



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* ALEJANDRO ESTEBAN PIMENTEL ALARCON

*Asignatura:* FUNDAMENTOS DE PROGRAMACIÓN

*Grupo:* 3

*No de Práctica(s):* 7

*Integrante(s):* MIRANDA GUTIERREZ CELINE

*No. de Equipo de  
cómputo empleado:* 41

*No. de Lista o Brigada:* 30

*Semestre:* 2020-1

*Fecha de entrega:* 03/10/2019

*Observaciones:*

---

---

CALIFICACIÓN: \_\_\_\_\_

**PRÁCTICA 7**  
**“FUNDAMENTOS DE LENGUAJE C”**

INTRODUCCIÓN

El Lenguaje C fue creado por Denis Ritchie y Ken Thompson en 1972 como herramienta para los programadores y su principal objetivo consiste en ser un lenguaje útil. Posee las siguientes características:

- **Moderno:**

Los programas desarrollados utilizan las técnicas de programación estructurada y poseen un diseño modular.

- **Eficiente:**

Su diseño aprovecha las posibilidades y los recursos de los ordenadores actuales.

- **Portátil:**

Los programas escritos en C para un ordenador concreto son, por lo general, fácilmente transportables a otro ordenador de distinta arquitectura con muy pocas modificaciones, siempre que en el desarrollo de éstos se utilice el conjunto estándar de bibliotecas del lenguaje.

- **Potente y flexible:**

Se emplea para resolver problemas físicos y de ingeniería. A diferencia de los demás lenguajes de alto nivel, posee control sobre aspectos asociados a los lenguajes ensambladores.

- **Compilado:**

Es necesario disponer de un editor de texto y un compilador en modo de línea de órdenes, o de un entorno integrado de desarrollo (IDE). El compilador permite, a partir del programa en código fuente, obtener la traducción a código objeto en formato ejecutable. En el sistema operativo Windows, los ficheros ejecutables poseen la extensión .EXE.

Un programa necesita trabajar con datos.

Algunos están preseleccionados antes de la ejecución del programa y mantienen sus valores inalterados durante la misma. Se denominan constantes. Otros pueden recibir una o más asignaciones de valor durante la ejecución del programa.

Se denominan **variables**.

<i>Tipo</i>	<i>Bits</i>	<i>Valor Mínimo</i>	<i>Valor Máximo</i>
<i>float</i>	32	3.4 E-38	3.4 E38
<i>double</i>	64	1.7 E-308	1.7 E308
<i>long double</i>	80	3.4 E-4932	3.4 E4932

### MOSTRAR Y LEER

<i>Tipo de dato</i>	<i>Especificador de formato</i>
<i>Entero</i>	%d, %i, %ld, %li, %o, %x
<i>Flotante</i>	%f, %lf, %e, %g
<i>Carácter</i>	%c, %d, %i, %o, %x
<i>Cadena de caracteres</i>	%s

### OPERADORES

<i>Operador</i>	<i>Operación</i>	<i>Uso</i>	<i>Resultado</i>
<b>+</b>	Suma	125.78 + 62.5	188.28
<b>-</b>	Resta	65.3 - 32.33	32.97
<b>*</b>	Multiplicación	8.27 * 7	57.75
<b>/</b>	División	15 / 4	3.75
<b>%</b>	Módulo	4 % 2	0

### COMPARADORES

<i>Operador</i>	<i>Operación</i>	<i>Uso</i>	<i>Resultado</i>
<b>==</b>	Igual que	'h' == 'H'	Falso
<b>!=</b>	Diferente a	'a' != 'b'	Verdadero
<b>&lt;</b>	Menor que	7 < 15	Verdadero
<b>&gt;</b>	Mayor que	11 > 22	Falso
<b>&lt;=</b>	Menor o igual	15 <= 22	Verdadero
<b>&gt;=</b>	Mayor o igual	20 >= 35	Falso

