Proyecto C

Este laboratorio se desarrollará utilizando una instalación (hecha por el estudiante) de Ubuntu 18.04 LTS, 20.04 LTS, o 22.04 LTS. **Si el estudiante usa otra versión, debe indicarlo en el documento.**

Instrucciones generales:
- El trabajo es grupal, 2 o 3 estudiantes (altamente recomendado que sean 3).

- Se deben entregar tres archivos “proyecto1.c”, “proyecto2.c”, y “proyecto3.c”, y se deben subir a Mediación Virtual. Puede añadir un archivo README si la compilación necesitara instrucciones especiales.

- No debe entregar el ejecutable compilado (el profesor lo compilará manualmente), entregue sólo el código .c.

- El código debe venir bien documentado.

- Cada estudiante debe entregar una autoevaluación del grupo (se asigna nota a cada integrante), la cual tendrá un peso del 40% de la nota. Esta autoevaluación debe ser añadida como comentario en la entrega de Mediación Virtual.

- Use un encabezado al inicio del programa para dar una pequeña descripción del programa, incluyendo su nombre.

- Se penalizará con un 40% de la nota si no se respetan todos estos lineamientos.

Parte 1: Números primos (36%)

Cree un programa que descomponga cualquier número entero dando sus divisores primos, con sus respectivas potencias. El formato para la descomposición debe ser N: (d1, p1), (d2, p2), …, (dm, pm); donde N es el número por descomponer, d es un divisor, y p es una potencia.

Ejemplo (no incluya los comentarios con #):

16: (2, 4) # 2*2*2*2 = 16

15: (3, 1), (5, 1) # 3*5=15

120: (2, 3), (3, 1), (5, 1) # 2*2*2*3*5 = 120

Consejo: primero empiece por crear un programa que calcule los números primos hasta cualquier N.

Aspectos a evaluar (los puntos suman 100, que equivalen al 36%):

1. (10 pts) El programa debe recibir el número N por la línea de comandos.

2. (80 pt) El programa imprime exitosamente la descomposición para cualquier número N.

3. (10 pt) El programa reconoce estos errores y da un mensaje de error en consola:

Más argumentos de la cuenta (sólo se puede recibir uno).

Menos argumentos de la cuenta.

Puede asumir que solamente se usarán números enteros como entrada.

Parte 2: Ángulos (36%)

Cree un programa que reciba mediante **LA LÍNEA DE COMANDOS** tres números (flotantes) que representan los tres lados de un triángulo. El programa debe encontrar los tres ángulos de dicho triángulo (**en grados**), o bien dar un mensaje de error si no se puede formar un triángulo con esos tres lados.

Nota: repase la ley de cosenos. Los triángulos no tienen naturaleza en particular (no asuma que son triángulos rectángulos).

Ejemplo:

Lados: 3, 4, 5

Ángulos: 53.13, 36.87, 90

Aspectos a evaluar (los puntos suman 100, que equivalen al 36%):

1. (10 pts) El programa recibe los tres números a través de línea de comandos.

2. (70 pts) El programa imprime exitosamente los ángulos.

3. (10 pts) El programa reconoce si el triángulo es válido o no.

4. (10 pt) El programa reconoce estos errores y da un mensaje de error en consola:

Número de argumentos inválido.

Puede asumir que solamente se usarán números flotantes como entrada.

Parte 3: Memoria Dinámica (28%)

El reto de este ejercicio es poder crear una función a partir de una descripción de texto, que cumpla con cada requerimiento. Los aspectos de implementación y formato se dejan a la libre. **El estudiante debe asumir cosas**, y recibirá poca ayuda del profesor adrede.

Cree una función que reciba del usuario una cantidad arbitraria de números **enteros**, que representen las posiciones dentro de una matrix NxM.

La función debe reservar memoria dinámicamente para colocar cada uno de los números ingresados por el usuario.

La función debe devolver al usuario las direcciones de memoria de los números que inician cada FILA.

La función retorna un número entero: 0 si no hay errores, -1 si hubo algún error.

Cree un programa principal de prueba (que usará el profesor) que llame a dicha función, donde se demuestre el correcto funcionamiento de la misma.

No olvide liberar la memoria utilizada.