



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

Profesor: M.I. Oscar René Valdez Casillas

Fundamentos de Programación

Práctica de estudio 06: Entorno y fundamentos del lenguaje C.

Integrantes:

Arteaga Gonzalez Jaime Alejandro

Márquez Alan

Valencia Francisco Valentín

García Jiménez Joel David

Grupo: 31

Arteaga, J., Márquez, A., García, J. y Valencia, V., (2024). Práctica 06:Entorno y fundamentos del lenguaje C.

Índice:

1. Resumen.....	
.....2	
2. Introducción.....	
.....2	
3. Objetivo.....	
.....2	
4. Investigación: funciones e indicaciones de lo que realizan y dar su prototipo definido.....	2,
3	
• sizeof.	
• printf.	
• scanf.	
5. Ejercicios.....	
.....3,4	
• Programa que muestre en pantalla el tamaño en Bytes que ocupa cada tipo de dato simple definido en C.	
• Programa en C que compare dos números dados e indique cuál de los dos es mayor.	
6. Conclusiones.....	
.....5,6	
• Reflexión individual de los integrantes del equipo.	
7. Bibliografía.....	
.....6	
• Fuentes y referencias utilizadas en el informe.	

Resumen: Por fin entramos de lleno a lo que es el lenguaje de programación C, después de muchos algoritmos, análisis de lecturas, empezamos a lo que venimos a hacer, la receta que teníamos, pasarla a la computadora. En el reporte haremos y programaremos en el lenguaje C, algunos de los problemas a plantear, utilizando variables, condiciones, etc, de esta manera poniendo en práctica los fundamentos de la programación en C e ir agarrando práctica para próximos problemas más complicados. En dicho reporte también investigamos algunas funciones importantes para la programación en C y para lo que conlleva en el.

Introducción: El aprendizaje de la programación en C nos ayuda a desarrollar habilidades para resolver problemas y en la creación de software. Este lenguaje, ofrece diversas instrucciones de control que permiten a los programadores estructurar sus programas de manera efectiva. La práctica tiene como objetivo elaborar programas en C utilizando instrucciones de control de tipo secuencia. Se aprenderá a declarar variables de diferentes tipos de datos, así como a realizar llamadas a funciones externas de entrada y salida. Esto permitirá asignar y mostrar valores de variables y expresiones de manera precisa, creando bases para el desarrollo de aplicaciones más complejas.

Objetivo:El alumno elaborará 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.

Investigación:

1. Buscar en qué biblioteca de funciones de C se encuentran las siguientes funciones, indicar que es lo que realizan y dar su prototipo definido:

-sizeof: Proporciona la cantidad de almacenamiento, en bytes, necesaria para almacenar un objeto del tipo del operando. Este operador permite no tener que especificar tamaños de datos dependientes del equipo en los programas.

-printf: La función printf en C se encuentra en la biblioteca estándar ANSI C, específicamente en el archivo stdio.h.

La biblioteca estándar está vinculada por defecto por el compilador de C y suele ser proporcionada por el sistema operativo o el compilador. Por ejemplo, en la mayoría de los sistemas Linux, la biblioteca se encuentra en libc.so, mientras que en MS Windows se encuentra en msvcrt.dll.

Una librería es un conjunto de funciones que pueden ser reutilizadas y que suelen tener características comunes o compartir un mismo objetivo.

-scanf: La función scanf() de C se encuentra en el archivo de encabezado <stdio.h>. La función scanf() permite leer datos desde el teclado, como enteros, números decimales o cadenas de caracteres. Para ello, se utiliza un parámetro de formato que es una cadena de caracteres que indica el tipo de dato que se espera.

Ejercicios:

Ejercicio 1: Realizar un programa que muestre en pantalla el tamaño en Bytes que ocupa cada tipo de dato simple en C.