## OPERADORES LOGICOS

### *Operador Operación*

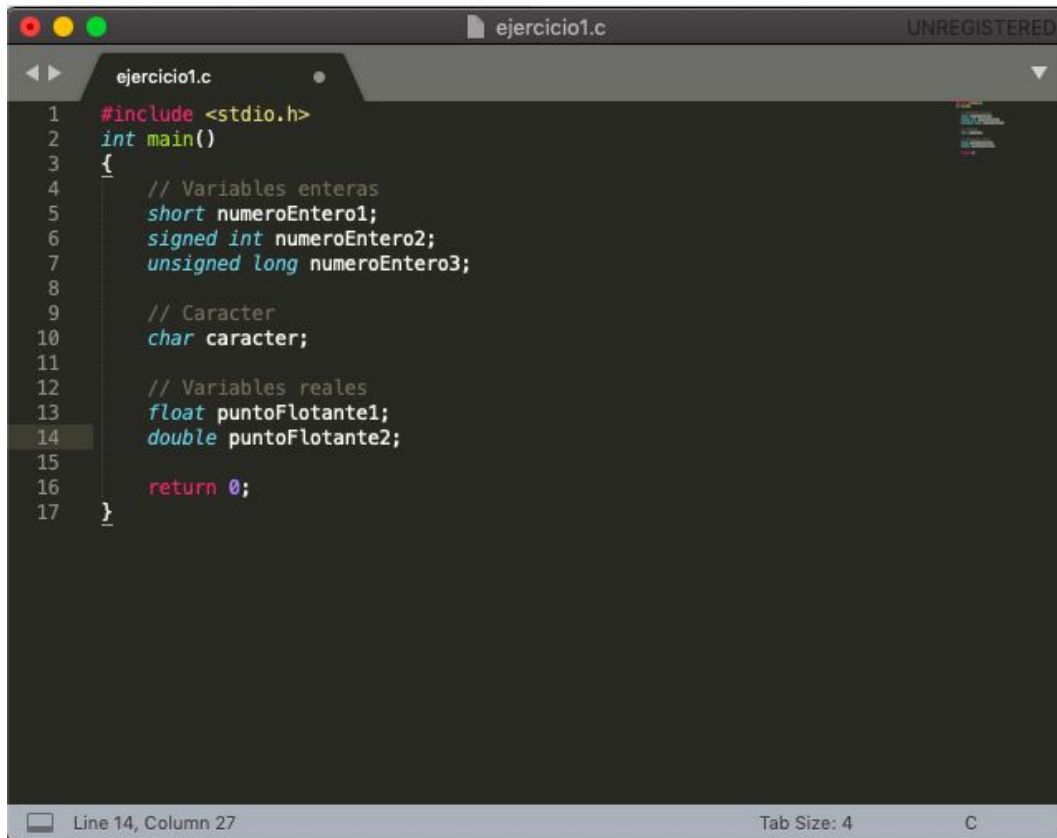
<i>!</i>	No
<i>&amp;&amp;</i>	Y
<i>  </i>	O

### OBJETIVO

Elaborar programas en lenguaje C utilizando las instrucciones de control de tipo secuencia, para realizar la declaración de variables de diferentes tipos de datos, así como efectuar llamadas a funciones externas de entrada y salida para asignar y mostrar valores de variables y expresiones.

