

# **Tarea 5 de la Unidad 2: Herramientas de desarrollo**

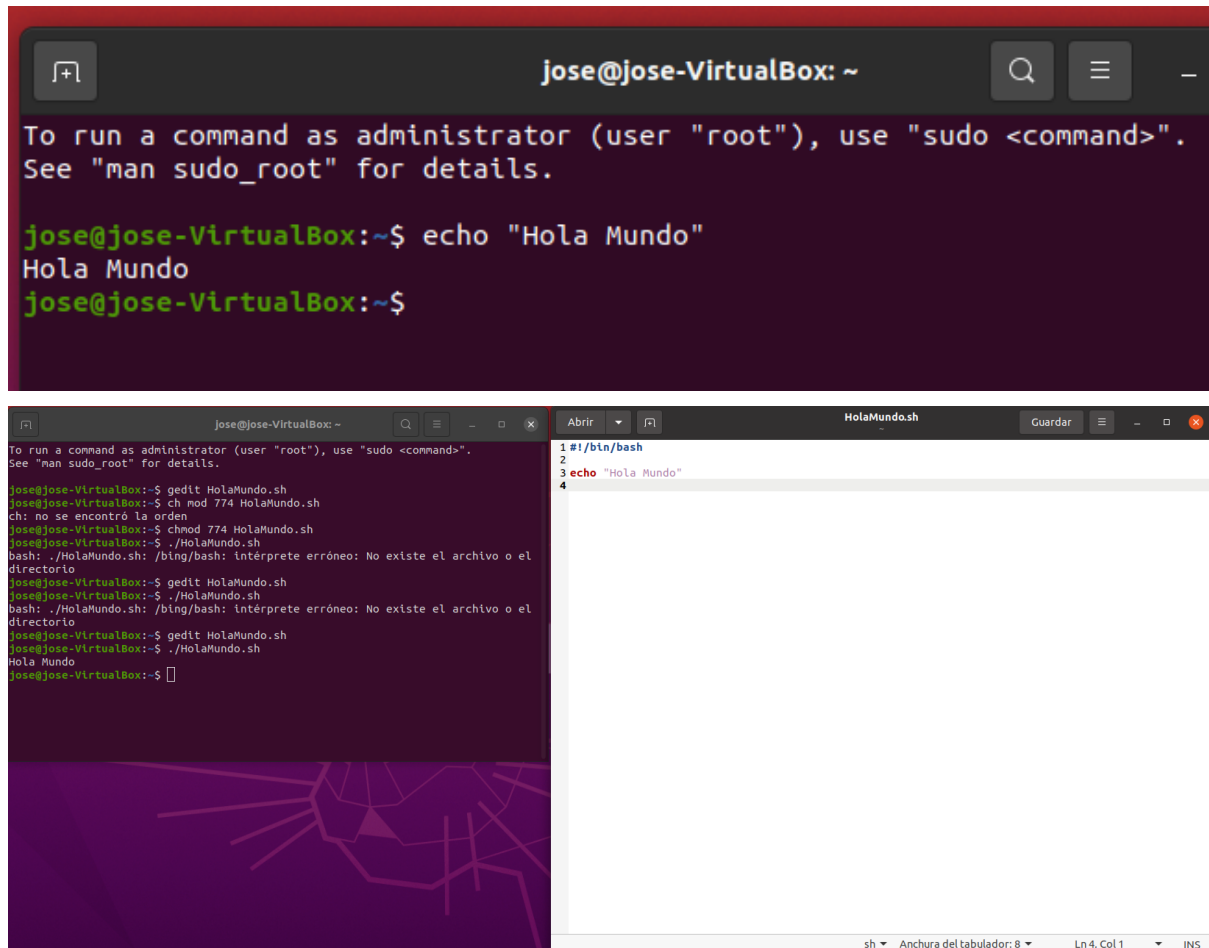
**José Antonio Ramos Molina**

[https://github.com/jrammol/ED\\_CARRILLO\\_22.git](https://github.com/jrammol/ED_CARRILLO_22.git)

# 1. Ejecuta el programa 'Hola mundo' en los siguientes lenguajes:

## Bash

### Ejecución y Script



```
jose@jose-VirtualBox: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
jose@jose-VirtualBox:~$ echo "Hola Mundo"  
Hola Mundo  
jose@jose-VirtualBox:~$  
  
jose@jose-VirtualBox:~$ gedit HolaMundo.sh  
jose@jose-VirtualBox:~$ ch mod 774 HolaMundo.sh  
ch: no se encontró la orden  
jose@jose-VirtualBox:~$ chmod 774 HolaMundo.sh  
jose@jose-VirtualBox:~$ ./HolaMundo.sh  
bash: ./HolaMundo.sh: /bin/bash: intérprete erróneo: No existe el archivo o el directorio  
jose@jose-VirtualBox:~$ gedit HolaMundo.sh  
jose@jose-VirtualBox:~$ ./HolaMundo.sh  
bash: ./HolaMundo.sh: /bin/bash: intérprete erróneo: No existe el archivo o el directorio  
jose@jose-VirtualBox:~$ gedit HolaMundo.sh  
jose@jose-VirtualBox:~$ ./HolaMundo.sh  
Hola Mundo  
jose@jose-VirtualBox:~$
```

```
1#!/bin/bash  
2  
3echo "Hola Mundo"  
4
```

## Python

Instalación de paquete python

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install python
[sudo] contraseña para joseramos:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Nota, seleccionando «python-is-python2» en lugar de «python»
Se instalarán los siguientes paquetes adicionales:
  libpython2-stdlib libpython2.7-minimal libpython2.7-stdlib python2
  python2-minimal python2.7 python2.7-minimal
Paquetes sugeridos:
  python2-doc python-tk python2.7-doc binutils binfmt-support
Se instalarán los siguientes paquetes NUEVOS:
  libpython2-stdlib libpython2.7-minimal libpython2.7-stdlib python-is-python2
  python2 python2-minimal python2.7 python2.7-minimal
0 actualizados, 8 nuevos se instalarán, 0 para eliminar y 151 no actualizados.
Se necesita descargar 3.815 kB de archivos.
Se utilizarán 16,5 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libpython
2.7-minimal amd64 2.7.18-1~20.04.3 [336 kB]
```

Ejecución

```
joseramos@joseramos-VirtualBox:~$ python
Python 2.7.18 (default, Jul 1 2022, 12:27:04)
[GCC 9.4.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print "Hola Mundo"
Hola Mundo
>>>
```

Script ejecutable

