Ernesto Ulate Ramírez - 2014092260

1. ¿Qué es un GNU Make?

Es una herramienta para la automatización del proceso de compilación de programas y bibliotecas dirigidos por un código fuente en un archivo llamado Makefile. Es ampliamente utilizada en entornos Unix/Linux.

La utilidad make determina qué fragmentos del código necesitan ser compiladas.

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

Reglas de un makefile: Serie de comandos que le indican a make cómo construir un archivo objetivo. Se suelen componer de tres partes (objetivo, comandos y dependencias).

Objetivo:

Usualmente el nombre de archivo de salida o alguna acción específica. No comienzan con tabulación o espacios.

Comandos:

Empiezan con 2 tabulaciones y son la acción que lleva a cabo la regla.

Pre-requisitos o dependencias:

Son objetivos que necesitan ser ejecutados antes del objetivo en sí mismo.

Asignaciones:

Se escriben en mayúscula por convención. Ej: SALUDO = "hola"

Comentarios:

Para documentación del archivo. Ej: # Comentario

3. ¿Cómo se definen (asignaciones) y utilizan los marcos dentro de un Makefile? Brinde un ejemplo.

Los macros se definen como las asignaciones. Ej: OBJS = data.o main.o io.o

Para acceder al contenido del macro, se debe utilizar así: $(OBJS)

Ejemplo de utilización:

```
OBJS = data.o main.o

ejecutable: $(OBJS)
	gcc $(OBJS) -o ejecutable

data.o: data.c data.h
	gcc -c data.c

main.o: main.c main.h
	gcc -c main.c
```

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

Nombrar diferentes herramientas mediante macros podría facilitar el trabajo si se necesita cambiar el compilador. Al hacerlo mediante macros, no será necesario ir sobre cada línea del archivo Makefile cambiando los datos, con solo cambiar el valor del macro el trabajo está hecho.

También, gracias a los macros, es posible cambiarlos desde el shell para mayor flexibilidad. Ej: make 'MACRO=nuevo_valor'

Para sobrescribir el valor de MACRO que se estableció en el Makefile.