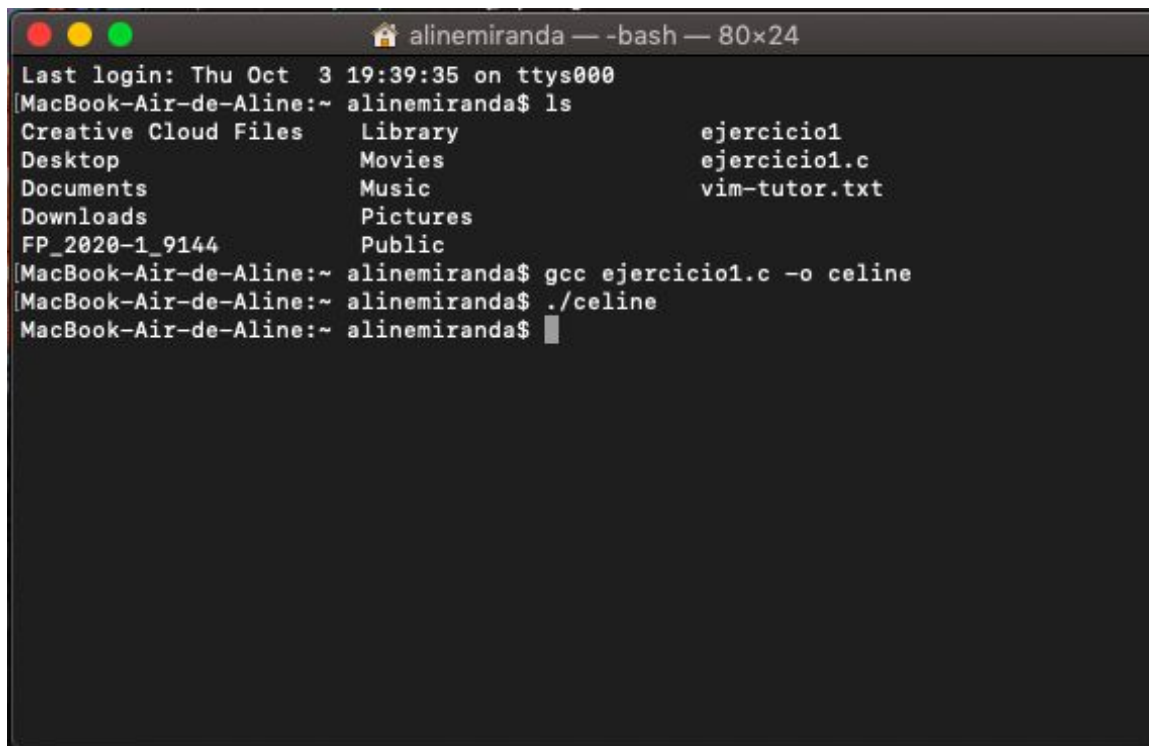
### ACTIVIDAD 1

Al realizar esta actividad en "Sublime text" lo que se puede observar es la declaración de variables para la mejor comprensión del funcionamiento del lenguaje c. Aquí claramente se puede observar como utilizamos variables enteras como: short, signed int y unsigned long. Se practicó con char el cual significa "carácter" y las variables reales como: float y double. Para finalizar el lenguaje c se debe poner "return 0;" y cerrar la llave. Con eso el programa sabe que se culmina.



```
ejercicio1.c
1  #include <stdio.h>
2  int main()
3  {
4      // Variables enteras
5      short numeroEntero1;
6      signed int numeroEntero2;
7      unsigned long numeroEntero3;
8
9      // Caracter
10     char caracter;
11
12     // Variables reales
13     float puntoFlotante1;
14     double puntoFlotante2;
15
16     return 0;
17 }
```

Line 14, Column 27      Tab Size: 4      C

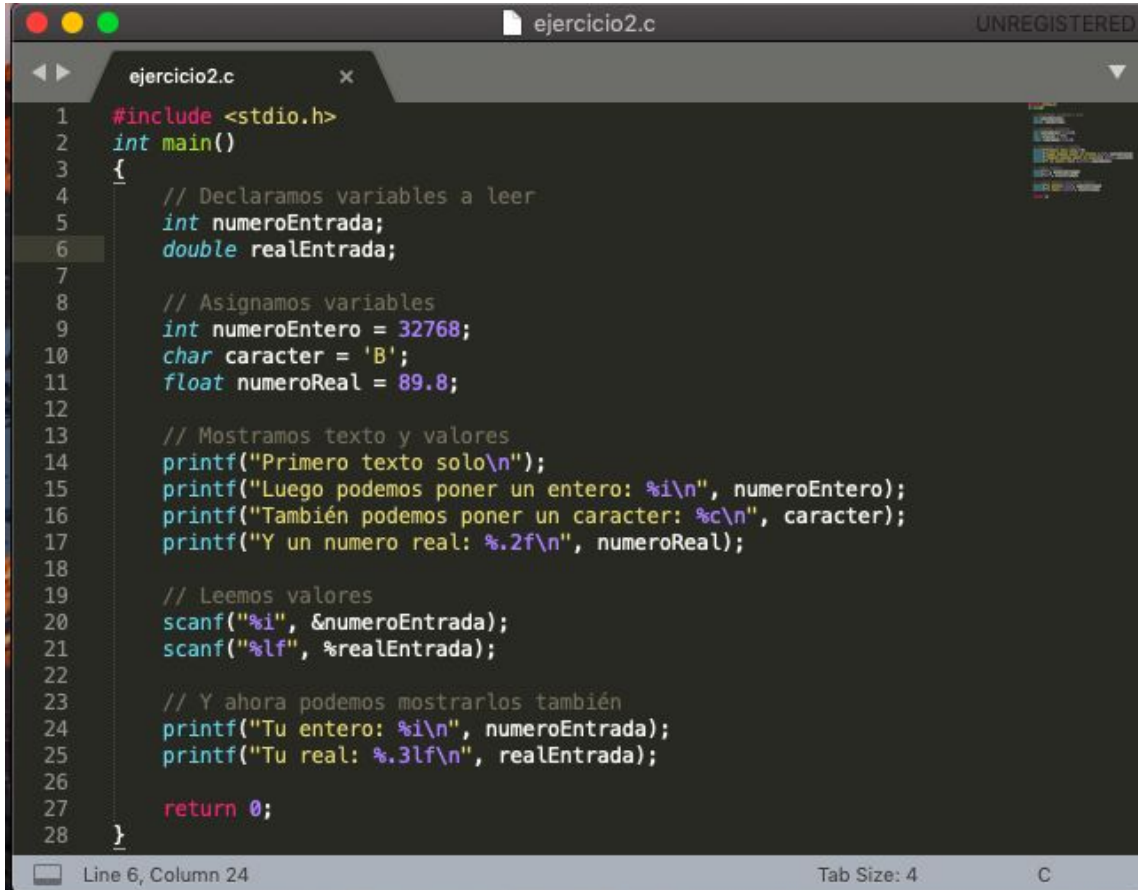


```
alinemiranda — -bash — 80x24
Last login: Thu Oct  3 19:39:35 on ttys000
MacBook-Air-de-Aline:~ alinemiranda$ ls
Creative Cloud Files  Library          ejercicio1
Desktop              Movies           ejercicio1.c
Documents            Music            vim-tutor.txt
Downloads            Pictures
FP_2020-1_9144       Public
MacBook-Air-de-Aline:~ alinemiranda$ gcc ejercicio1.c -o celine
MacBook-Air-de-Aline:~ alinemiranda$ ./celine
MacBook-Air-de-Aline:~ alinemiranda$
```