```
Abrir ▼ [icon] hola.py Guardar [icon] -
1 #!/usr/bin/env python
2
3 print "Hola Mundo"
```

```
joseramos@joseramos-VirtualBox:~$ gedit hola.py
joseramos@joseramos-VirtualBox:~$ chmod 774 hola.py
joseramos@joseramos-VirtualBox:~$ ./hola.py
Hola Mundo
joseramos@joseramos-VirtualBox:~$
```

## PHP

### Instalación paquete PHP

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install php
[sudo] contraseña para joseramos:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  apache2 apache2-bin apache2-data apache2-utils libapache2-mod-php7.4 libapr1
  libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0 php-common
  php7.4 php7.4-cli php7.4-common php7.4-json php7.4-openssl php7.4-readline
Paquetes sugeridos:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom php-pear
Se instalarán los siguientes paquetes NUEVOS:
  apache2 apache2-bin apache2-data apache2-utils libapache2-mod-php7.4 libapr1
  libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0 php
  php-common php7.4 php7.4-cli php7.4-common php7.4-json php7.4-openssl
  php7.4-readline
0 actualizados, 18 nuevos se instalarán, 0 para eliminar y 151 no actualizados.
Se necesita descargar 5.851 kB de archivos.
Se utilizarán 25,9 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal/main amd64 libapr1 amd64 1.6.5-1ubuntu1 [91,4 kB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal/main amd64 libaprutil1 amd64 1.6.1-4ubuntu2 [84,7 kB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.1-4ubuntu2 [10,5 kB]
Des:4 http://es.archive.ubuntu.com/ubuntu focal/main amd64 libaprutil1-ldap amd64 1.6.1-4ubuntu2 [8.736 B]
Des:5 http://es.archive.ubuntu.com/ubuntu focal/main amd64 liblua5.2-0 amd64 5.2.4-1.1build3 [106 kB]
```

### Ejecución

```
joseramos@joseramos-VirtualBox:~$ php -a
Interactive mode enabled

php > echo "Hola mundo\n";
Hola mundo
php > █
```

### Script ejecutable

```

Abrir  ▼  [+]  hola.php  Guardar  ≡  -  □  ×

1 #!/usr/bin/env php
2
3 <?php
4     echo "Hola mundo\n"
5     ?>
6
```

```
php > ^Djoseramos@joseramos-VirtualBox:~$ gedit hola.php
joseramos@joseramos-VirtualBox:~$ chmod 774 hola.php
joseramos@joseramos-VirtualBox:~$ ./hola.php

Hola mundo

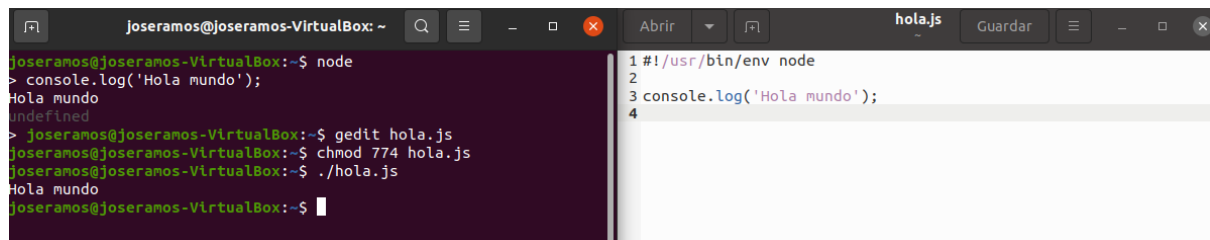
joseramos@joseramos-VirtualBox:~$
```

## Javascript

### Instalación paquete

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install nodejs
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libc-ares2 libnode64 nodejs-doc
Paquetes sugeridos:
  npm
Se instalarán los siguientes paquetes NUEVOS:
  libc-ares2 libnode64 nodejs nodejs-doc
0 actualizados, 4 nuevos se instalarán, 0 para eliminar y 151 no
actualizados.
Se necesita descargar 6.807 kB de archivos.
Se utilizarán 30,7 MB de espacio de disco adicional después de es
ta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 libc-ares2 amd64 1:1.5-0ubuntu0.1 [38.2 kB]
```

### Ejecución y Script ejecutable



The screenshot shows a terminal window on the left and a code editor on the right. The terminal window displays the following commands and output:

```
joseramos@joseramos-VirtualBox:~$ node
> console.log('Hola mundo');
Hola mundo
undefined
> joseramos@joseramos-VirtualBox:~$ gedit hola.js
joseramos@joseramos-VirtualBox:~$ chmod 774 hola.js
joseramos@joseramos-VirtualBox:~$ ./hola.js
Hola mundo
joseramos@joseramos-VirtualBox:~$
```

The code editor on the right shows the content of the file `hola.js`:

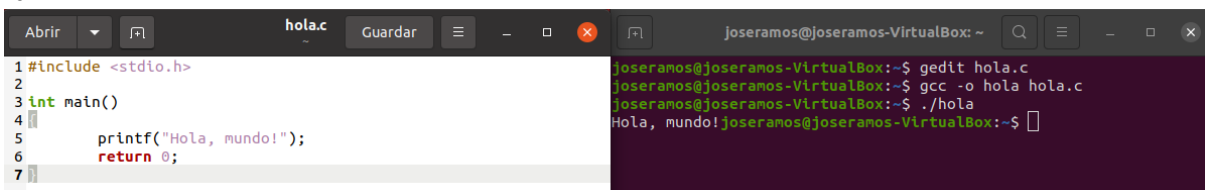
```
1 #!/usr/bin/env node
2
3 console.log('Hola mundo');
4
```

## C

### Instalación paquete C

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install gcc
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  binutils binutils-common binutils-x86-64-linux-gnu gcc-9
  libasan5 libatomic1 libbinutils libc-dev-bin libc6-dev
  libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev libitm1
  liblsan0 libquadmath0 libtsan0 libubsan1 linux-libc-dev
  manpages-dev
Paquetes sugeridos:
  binutils-doc gcc-multilib make autoconf automake libtool flex
  bison gcc-doc gcc-9-multilib gcc-9-doc gcc-9-locales
  glibc-doc
Se instalarán los siguientes paquetes NUEVOS:
  binutils binutils-common binutils-x86-64-linux-gnu gcc gcc-9
  libasan5 libatomic1 libbinutils libc-dev-bin libc6-dev
  libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev libitm1
  liblsan0 libquadmath0 libtsan0 libubsan1 linux-libc-dev
  manpages-dev
0 actualizados, 21 nuevos se instalarán, 0 para eliminar y 151 no
actualizados.
Se necesita descargar 25,7 MB de archivos.
Se utilizarán 120 MB de espacio de disco adicional después de esta
operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 binutils-common amd64 2.34-6ubuntu1.4 [207 kB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 libbinutils amd64 2.34-6ubuntu1.4 [474 kB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 libctf-nobfd0 amd64 2.34-6ubuntu1.4 [47,2 kB]
```

