**Taller #3**

1. ¿Qué es GNU Make?

GNU Make es una herramienta de construcción de software ampliamente utilizada en sistemas Unix y Linux, así como en otros entornos. Fue desarrollada como parte del proyecto GNU y se utiliza para automatizar el proceso de compilación y construcción de proyectos de software. Su función principal es gestionar las dependencias entre archivos fuente y archivos objeto, de manera que solo se vuelvan a compilar los archivos que han cambiado desde la última compilación.

1. ¿Cuáles son los componentes más importantes de un archivo Makefile?

**Objetivos:** Los objetivos son los elementos fundamentales de un Makefile.

Reglas: Las reglas son las instrucciones que le dicen a Make cómo crear un objetivo específico.

**Variables:** Las variables son valores que se pueden utilizar en el archivo Makefile para simplificar y parametrizar las reglas y comandos. Pueden definirse utilizando el operador = o := y se accede a ellas utilizando la sintaxis $(nombre\_variable).

**Directivas de Make:** Las directivas de Make son comandos especiales que controlan el comportamiento de Make.

**Reglas implícitas:** Make incluye reglas implícitas para ciertas extensiones de archivo (por ejemplo, .c a .o para archivos fuente C a archivos objeto).

1. ¿Cómo se define (asignaciones) y utilizan los macros dentro de un Makefile?

Puedes definir macros en un Makefile utilizando el operador de asignación = o :=. La diferencia entre ellos radica en cuándo se evalúan. El operador = evalúa la variable cada vez que se usa, mientras que := evalúa la variable solo una vez y almacena el resultado.

# Usando el operador de asignación =

CC = gcc

CFLAGS = -Wall -O2

# Usando el operador de asignación :=

SOURCES := main.c util.c

1. ¿Qué utilidad tienen los macros que hacen referencia a herramientas del toolchain?

**Portabilidad:** Al utilizar macros para referirse a herramientas del toolchain, puedes hacer que tu Makefile sea más portátil, ya que los nombres de las herramientas pueden variar entre diferentes sistemas operativos y configuraciones.

**Facilidad de configuración**: Los macros que hacen referencia a herramientas del toolchain hacen que sea fácil cambiar las herramientas utilizadas sin tener que modificar manualmente todas las reglas y comandos en el Makefile.

**Consistencia:** Al utilizar macros para las herramientas del toolchain, garantizas que todas las reglas y comandos utilicen las mismas herramientas en todo el proyecto, lo que ayuda a mantener la consistencia y evitar problemas de construcción causados por la discrepancia en las herramientas utilizadas.

1. ¿Por qué es poco efectivo establecer un Makefile como si fuese un script?

Los Makefiles son herramientas poderosas para automatizar la construcción de software, gestionar dependencias y proporcionar flexibilidad. Tratar un Makefile como un simple script lineal limita su efectividad y puede llevar a problemas de mantenimiento, eficiencia y portabilidad en proyectos más grandes y complejos.

**Parte Práctica**

![A screenshot of a computer program

Description automatically generated]()

![A screenshot of a computer

Description automatically generated]()

![A screenshot of a computer program

Description automatically generated]()