<https://onlinegdb.com/Wrq-1IPqUB>

```
1  /*Arteaga Gonzales Jaime Alejandro
2  Márquez Perez Alan
3  Valencia Francisco Jaime Alejandro
4  Garcia Jiménez Joel David
5  24 de septiembre de 2024
6
7  Realizar un programa que muestre en pantalla el tamaño en Bytes que ocupa cada tipo de dato simple definido en C
8  Inicio
9      Declarar variable "a" de tipo entero
10     Declarar variable "b" de tipo flotante
11     Declarar variable "c" de tipo precision
12     Declarar variable "d" de tipo carácter
13
14     Declarar arreglo "A" de enteros con valores {1, 2, 3, 4}
15     Definir estructura "e" que contiene:
16         variable "a1" de tipo entero
17         variable "a2" de tipo doble precision
18     Imprimir el tamaño en bytes de la variable "a"
19     Imprimir el tamaño en bytes de la variable "b"
20     Imprimir el tamaño bytes de la variable "c"
21     Imprimir el tamaño en bytes de la variable "d"
22     Imprimir el tamaño total en bytes del arreglo arreglo "A"
23     Calcular e Imprimir el número de elementos del arreglo "A"
24     Imprimir el tamaño en bytes de la estructura "e"
25     Fin*/
26
27 #include <stdio.h>
```

```
main.c
26
27 #include <stdio.h>
28
29 int main(){
30
31     int a;
32     float b;
33     double c;
34     char d;
35
36     int A[] = {1,2,3,4};
37
38     struct e{
39         int a1;
40         double a2;
41     };
42
43     printf("El tamaño de entero es : %d bytes\n",sizeof(a));
44     printf("El tamaño de float es : %d bytes\n",sizeof(b));
45     printf("El tamaño de double es : %d bytes\n",sizeof(c));
46     printf("El tamaño de char es : %d bytes\n",sizeof(d));
47     printf("El tamaño total del arreglo es : %d bytes\n",sizeof(A));
48     printf("El numero de casillas del arreglo es : %d bytes\n",sizeof(A)/sizeof(A[0]));
49     printf("El tamaño de la estructura es : %d bytes\n",sizeof(struct e));
50
51     return 0;
52 }
```

```
El tamaño de entero es : 4 bytes
El tamaño de float es : 4 bytes
El tamaño de double es : 8 bytes
El tamaño de char es : 1 bytes
El tamaño total del arreglo es : 16 bytes
El numero de casillas del arreglo es : 4 bytes
El tamaño de la estructura es : 16 bytes
```

Ejercicio 2: Realizar un programa en C que compare dos números dados e indique cuál de los dos es mayor. Mostrar todas las comparaciones en pantalla.

```
1: /* Arteaga Gonzalez Jaime Alejandro
2: Márquez Pérez Alan
3: Valencia Francisco Valentín
4: García Jiménez Joel David
5: 24 de Septiembre de 2024
6
7: Realizar un programa en C que compare dos números dados e indique cuál de los dos es mayor. Mostrar todas las comparaciones en pantalla.
8
9: INICIO
10:  DECLARAR a, b COMO ENTERO
11
12:  IMPRIMIR "Escriba un número: "
13:  LEER a
14
15:  IMPRIMIR "Escriba otro número: "
16:  LEER b
17
18:  SI a > b ENTONCES
19:      IMPRIMIR a, " - ", b, " es positivo"
20:      IMPRIMIR a, " es mayor."
21:  SINO SI a < b ENTONCES
22:      IMPRIMIR a, " - ", b, " es negativo"
23:      IMPRIMIR b, " es mayor."
24:  SINO
25:      IMPRIMIR a, " - ", b, " es igual a 0. Los números son iguales"
26:  FIN SI
27
28: FIN
29: */
```

```
31 #include <stdio.h>
32
33 int main()
34 {
35     int a;
36     int b;
37     printf("\n Escriba un número: ");
38     scanf("%d", &a );
39     printf("\n Escriba otro numero: ");
40     scanf("%d", &b );
41     if ( a > b )
42         printf( "\n %d - %d es positivo \n %d es mayor.", a, b, a );
43     else if( a < b )
44         printf( "\n %d - %d es negativo \n %d es mayor.", a, b, b );
45     else
46         printf( "\n %d - %d es igual a 0. Los números son iguales" );
47     return 0;
48 }
```