### Ejecución



The image shows a code editor window titled 'hola.c' with the following code:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("Hola, mundo!");
6     return 0;
7 }
```

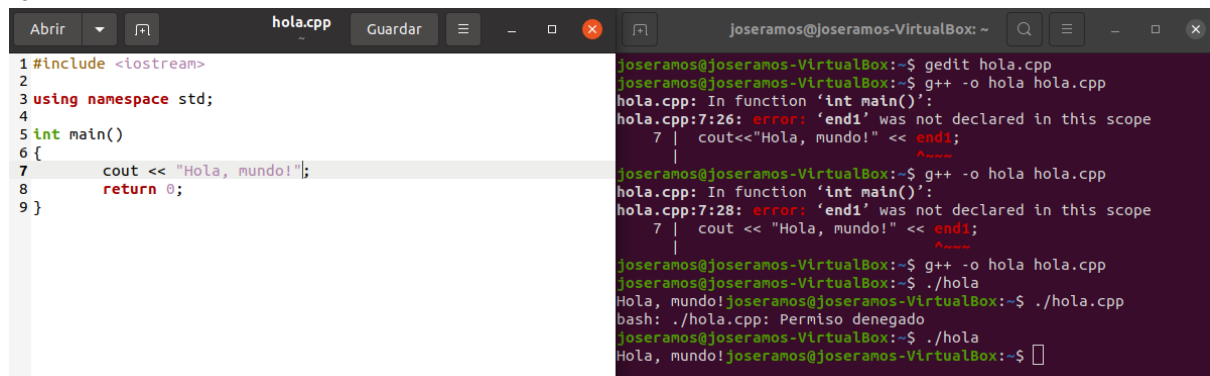
Below the code editor is a terminal window showing the execution of the program:

```
joseramos@joseramos-VirtualBox:~$ gedit hola.c
joseramos@joseramos-VirtualBox:~$ gcc -o hola hola.c
joseramos@joseramos-VirtualBox:~$ ./hola
Hola, mundo!joseramos@joseramos-VirtualBox:~$
```

## C++

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install g++
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  g++-9 libstdc++-9-dev
Paquetes sugeridos:
  g++-multilib g++-9-multilib gcc-9-doc libstdc++-9-doc
Se instalarán los siguientes paquetes NUEVOS:
  g++ g++-9 libstdc++-9-dev
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 151 no
actualizados.
Se necesita descargar 10,1 MB de archivos.
Se utilizarán 46,8 MB de espacio de disco adicional después de e
ta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd
4 libstdc++-9-dev amd64 9.4.0-1ubuntu1~20.04.1 [1.722 kB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd
4 g++-9 amd64 9.4.0-1ubuntu1~20.04.1 [8.420 kB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal/main amd64 g++ a
d64 4:9.3.0-1ubuntu2 [1.604 B]
```

## Ejecución



```
hola.cpp  Guardar  joseramos@joseramos-VirtualBox: ~
1 #include <iostream>
2
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hola, mundo!";
8     return 0;
9 }

joseramos@joseramos-VirtualBox:~$ gedit hola.cpp
joseramos@joseramos-VirtualBox:~$ g++ -o hola hola.cpp
hola.cpp: In function 'int main()':
hola.cpp:7:26: error: 'endl' was not declared in this scope
7 | cout<<"Hola, mundo!" << endl;
  |                          ^~~~~
joseramos@joseramos-VirtualBox:~$ g++ -o hola hola.cpp
hola.cpp: In function 'int main()':
hola.cpp:7:28: error: 'endl' was not declared in this scope
7 | cout << "Hola, mundo!" << endl;
  |                            ^~~~~
joseramos@joseramos-VirtualBox:~$ g++ -o hola hola.cpp
joseramos@joseramos-VirtualBox:~$ ./hola
Hola, mundo!joseramos@joseramos-VirtualBox:~$ ./hola.cpp
bash: ./hola.cpp: Permiso denegado
joseramos@joseramos-VirtualBox:~$ ./hola
Hola, mundo!joseramos@joseramos-VirtualBox:~$
```