## ACTIVIDAD 2

En la actividad se practica utilizando especificadores de formato tales como: %i %lf. Sabemos que al poner un punto antes de y en seguida cualquier número, se da la instrucción que lea después del punto decimal el número que nosotros ingresamos. También se ocupan int, char, float. Para que en nuestra terminal se impriman los valores debemos poner **printf**("texto que queremos se vea reflejado") y para saltar espacios se utiliza **\n**.

Esta actividad nos sirve de práctica para poder observar como es el manejo del lenguaje C y las condiciones para que se vea reflejado correctamente en la terminal.



```
1  #include <stdio.h>
2  int main()
3  {
4      // Declaramos variables a leer
5      int numeroEntrada;
6      double realEntrada;
7
8      // Asignamos variables
9      int numeroEntero = 32768;
10     char caracter = 'B';
11     float numeroReal = 89.8;
12
13     // Mostramos texto y valores
14     printf("Primero texto solo\n");
15     printf("Luego podemos poner un entero: %i\n", numeroEntero);
16     printf("También podemos poner un caracter: %c\n", caracter);
17     printf("Y un numero real: %.2f\n", numeroReal);
18
19     // Leemos valores
20     scanf("%i", &numeroEntrada);
21     scanf("%lf", &realEntrada);
22
23     // Y ahora podemos mostrarlos también
24     printf("Tu entero: %i\n", numeroEntrada);
25     printf("Tu real: %.3lf\n", realEntrada);
26
27     return 0;
28 }
```

Line 6, Column 24      Tab Size: 4      C

```
alinemiranda — ejecutar — 80x24
Last login: Thu Oct  3 21:25:03 on ttys000
MacBook-Air-de-Aline:~ alinemiranda$ ls
Creative Cloud Files  Library          celine
Desktop              Movies           ejercicio1
Documents            Music            ejercicio1.c
Downloads            Pictures         ejercicio2.c
FP_2020-1_9144       Public           vim-tutor.txt
MacBook-Air-de-Aline:~ alinemiranda$ gcc ejercicio2.c -o ejecutar
MacBook-Air-de-Aline:~ alinemiranda$ ./ejecutar
Primero texto solo
Luego podemos poner un entero: 32768
También podemos poner un caracter: B
Y un numero real: 89.80
█
```

### ACTIVIDAD 3

Aquí practicamos con operadores como la división entre dos números, los cuales designamos con anterioridad, aquí el elemento clave es “doublé” es decir, le da más precisión a la operación, el cual se ve reflejado en la terminal.

```
ejercicio3.c  UNREGISTERED
ejercicio3.c x
1  #include <stdio.h>
2  int main()
3  {
4      int dos, tres, cuatro, cinco;
5      double resultado;
6
7      dos = 2;
8      tres = 3;
9      cuatro = 4;
10     cinco = 5;
11
12     resultado = cinco/dos;
13     printf("5 / 2 = %.1lf\n", resultado);
14
15     resultado = (double)cinco/dos;
16     printf("5 / 2 = %.1lf\n", resultado);
17
18     return 0;
19 }
```

