como podemos ver esto marca un error en tratar de darle una segunda asignacion al puntero.

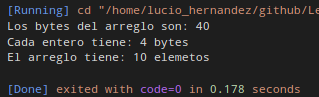
Los punteros tambien pueden presentarse como parametros de una funcion, ya que estas son un tipo de variable, con la diferiencia de que guardan la instancia del valor en memoria.



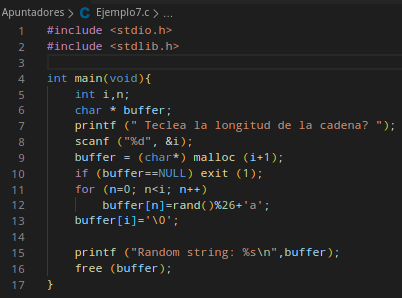


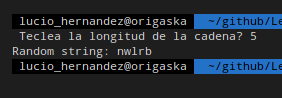
Ahora vemos la funcion sizeof, que retorna el tamaño de una estructura o un tipo de dato en especifico.



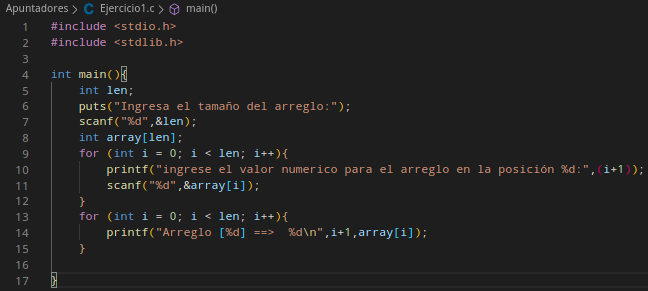


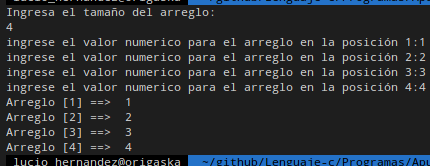
En c tambien podemos colocar espacios de memoria dinamica, esto es util para programas en los que el usuario interactua con el programa, y para esto esisten las funciones malloc y free como podemos ver en el ejemplo siguiente.



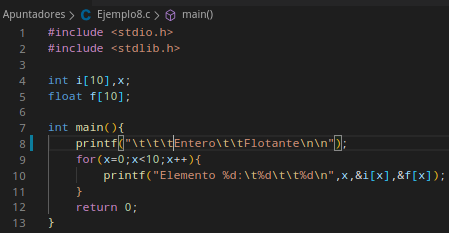


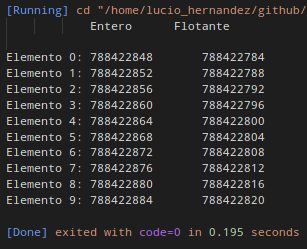
El primer ejercicio propuesto es crear un arreglo de el cual el usuario ingrese su longitud, para este no use malloc ya que es posible hacerlo sin el uso de esta herramienta, como se muestra a continuación:



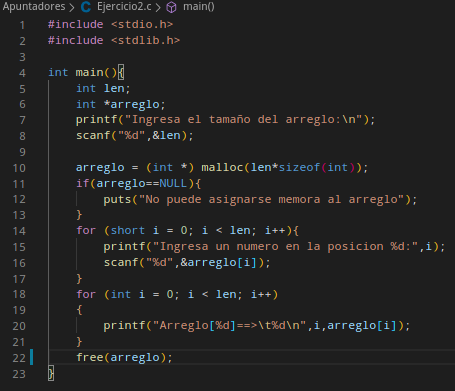


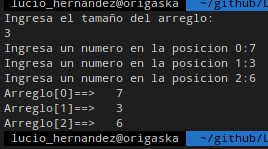
En el siguiente ejemplo podemos ver como es que estan colocados los espacios en memoria de un arreglo y que cada posicion de este es una seccion distinta, para saber su direccion se usa ‘&’.



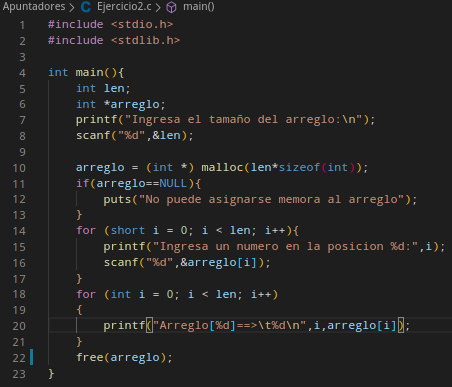


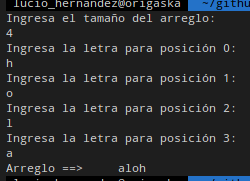
Ahora realizamos un programa que recupere la cantidad de numeros que recibe un arreglo, a este le asignamos memoria, lo llenamos y mostramos por medio de apuntadores, esto lo realizamos de la siguiente forma:





Como ultimo ejemplo nos solicita que inresemos un prohgrama que asigne de forma dinamica una cadena de tipo char, que el usuario ingrese la cantidad de datos, las letras o caracteres y que los muestre de forma inversa a como se introducen.



.

**V. Conclusiones:**

El uso de apuntadores nos ayuda a comprender como es que se colocan los datos de un programa en la memoria ram y nos ayuda a mejorar el tiempo de respuesta de nuestros programas, esto convinado con la asignacion de memoria dinamica nos ayudan a comprender como funcionan las funciones que nos otorga el lenguaje y podemos realizar programas mas eficientes en las tareas que decemos realizar.