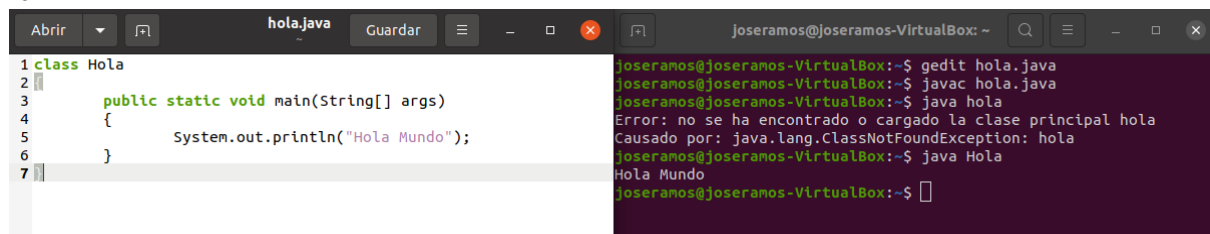


## Java

### Instalación paquete

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install default-jdk
[sudo] contraseña para joseramos:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  ca-certificates-java default-jdk-headless default-jre default-jre-headless fonts-dejavu-extra java-common
  libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev
  libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre
  openjdk-11-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Paquetes sugeridos:
  libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source visualvm
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei
Se instalarán los siguientes paquetes NUEVOS:
  ca-certificates-java default-jdk default-jdk-headless default-jre default-jre-headless fonts-dejavu-extra
  java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev
  libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre
  openjdk-11-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 actualizados, 25 nuevos se instalarán, 0 para eliminar y 151 no actualizados.
Se necesita descargar 267 MB de archivos.
Se utilizarán 422 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal/main amd64 java-common all 0.72 [6.816 B]
Des:2 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 openjdk-11-jre-headless amd64 11.0.17+8-1ubuntu
2~20.04 [37,4 MB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal/main amd64 default-jre-headless amd64 2:1.11-72 [3.192 B]
Des:4 http://es.archive.ubuntu.com/ubuntu focal/main amd64 ca-certificates-java all 20190405ubuntu1 [12,2 kB]
Des:5 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 openjdk-11-jre amd64 11.0.17+8-1ubuntu2~20.04 [
175 kB]
Des:6 http://es.archive.ubuntu.com/ubuntu focal/main amd64 default-jre amd64 2:1.11-72 [1.084 B]
Des:7 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 openjdk-11-jdk-headless amd64 11.0.17+8-1ubuntu
2~20.04 [224 MB]
```

### Ejecución

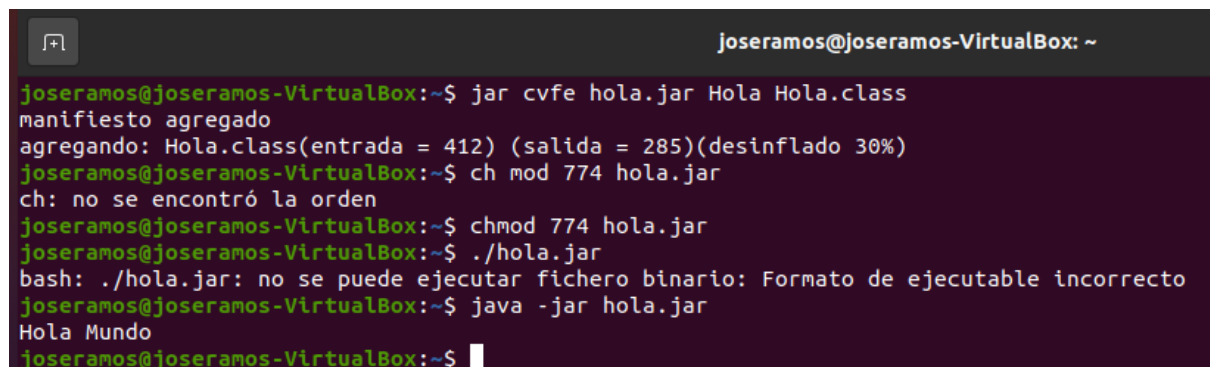


```
hola.java  Guardar
1 class Hola
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hola Mundo");
6     }
7 }
```

```
joseramos@joseramos-VirtualBox:~$ gedit hola.java
joseramos@joseramos-VirtualBox:~$ javac hola.java
joseramos@joseramos-VirtualBox:~$ java hola
Error: no se ha encontrado o cargado la clase principal hola
Causado por: java.lang.ClassNotFoundException: hola
joseramos@joseramos-VirtualBox:~$ java Hola
Hola Mundo
joseramos@joseramos-VirtualBox:~$
```

(malditas mayúsculas)

### Script ejecutable



```
joseramos@joseramos-VirtualBox:~$ jar cvfe hola.jar Hola.class
manifiesto agregado
agregando: Hola.class(entrada = 412) (salida = 285)(desinflado 30%)
joseramos@joseramos-VirtualBox:~$ ch mod 774 hola.jar
ch: no se encontró la orden
joseramos@joseramos-VirtualBox:~$ chmod 774 hola.jar
joseramos@joseramos-VirtualBox:~$ ./hola.jar
bash: ./hola.jar: no se puede ejecutar fichero binario: Formato de ejecutable incorrecto
joseramos@joseramos-VirtualBox:~$ java -jar hola.jar
Hola Mundo
joseramos@joseramos-VirtualBox:~$
```

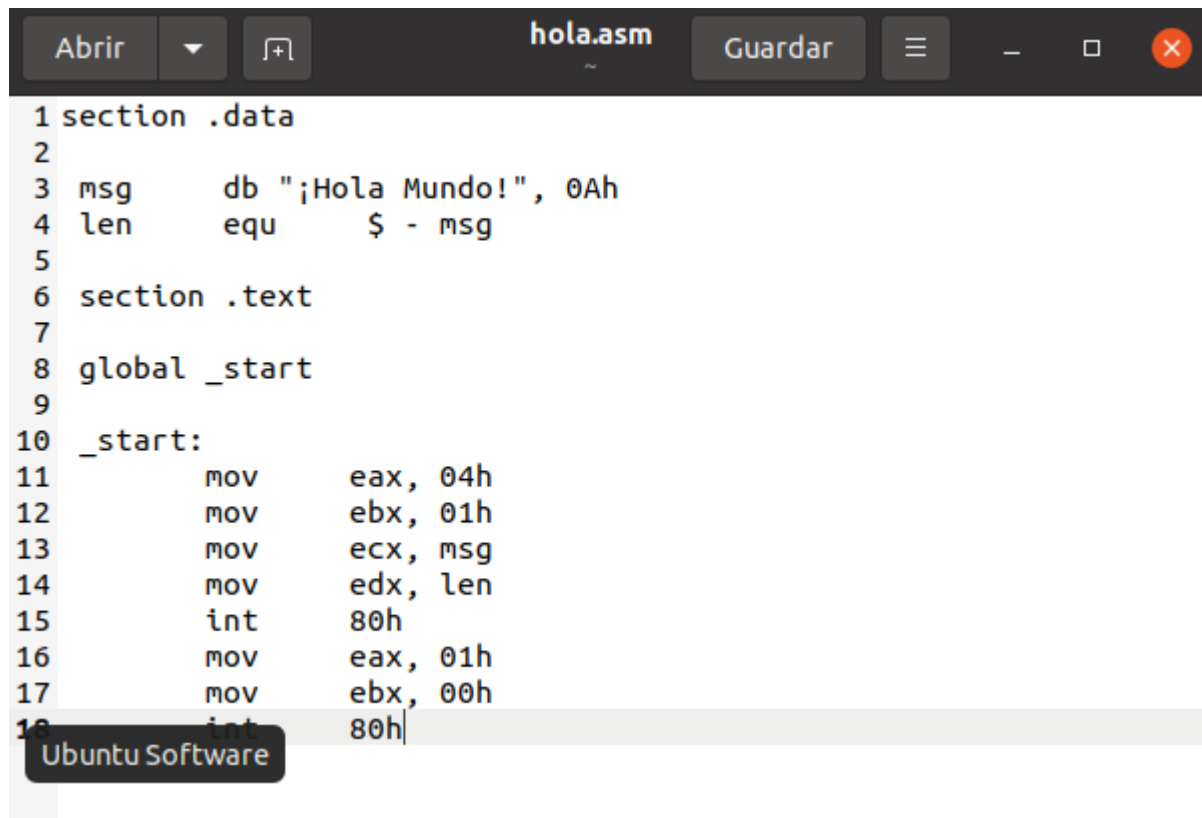


## Ensamblador

### Instalación paquete

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install nasm
[sudo] contraseña para joseramos:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  nasm
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 151 no actualizados.
Se necesita descargar 362 kB de archivos.
Se utilizarán 3.374 kB de espacio de disco adicional después de esta operación.
Des:1 http://es.archive.ubuntu.com/ubuntu focal/universe amd64 nasm amd64 2.14.02-1 [362 kB]
Descargados 362 kB en 0s (1.005 kB/s)
Seleccionando el paquete nasm previamente no seleccionado.
(Leyendo la base de datos ... 192049 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../nasm_2.14.02-1_amd64.deb ...
Desempaquetando nasm (2.14.02-1) ...
Configurando nasm (2.14.02-1) ...
Procesando disparadores para man-db (2.9.1-1) ...
joseramos@joseramos-VirtualBox:~$
```