Line 18, Column 14      Tab Size: 4      C

```
alinemiranda — -bash — 80x24
Last login: Thu Oct  3 21:27:19 on ttys000
MacBook-Air-de-Aline:~ alinemiranda$ ls
Creative Cloud Files  Movies                ejercicio1
Desktop              Music                 ejercicio1.c
Documents            Pictures              ejercicio2.c
Downloads            Public                ejercicio3.c
FP_2020-1_9144       celine                vim-tutor.txt
Library              ejecutar
MacBook-Air-de-Aline:~ alinemiranda$ gcc ejercicio3.c -o ejecutar3
MacBook-Air-de-Aline:~ alinemiranda$ ./ejecutar3
5 / 2 = 2.0
5 / 2 = 2.5
MacBook-Air-de-Aline:~ alinemiranda$
```

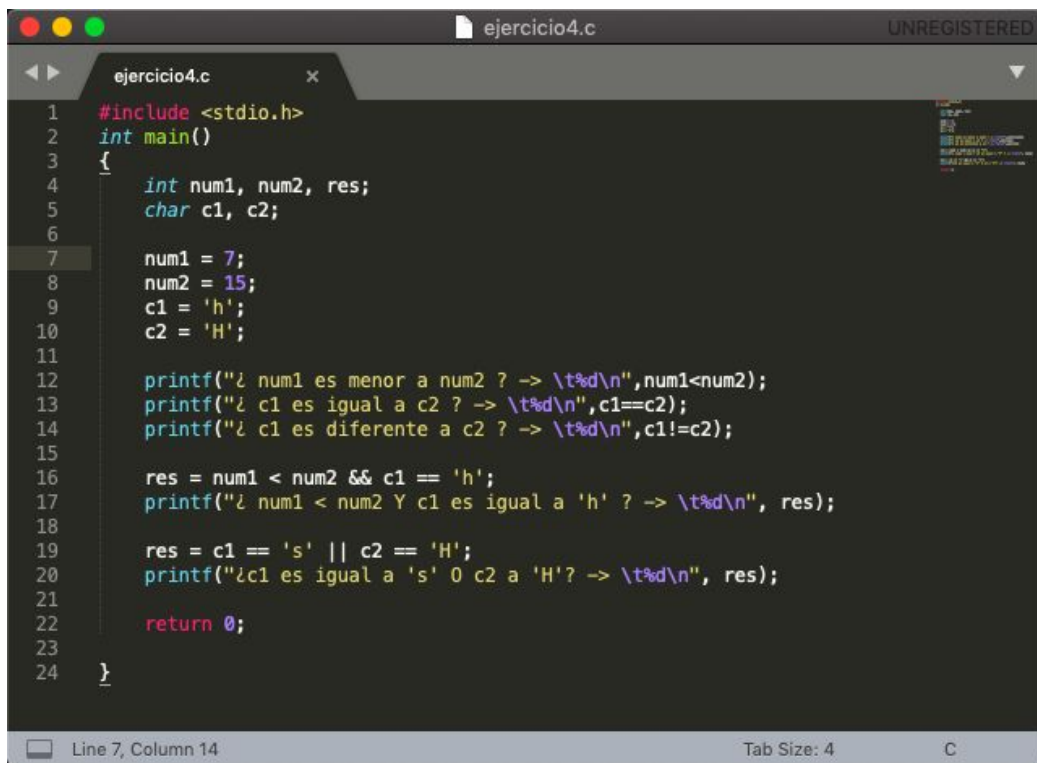
## ACTIVIDAD 4

En esta actividad se utilizamos comparadores tale como: == y <

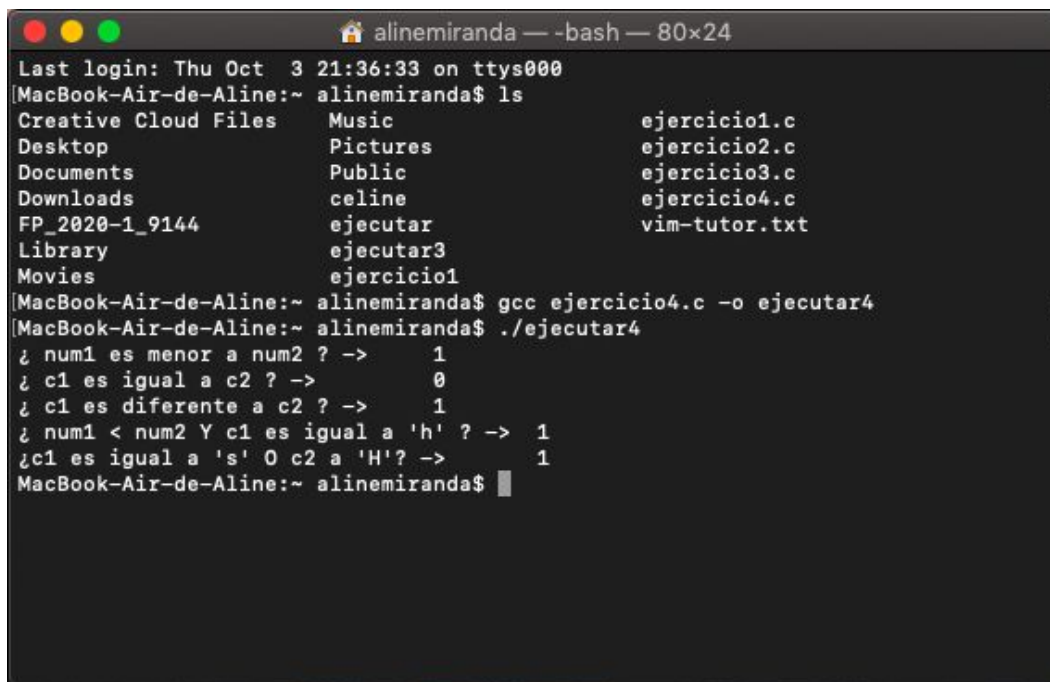
También operadores lógicos como: && que significa “Y”, || que significa “O” y ! que en este



caso es "NO". Este lenguaje es muy parecido a los diagramas de flujo que vemos en clase ya que utiliza comparadores e ingresa números y variables.



```
ejercicio4.c
1  #include <stdio.h>
2  int main()
3  {
4      int num1, num2, res;
5      char c1, c2;
6
7      num1 = 7;
8      num2 = 15;
9      c1 = 'h';
10     c2 = 'H';
11
12     printf("¿ num1 es menor a num2 ? -> %d\n", num1 < num2);
13     printf("¿ c1 es igual a c2 ? -> %d\n", c1 == c2);
14     printf("¿ c1 es diferente a c2 ? -> %d\n", c1 != c2);
15
16     res = num1 < num2 && c1 == 'h';
17     printf("¿ num1 < num2 Y c1 es igual a 'h' ? -> %d\n", res);
18
19     res = c1 == 's' || c2 == 'H';
