UNIVERSIDAD DE LOS ANDES
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN
FUNDAMENTOS DE INFRAESTRUCTURA TECNOLÓGICA - ISIS1304
Trabajo práctico 3

Este proyecto vale 5% de la nota del curso.
Debe ser elaborado individualmente.
No se permite ningún tipo de consulta sobre el proyecto con otras personas.
Se debe entregar por Sicua+ a más tardar el 25 de abril de 2021

A. OBJETIVOS

- Practicar el lenguaje C y desarrollar un programa de baja complejidad.
- Conocer las operaciones de C para el manejo de bits.
- Aplicar lo anterior empaquetando una cadena de caracteres en un vector de enteros.

B. DESCRIPCIÓN DEL TRABAJO

El objetivo del proyecto es desarrollar un programa en C que lea una cadena de caracteres del teclado y la empaquete en un vector de enteros (4 `char` por cada `int`). La cadena puede tener cualquier longitud.

Se recorre la cadena desde el comienzo y cada uno de sus caracteres se va almacenando en el vector. Los caracteres se almacenan a partir de la posición cero, y en cada posición se almacenan en los bytes del entero del más significativo hacia el menos significativo (el carácter nulo del final no se incluye). Si en la última posición sobran bytes, se rellena con el carácter nulo.

Ejemplo. Cadena:

Hexa	0x48	0x6F	0x6C	0x61	0x2E	0x00
Ascii	`'H'`	`'o'`	`'l'`	`'a'`	`'.'`	

Vector de `int`:

	+		significativo	-	+		significativo	-
Hexa	0x48	0x6F	0x6C	0x61	0x2E	0x00	0x00	0x00
Decimal	`1.215'261.793`				`771'751.936`			

Estructuras de datos

- Un vector de `char` para almacenar la cadena (máximo 100 caracteres).
- Un apuntador a `int`: para apuntar al vector de enteros donde se almacena el resultado.

Estructura del programa

El programa solo debe tener dos procedimientos: el `main` y otro procedimiento encargado de calcular las paridades.

`main`:

- Declara e inicializa la cadena. Esto consiste en leer la cadena de caracteres tecleada por el usuario (la cadena no tiene blancos).
- Declara e inicializa el apuntador al vector de `int`. La inicialización consiste en ponerlo a apuntar a un vector de `int` del tamaño mínimo necesario para almacenar la cadena de caracteres. El vector se crea dinámicamente usando la función de C `calloc`.
- Invoca el procedimiento que empaqueta la cadena.
- Imprime el resultado (el vector de `int`) en hexa (ver formato `%X` de `printf`).

Procedimiento de empaquetamiento:

- Tiene dos parámetros: el apuntador a la cadena y el apuntador al vector de enteros.
- Recorre la cadena, carácter por carácter, empaquetándolos en el vector de enteros.

Restricciones y consideraciones

- Las estructuras de datos se declaran e inicializan en el `main`.
- El programa solo debe constar del `main` y el procedimiento de cálculo.
- No puede usar librerías externas que resuelvan directamente el problema; solo las básicas de C (entrada/salida, manejo de cadenas, etc.).
- La cadena tiene entre 1 y 100 caracteres.
- Los programas se calificarán únicamente usando el ambiente de visual de las máquinas virtuales. Si el programa no compila en este ambiente, se considerará que no corre (así compile en otros ambientes).

C. CONDICIONES DE ENTREGA

Entregar el código fuente junto con el ejecutable en un archivo `.zip`. Al comienzo del archivo fuente escriba su nombre, código y correo.

Si su programa no funciona o si su solución tiene particularidades, puede enviar un archivo `.docx` o `.pdf` explicando por qué cree que no funciona o qué fue lo que hizo.

El trabajo es individual. No debe haber consultas con otros estudiantes.

Se puede solicitar una sustentación sobre cualquier parte del trabajo. Dicha sustentación puede afectar la nota.

El proyecto debe ser entregado por Sicua+ a más tardar el 25 de abril de 2021 hasta las 11:50 pm.

D. CRITERIOS DE CALIFICACIÓN PARA LOS PROGRAMAS

La calificación consta de dos partes:

- Ejecución (50%). Se harán 5 pruebas (los dos casos de prueba y otras 3 adicionales). Para cada caso, se revisará si la salida es correcta o no según los requerimientos establecidos en el enunciado. Cada prueba vale 10%.
- Inspección del código (50%). Se consideran tres aspectos:
  - 10% - legibilidad (nombres dicientes de variables, comentarios, indentación)
  - 20% - manejo de bits (uso correcto y eficaz de los operadores de bits de C: >>, &, etc. para resolver el problema)

- 20% - Correcto y eficaz manejo de la estructura de datos (recorrido, manejo de los elementos).

E. CASOS DE PRUEBA

Caso 1:

Entrada: “Hola”

Salida: 0x486F6C61, 0x0

Caso 2:

Entrada: “cadena”

Salida: 0x63616465, 0x6E610000