### Ejecución



```
1 section .data
2
3 msg      db ";Hola Mundo!", 0Ah
4 len      equ    $ - msg
5
6 section .text
7
8 global _start
9
10 _start:
11         mov     eax, 04h
12         mov     ebx, 01h
13         mov     ecx, msg
14         mov     edx, len
15         int     80h
16         mov     eax, 01h
17         mov     ebx, 00h
18         int     80h
```

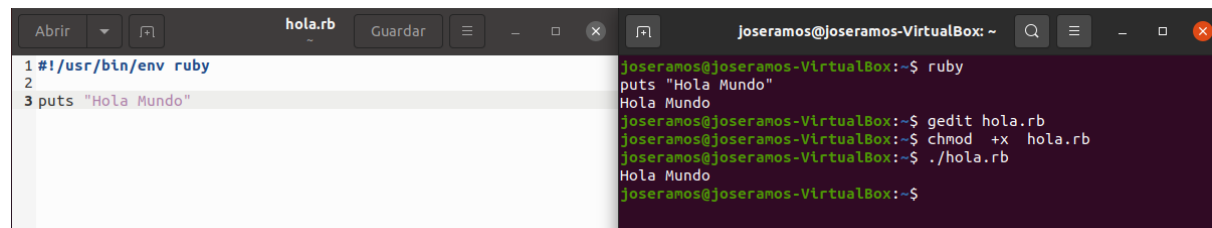
```
joseramos@joseramos-VirtualBox:~$ gedit hola.asm
joseramos@joseramos-VirtualBox:~$ nasm -f elf64 hola.asm
hola.asm:8: warning: label alone on a line without a colon might be in error [-w-orphan-labels]
joseramos@joseramos-VirtualBox:~$ nasm -f elf64 hola.asm
joseramos@joseramos-VirtualBox:~$ ld hola.o -o hola
joseramos@joseramos-VirtualBox:~$ ./hola
;Hola Mundo!
joseramos@joseramos-VirtualBox:~$
```

## Ruby

### Instalación del paquete

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install ruby
[sudo] contraseña para joseramos:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 fonts-lato javascript-common libjs-jquery libruby2.7 rake
 ruby-minitest ruby-net-telnet ruby-power-assert
 ruby-test-unit ruby-xmlrpc ruby2.7 rubygems-integration
Paquetes sugeridos:
 ri ruby-dev bundler
Se instalarán los siguientes paquetes NUEVOS:
 fonts-lato javascript-common libjs-jquery libruby2.7 rake
 ruby ruby-minitest ruby-net-telnet ruby-power-assert
 ruby-test-unit ruby-xmlrpc ruby2.7 rubygems-integration
0 actualizados, 13 nuevos se instalarán, 0 para eliminar y 151 no actualizados.
Se necesita descargar 6.896 kB de archivos.
Se utilizarán 31,5 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal/main amd64 fonts-lato all 2.0-2 [2.698 kB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal/main amd64 javascript-common all 11 [6.066 B]
Des:3 http://es.archive.ubuntu.com/ubuntu focal/main amd64 libjs-jquery all 3.3.1~dfsg-3 [329 kB]
```

### Ejecución y Script



```
1 #!/usr/bin/env ruby
2
3 puts "Hola Mundo"

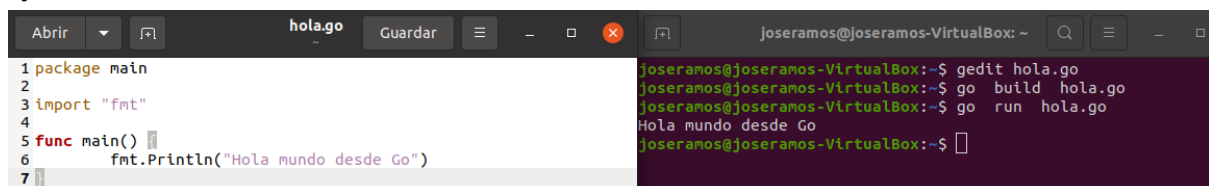
joseramos@joseramos-VirtualBox:~$ ruby
puts "Hola Mundo"
Hola Mundo
joseramos@joseramos-VirtualBox:~$ gedit hola.rb
joseramos@joseramos-VirtualBox:~$ chmod +x hola.rb
joseramos@joseramos-VirtualBox:~$ ./hola.rb
Hola Mundo
joseramos@joseramos-VirtualBox:~$
```

## Go

### Instalación paquete

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install golang
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  golang-1.13 golang-1.13-doc golang-1.13-go
  golang-1.13-race-detector-runtime golang-1.13-src golang-doc
  golang-go golang-race-detector-runtime golang-src
Paquetes sugeridos:
  bzr | brz git mercurial subversion
Se instalarán los siguientes paquetes NUEVOS:
  golang golang-1.13 golang-1.13-doc golang-1.13-go
  golang-1.13-race-detector-runtime golang-1.13-src golang-doc
  golang-go golang-race-detector-runtime golang-src
0 actualizados, 10 nuevos se instalarán, 0 para eliminar y 151 no
actualizados.
Se necesita descargar 63,5 MB de archivos.
Se utilizarán 329 MB de espacio de disco adicional después de esta
operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 golang-1.13-src amd64 1.13.8-1ubuntu1.1 [12,6 MB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 golang-1.13-go amd64 1.13.8-1ubuntu1.1 [47,6 MB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 golang-1.13-doc all 1.13.8-1ubuntu1.1 [2.525 kB]
Des:4 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd6
4 golang-1.13 all 1.13.8-1ubuntu1.1 [11,3 kB]
Des:5 http://es.archive.ubuntu.com/ubuntu focal/main amd64 golang
-src amd64 2:1.13~1ubuntu2 [4.044 B]
Des:6 http://es.archive.ubuntu.com/ubuntu focal/main amd64 golang
-go amd64 2:1.13~1ubuntu2 [22,0 kB]
```