20     printf("¿ c1 es igual a 's' O c2 a 'H' ? -> %d\n", res);
21
22     return 0;
23 }
24
```



```
alinemiranda — -bash — 80x24
Last login: Thu Oct 3 21:36:33 on ttys000
MacBook-Air-de-Aline:~ alinemiranda$ ls
Creative Cloud Files  Music          ejercicio1.c
Desktop              Pictures       ejercicio2.c
Documents            Public         ejercicio3.c
Downloads            celine        ejercicio4.c
FP_2020-1_9144       ejecutar      vim-tutor.txt
Library              ejecutar3
Movies               ejercicio1
MacBook-Air-de-Aline:~ alinemiranda$ gcc ejercicio4.c -o ejecutar4
MacBook-Air-de-Aline:~ alinemiranda$ ./ejecutar4
¿ num1 es menor a num2 ? -> 1
¿ c1 es igual a c2 ? -> 0
¿ c1 es diferente a c2 ? -> 1
¿ num1 < num2 Y c1 es igual a 'h' ? -> 1
¿ c1 es igual a 's' O c2 a 'H' ? -> 1
MacBook-Air-de-Aline:~ alinemiranda$
```

CONCLUSIONES

Considero que el lenguaje de C es muy importante debido a que nos permite visualizar y plantear la resolución de problemas, ahorita apenas estamos introduciéndonos y aprendiendo a utilizar de manera correcta debido a que lo que pude observar durante la realización de las practicas es que por un mínimo error no sale correctamente en la terminal. Una ventaja es que en mi caso, la misma terminal te resalta tu error y el motivo por el cual no puede ejecutar el programa así que para nosotros es muchísimo más sencillo ubicar y corregir para la mejor ejecución del mismo. Es un programa que nos servirá a lo largo de nuestra carrera y no dudo que también en nuestra vida profesional.