Práctica con el compilador `gcc` –Biblioteca estática–

Informática II – UTN-FRC

Gonzalo F. Perez Paina

1. Enunciado

Escribir y construir una biblioteca estática `libhola.a` que permita compilar la aplicación del listado 1. Esta biblioteca debe implementar las funciones `hola()` y `adios()` llamadas en el código fuente de `saludo.c`, las cuales deberán estar separadas en dos archivos fuentes diferentes. Además, se debe codificar un archivo de cabecera `hola.h` que declare los prototipos de dichas funciones. La salida de la aplicación se muestra en el listado 2.

Listing 1: Archivo `saludo.c`. Aplicación de prueba.

```c
#include "hola.h"

int main(void)
{
  hola("Info-II");
  adios();
  return 0;
}
```

Listing 2: Ejecución de la aplicación.

```
./saludo
Hola, Info-II!
Adios!
```

2. Implementación de la biblioteca

Se muestra la implementación de la biblioteca antes enunciada codificada en dos archivos fuentes `.c` y un archivo de cabecera `.h`. Luego, se muestra como construir la biblioteca estática `libhola.a`.

2.1. Código fuente

El listado 3 y 4 muestra los archivos fuentes (`.c`) donde están implementadas las funciones `hola()` y `adios()`, respectivamente. Ambos archivos fuentes incluyen el archivo de cabecera `stdio.h` dado que utilizan la función de salida estándar `printf()`. Además, incluyen el archivo `hola.h` donde están los prototipos de las funciones implementadas por la biblioteca. Al utilizar la directiva de pre-procesador `#include` con `<>` se le indica que busque el archivo de cabecera en los directorios predefinidos del sistema, mientras que al utilizarla con `""` le indica que lo busque en el mismo directorio del archivo fuente, y a continuación en los directorios del sistema.

El archivo de cabecera `hola.h` incluye directivas del pre-procesador para la compilación condicional que evitan tener múltiples declaraciones de variables si dicho archivo de cabecera se incluye en más de un archivo fuente. En la primer inclusión del archivo `hola.h` al compilar un archivo `.c` al no estar todavía declarada la constante simbólica `_HOLA_H` la directiva `#ifndef` resulta verdadero por lo que se incluye

Listing 3: Archivo `hola_fn.c`. Implementación de función `hola`.

```c
#include <stdio.h>
#include "hola.h"

void hola(const char * nombre)
{
  printf("Hola, %s!\n", nombre);
}
```

Listing 4: Archivo `adios_fn.c`. Implementación de función `adios`.

```c
#include <stdio.h>
#include "hola.h"

void adios(void)
{
  printf("Adios!\n");
}
```

el cuerpo del archivo de cabecera y se declara dicha constante. De esta forma, dicha constante ya se encuentra declarada al momento de compilar otro archivo `.c` que incluya este archivo de cabecera.

Listing 5: Archivo de cabecera `hola.h`.

```c
#ifndef _HOLA_H
#define _HOLA_H

void hola(const char * );
void adios(void);

#endif
```

2.2. Construcción

El primer paso para la construcción de la biblioteca es compilar el código fuente de los módulos que la componen, en este caso los archivos fuentes `hola_fn.c` y `adios_fn.c` de los listados 3 y 4. El listado 6 muestra la etapa de compilación. La opción `-c` se utiliza para compilar el código fuente `.c` y generar un archivo objeto `.o`.

Listing 6: Compilar código fuente de archivos de biblioteca.

```
> gcc -Wall -c hola_fn.c
> gcc -Wall -c adios_fn.c
```

El listado 7 muestra la utilización de la aplicación (comando) `ar` para empaquetar los archivos objetos antes compilados y generar la biblioteca estática `libhola.a`. La opción `c` crea el archivo `libhola.a`, mientras que la opción `r` inserta el archivo objeto dentro de la biblioteca y lo reemplaza si ya está incluido.

Listing 7: Crear biblioteca estática.

```
> ar cr libhola.a hola_fn.o adios_fn.o
```

De forma separada a la construcción de la biblioteca estática, se puede verificar el correcto funcionamiento del código fuente escrito mediante los comandos indicados en el listado 8. Aquí, primero se compila el código fuente de la aplicación (el archivo `.c` donde está la función `main()`) y luego se enlazan

todos los archivos objetos para generar el binario de nombre `saludo`. El listado 2 muestra la salida al ejecutar la aplicación construida.

Listing 8: Compilar código fuente de aplicación y enlazar.

```
> gcc -Wall -c saludo.c
> gcc hola_fn.o adios_fn.o saludo.o -o saludo
```

3. Uso de la biblioteca

Teniendo el archivo objeto de la aplicación `saludo.o` y la biblioteca `libhola.a` generados, se puede obtener el binario final mediante el comando del listado 9. Aquí, la opción `-lhola` es la que indica enlazar la aplicación con la biblioteca `libhola.a`. El formato de la opción `-l` se utiliza como `-lNOMBRE` donde `NOMBRE` es el nombre de la biblioteca sin el prefijo `lib`, como `libNOMBRE.a`.

Listing 9: Uso de la biblioteca.

```
> gcc saludo.o -lhola -o saludo
```

3.1. Opciones adicionales de compilación y enlazado

A continuación se muestra como utilizar la opción `-I` para indicar donde se encuentra el archivo de cabecera al compilar un archivo fuente, si no comparten el mismo directorio; y también como utilizar la opción `-L` para indicar donde se encuentra la biblioteca.

El listado 10 indica los comandos para eliminar el archivo objeto de la aplicación y la aplicación construida anteriormente. Crear además los directorios `header` y `lib` para luego mover el archivo de cabecera de la biblioteca y el archivo de biblioteca a estos directorios.

Listing 10: Pruebas.

```
> rm saludo.o saludo
> mkdir header
> mv hola.h header
> mkdir lib
> mv libhola.a lib
```

El listado 11 muestra el error de compilación al compilar (con la opción `-c`) el código fuente de la aplicación cuando se ha movido de directorio el archivo de cabecera `hola.h`. El listado 12 muestra como incluir la ruta donde se encuentra el archivo de cabecera utilizando la opción `-I` y salvar el error anterior.

Listing 11: Pruebas.

```
> gcc -Wall -c saludo.c
saludo.c:1:18: fatal error: hola.h: No such file or directory
compilation terminated.
```

Listing 12: Pruebas.

```
> gcc -Wall -c -I./header saludo.c
```

El listado 13 muestra el error de enlazado que se produce al intentar generar la aplicación y no encontrar el archivo de biblioteca `libhola.a` el cual fue movido a otro directorio. El listado 14 muestra

como incluir la ruta donde se encuentra el archivo de biblioteca utilizando la opción `-L` y salvar el error anterior.

Listing 13: Pruebas.

```
> gcc saludo.o -lhola -o saludo
/usr/bin/ld: cannot find -lhola
collect2: error: ld returned 1 exit status
```

Listing 14: Pruebas.

```
> gcc saludo.o -L./lib -lhola -o saludo
```