### Ejecutamos



The screenshot shows a code editor window titled 'hola.go' with the following Go code:

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     fmt.Println("Hola mundo desde Go")
7 }
```

Next to the code editor is a terminal window titled 'joseramos@joseramos-VirtualBox: ~'. The terminal shows the following commands and output:

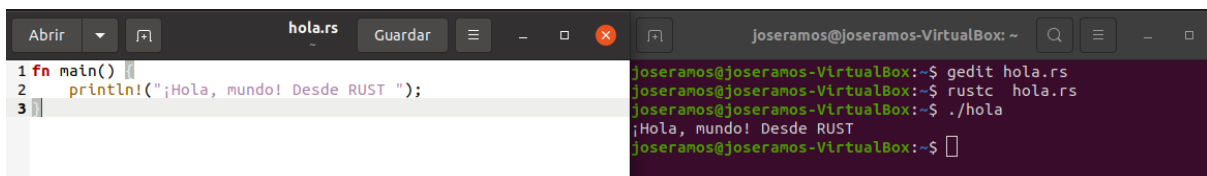
```
joseramos@joseramos-VirtualBox:~$ gedit hola.go
joseramos@joseramos-VirtualBox:~$ go build hola.go
joseramos@joseramos-VirtualBox:~$ go run hola.go
Hola mundo desde Go
joseramos@joseramos-VirtualBox:~$
```

## Rust

### Instalación de paquetes

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install rustc
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libstd-rust-1.61 libstd-rust-dev
Paquetes sugeridos:
  cargo llvm-14 lld-14 clang-14
Se instalarán los siguientes paquetes NUEVOS:
  libstd-rust-1.61 libstd-rust-dev rustc
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 151 no
actualizados.
Se necesita descargar 78,4 MB de archivos.
Se utilizarán 366 MB de espacio de disco adicional después de est
a operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/universe
amd64 libstd-rust-1.61 amd64 1.61.0+dfsg1~llvm-1~exp1ubuntu0.20.0
4.1 [40,6 MB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal-updates/universe
amd64 libstd-rust-dev amd64 1.61.0+dfsg1~llvm-1~exp1ubuntu0.20.04
.1 [34,7 MB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal-updates/universe
amd64 rustc amd64 1.61.0+dfsg1~llvm-1~exp1ubuntu0.20.04.1 [3.169
kB]
Descargados 78,4 MB en 50s (1.562 kB/s)
Seleccionando el paquete libstd-rust-1.61:amd64 previamente no se
leccionado.
```

### Ejecutamos



```
hola.rs
1 fn main() {
2     println!("¡Hola, mundo! Desde RUST ");
3 }

joseramos@joseramos-VirtualBox:~$ gedit hola.rs
joseramos@joseramos-VirtualBox:~$ rustc hola.rs
joseramos@joseramos-VirtualBox:~$ ./hola
¡Hola, mundo! Desde RUST
joseramos@joseramos-VirtualBox:~$
```

## Lisp

## Instalación de paquetes

```
joseramos@joseramos-VirtualBox:~$ sudo apt-get install clisp
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libffcall1b libsigsegv2
Paquetes sugeridos:
  clisp-doc slime clisp-module-berkeley-db clisp-module-clx
  clisp-module-dbus clisp-module-gdbm clisp-module-pcre
  clisp-module-postgresql clisp-module-zlib
Se instalarán los siguientes paquetes NUEVOS:
  clisp libffcall1b libsigsegv2
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 151 no
actualizados.
Se necesita descargar 5.018 kB de archivos.
Se utilizarán 35,5 MB de espacio de disco adicional después de es
ta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu focal/universe amd64 li
bffcall1b amd64 2.2-1 [12,4 kB]
Des:2 http://es.archive.ubuntu.com/ubuntu focal/main amd64 libsig
segv2 amd64 2.12-2 [13,9 kB]
Des:3 http://es.archive.ubuntu.com/ubuntu focal/universe amd64 cl
isp amd64 1:2.49.20180218+really2.49.92-3build3 [4.992 kB]
Descargados 5.018 kB en 3s (1.663 kB/s)
Seleccionando el paquete libffcall1b:amd64 previamente no selecc
ionado.
(Leyendo la base de datos ... 204268 ficheros o directorios insta
lados actualmente.)
```

## Ejecutamos y Script

```
hola.lisp
Guardar
joseramos@joseramos-VirtualBox: ~
1 #!/usr/bin/env clisp
2
3 (format t "~;Hola, mundo!")

joseramos@joseramos-VirtualBox:~$ gedit hola.lisp
joseramos@joseramos-VirtualBox:~$ clisp
      00000  0      0000000  00000  00000
      8      8      8      8      8      8
      8      8      8      8      8      8
      8      8      8      00000  80000
      8      8      8      8      8
      8      0      8      8      8      8
      00000  8000000  0008000  00000  8

Bienvenido a GNU CLISP 2.49.92 (2018-02-18) <http://clisp.org/>

Copyright (c) Bruno Haible, Michael Stoll 1992-1993
Copyright (c) Bruno Haible, Marcus Daniels 1994-1997
Copyright (c) Bruno Haible, Pierpaolo Bernardi, Sam Steingold 1998
Copyright (c) Bruno Haible, Sam Steingold 1999-2000
Copyright (c) Sam Steingold, Bruno Haible 2001-2018

Teclee :h y pulse Intro para ayuda contextual.

[1]> (format t "~;Hola, mundo!")
;Hola, mundo!
NIL
[2]>
Adiós.

joseramos@joseramos-VirtualBox:~$ chmod +x hola.lisp
joseramos@joseramos-VirtualBox:~$ ./hola.lisp
;Hola, mundo!
joseramos@joseramos-VirtualBox:~$
```

**2. Para cada uno de los lenguajes anteriores, indica el proceso realizado para conseguir ejecutar el código: ¿compilación o interpretación?**

El proceso para ejecutar cada código aparece en las capturas de pantalla del ejercicio 1.

