Sesión 1: Contacto C

Hoja de problemas

Programación para Sistemas

Ángel Herranz
aherranz@fi.upm.es
Universidad Politécnica de Madrid

2021-2022

Ejercicio 1. Repasa las transparencias de clase.

Ejercicio 2. Quizás ya te hayas dado cuenta de que en las transparencias y en las hojas de ejercicios de vez en cuando aparecen algunos iconos. Aquí tienes un pequeño diccionario:

- *peligro, atención*
- *pedir ayuda*
- *leer, convenciones*
- *recordar*
- *navegar, buscar en internet*
- *éxito*
- *buscar*
- *fracaso*
- *programar*
- *en casa*
- *tutorial*
- *tiempo*
- *pensar, observar*
- *pausa*
- *adelantarse*

Ejercicio 3. Busca en internet qué es GNU y qué es la FSF.

Ejercicio 4. Tu primera tarea es ser capaz de compilar y ejecutar un *hola mundo* en C. Para ello, necesitas un editor de texto que ya deberías tener instalado y un compilador de C.

Lo más normal es que instales el compilador de C de GNU: GCC. En Ubuntu, el paquete a instalar es `gcc`. Puedes instalarlo con la siguiente línea de comandos en tu *shell*:

```
$ sudo apt-get install gcc
```

Pero te recomiendo instalar un paquete que además te va a instalar Make, una herramienta que usaremos de forma masiva. El paquete en cuestión es `build-essential`:

```
$ sudo apt-get install build-essential
```

Ejercicio 5. Observa con atención cada uno de los mensajes que puedes leer cuando ejecutas una línea de comando. En el caso anterior, si lo repasas, podrás ver que al instalar un *paquete* también se instalan otros de los que el primero **depende**.

Ejercicio 6. Ya deberías tener instalado el compilador de C. Un compilador no deja de ser un programa. Y un programa no deja de ser un fichero. Todos los ficheros tienen un nombre que dice en qué sitio del disco vive. El nombre *absoluto* del compilador de C suele ser `/usr/bin/gcc` aunque en tu instalación puede ser otro. Vamos a preguntar al sistema operativo a través de nuestro *shell* (Bash):

- Ejecuta en la línea de comandos `which gcc`
- Ejecuta en la línea de comandos `which cc` (a veces el compilador de C se llama simplemente `cc`)
- ¿Qué hace `which`? (usa `man which`)
- Ejecuta en la línea de comandos `ls -l /usr/bin/gcc`

Ejercicio 7. Ahora deberías estar en condiciones de seguir las instrucciones de las transparencias:

1. Crea un directorio en el que dejar el trabajo de la asignatura, `clases_pps` por ejemplo:

   ```
   $ mkdir clases_pps
   ```

2. Entra en el directorio:

   ```
   $ cd clases_pps
   ```

3. Crea el fichero `hola.c` con tu editor de texto favorito.
4. Comprueba los ficheros que hay en tu directorio:

   ```
   $ ls -l
   ```

5. Compila:

   ```
   $ gcc hola.c
   ```

6. Comprueba los ficheros que hay en tu directorio:

   ```
   $ ls -l
   ```

7. Ejecuta:

   ```
   $ ./a.out
   ```

Ejercicio 8. ¿Te cuadran las transparencias? ¿Entiendes el proceso de compilación?

Ejercicio 9. Dale un nombre más razonable al ejecutable, en vez de `a.out`, haz que se llame `hola` (usa el programa `mv`, para saber cómo puedes pedir ayuda al manual).

Ejercicio 10. No permitas que `gcc` te oculte lo que está haciendo internamente. Compila primero y *linka* después. Deberás ver el fichero *objeto* intermedio: `hola.o`.

Ejercicio 11. Recuerda que tienes un disponible el manual de GCC con la línea

```
$ man gcc
```

Ejercicio 12. No permitas que `gcc` te oculte nada de lo que está haciendo internamente. Descubre cuál es el significado de **`#include`** usando la opción `-E` del compilador.

Ejercicio 13. ¿Qué crees que está pasando? ¿Puedes encontrar el fichero `stdio.h`? Quizás puedes buscarlo con `locate`.

`#include` simplemente expande, como si fuera un Copy & Paste el fichero nombrado, en este caso `stdio.h`. Nada más.

Ejercicio 14. Ejecuta el manual de `gcc` o busca en internet hasta descubrir de qué forma podrías generar un fichero con código ensamblador a partir del fichero `hola.c`.