Conclusiones:

Alan: Uno de los logros principales de esta práctica es el manejo de variables de diferentes tipos de datos, además de la implementación de las funciones de entrada y salida, como printf y scanf. El uso de estas funciones junto con la exploración de operadores como sizeof nos permite entender cómo interactúa el código con la memoria del sistema. Esto no solo ayuda a optimizar recursos, sino que también enseña la importancia de la correcta administración de la memoria en programas que escalan en complejidad.

La práctica resalta la importancia de la precisión y el cuidado en la codificación en C, ya que este lenguaje, a diferencia de otros como Java, requiere una mayor atención al detalle en cuanto a la gestión de la memoria y los tipos de datos. Errores menores pueden tener un

impacto considerable en la ejecución del programa, como lo mencionan algunos de los integrantes en sus conclusiones.

Jaime: Como se mencionó en el resumen y en la introducción, ya estamos de lleno programando en C, es algo difícil adaptarse a este nuevo lenguaje de programación, bueno no difícil pero yo personalmente siento que sí tiene sus diferencias con los diversos lenguajes que hay, por ejemplo en lo personal yo trabajaba con un Lenguaje Java, y aunque sí hay diferencias, el lenguaje de programación C, es una herramienta muy útil para ver de mejor manera la estructura y datos de nuestro programa a seguir, todo eso formó parte de la práctica de hoy, ya que con la investigación, nos dimos cuenta que incluso en la programación tiene muchas diferencias con el pseudocódigo, estamos conociendo que claro es la receta pero al ya estar programando tienes que ser muy cauteloso y ver que vas a poner antes de indicarle una función a C, hay que recordar que algún error que tengamos a la hora de programación, nos arruinara o no tendremos una estructura correcta de todo el programa, por lo que un mínimo error, nos puede llevar a la confusión o al no hacer nuestra práctica de manera correcta. En conclusión los objetivos planteados al principio de la práctica fueron correctos y me da mucho gusto que ya empezamos de lleno a lo que venimos a hacer, que es la programación y aún mejor en un nuevo lenguaje.

Valentín: En esta práctica se aplicaron conocimientos fundamentales sobre el manejo de variables, funciones de entrada y salida como printf y scanf, así como la importancia de operadores como sizeof para la gestión de memoria. Cada paso en el proceso permite ver de manera más clara cómo interactúa el código con el hardware, optimizando el uso de los recursos del sistema.

En conclusión, esta práctica ha permitido fortalecer fundamentos en programación, y las herramientas básicas necesarias para problemas más complejos en este lenguaje.

Joel: La elaboración de programas en lenguaje C utilizando instrucciones de control de tipo secuencia nos ha permitido adquirir competencias en la programación. A través de la declaración de variables de diferentes tipos de datos y la implementación de funciones de entrada y salida, hemos logrado no solo asignar y mostrar valores, sino también comprender la importancia de la estructura y

el flujo lógico en sus programas. Esta práctica proporciona una base sólida para avanzar hacia conceptos más complejos en la programación.

BIBLIOGRAFÍAS:

colaboradores de Wikipedia. (2024c, mayo 13). *Scanf*. Wikipedia, la Enciclopedia Libre.

<https://es.wikipedia.org/wiki/Scanf>

Proyecto: Sistema de ayuda al C. (s. f.).

https://ccia.ugr.es/~jfv/ed1/c/cdrom/cap3/f_cap35.htm#:~:text=La%20funci%C3%B3n%20printf%20transporta%20datos,el%20n%C3%BAmero%20de%20caracteres%20escritos.