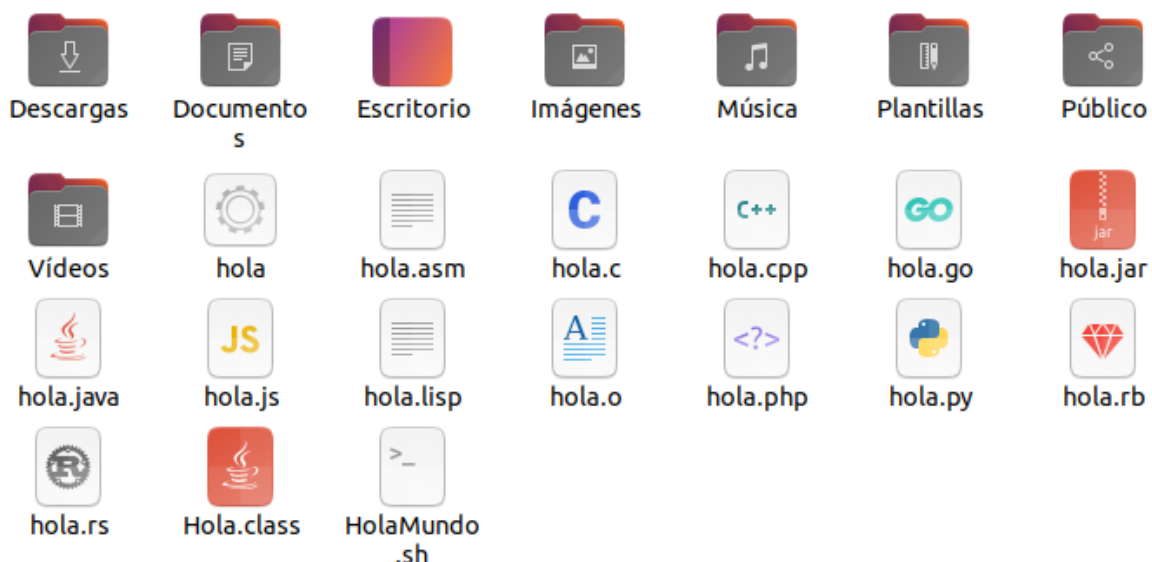
A continuación se indica cuales son compilados e interpretados:

- Lenguajes compilados: C, C + +, Java, Ensamblador, Go, Rust.
- Lenguajes interpretados: Bash, Python, PHP, Javascript, Ruby, Lisp.

**3. Para cada uno de los lenguajes anteriores, indica el nombre del compilador o intérprete utilizado en GNU/Linux.**

- bash: bash
- python: python
- php: php
- javascript (nodejs): node
- c: gcc
- c++: g++
- java: javac
- ruby: ruby
- go: go
- rust: rustc
- lisp: lisp
- ensamblador (nasm): nasm

**4. Investiga y averigua qué extensión tienen los archivos de código fuente de los siguientes lenguajes:**



La extensión de los archivos es:

bash: .sh  
python: .py  
php: .php  
javascript: .js  
c: .c  
c++: .cpp  
java: .java  
ensamblador: .asm  
ruby: .rb  
go: .go  
rust: .rs  
lisp: .lisp

**5. Scripts ejecutables para los lenguajes interpretados. Edita los scripts para los siguientes lenguajes.**

En las capturas de pantalla del ejercicio 1 aparecen los scripts ejecutables.

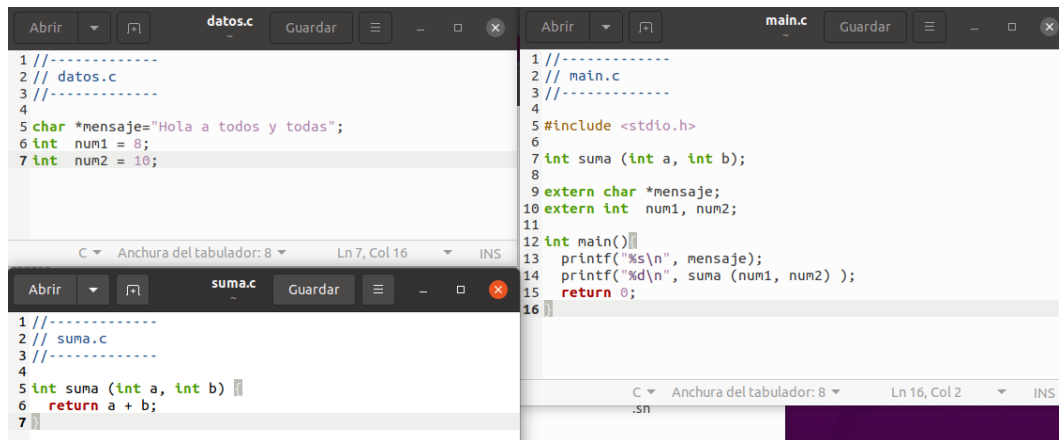
**6. ¿Qué extensión tienen los archivos de código objeto?**

- bash: no se utilizan archivos de código objeto en bash
- python: no se utilizan archivos de código objeto en python
- php: no se utilizan archivos de código objeto en php
- javascript (nodejs): no se utilizan archivos de código objeto en javascript (nodejs)
- c: .o
- c++: .obj
- java: .class
- ruby: no se utilizan archivos de código objeto en ruby
- go: .o
- rust: .o
- lisp: no se utilizan archivos de código objeto en lisp
- ensamblador (nasm): .o



## 7. Lenguaje C. Código en varios archivos. Obtener el código objeto a partir del código fuente de los 3 archivos siguientes:

Escribimos el código en cada uno de los archivos.

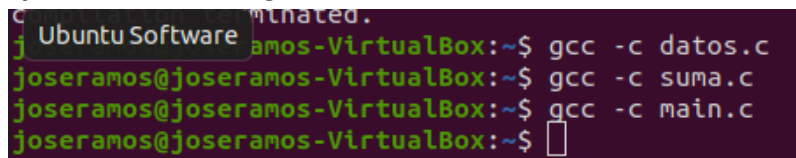


```
1 //-----
2 // datos.c
3 //-----
4
5 char *mensaje="Hola a todos y todas";
6 int num1 = 8;
7 int num2 = 10;

1 //-----
2 // main.c
3 //-----
4
5 #include <stdio.h>
6
7 int suma (int a, int b);
8
9 extern char *mensaje;
10 extern int num1, num2;
11
12 int main()
13 {
14     printf("%s\n", mensaje);
15     printf("%d\n", suma (num1, num2) );
16     return 0;
17 }

1 //-----
2 // suma.c
3 //-----
4
5 int suma (int a, int b)
6 {
7     return a + b;
8 }
```

Ejecutamos el código.



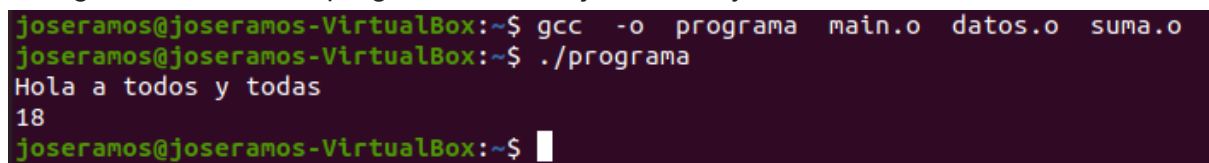
```
joseramos@joseramos-VirtualBox:~$ gcc -c datos.c
joseramos@joseramos-VirtualBox:~$ gcc -c suma.c
joseramos@joseramos-VirtualBox:~$ gcc -c main.c
joseramos@joseramos-VirtualBox:~$
```

Aquí tenemos los archivos .o



## 8. Lenguaje C. Código en varios archivos. Obtener el código binario ejecutable a partir del código objeto de los 3 archivos anteriores:

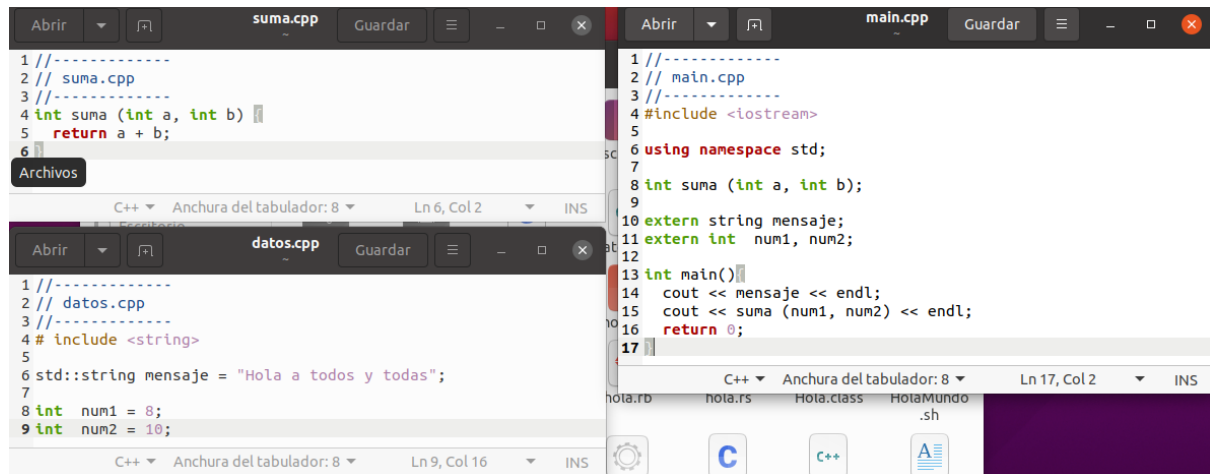
Tras generar el archivo programa.exe, lo ejecutamos y vemos el resultado.



```
joseramos@joseramos-VirtualBox:~$ gcc -o programa main.o datos.o suma.o
joseramos@joseramos-VirtualBox:~$ ./programa
Hola a todos y todas
18
joseramos@joseramos-VirtualBox:~$
```

## 9. Lenguaje C++. Código en varios archivos. Obtener el código objeto a partir del código fuente de los 3 archivos siguientes:

Generamos los tres archivos que necesitaremos.

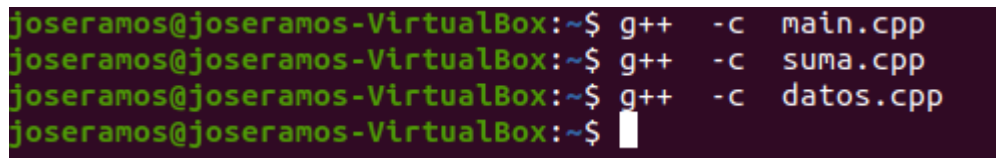


```
1 //-----
2 // suma.cpp
3 //-----
4 int suma (int a, int b) {
5     return a + b;
6 }

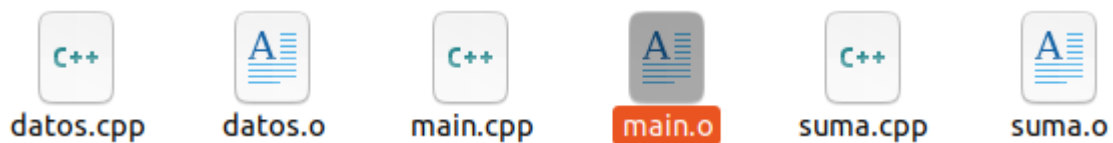
1 //-----
2 // datos.cpp
3 //-----
4 #include <string>
5
6 std::string mensaje = "Hola a todos y todas";
7
8 int num1 = 8;
9 int num2 = 10;

1 //-----
2 // main.cpp
3 //-----
4 #include <iostream>
5
6 using namespace std;
7
8 int suma (int a, int b);
9
10 extern string mensaje;
11 extern int num1, num2;
12
13 int main() {
14     cout << mensaje << endl;
15     cout << suma (num1, num2) << endl;
16     return 0;
17 }
```

Ejecutamos y obtenemos los archivos .o



```
joseramos@joseramos-VirtualBox:~$ g++ -c main.cpp
joseramos@joseramos-VirtualBox:~$ g++ -c suma.cpp
joseramos@joseramos-VirtualBox:~$ g++ -c datos.cpp
joseramos@joseramos-VirtualBox:~$
```



## 10. Lenguaje C++. Código en varios archivos. Obtener el código binario ejecutable a partir del código objeto de los 3 archivos anteriores:

Generamos el .exe y lo ejecutamos.

```
joseramos@joseramos-VirtualBox:~$ g++ -o programa main.o datos.o suma.o
joseramos@joseramos-VirtualBox:~$ ./programa
Hola a todos y todas
18
joseramos@joseramos-VirtualBox:~$
```

## 11. Bibliotecas. Define qué se entiende por biblioteca o librería y los tipos que existen.

En informática, una biblioteca o librería es un conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca.

A diferencia de un programa ejecutable, el comportamiento que implementa una biblioteca no espera ser utilizada de forma autónoma, sino que su fin es ser utilizada por otros programas, independientes y de forma simultánea. Por otra parte, el comportamiento de una biblioteca no tiene por qué diferenciarse demasiado del que pudiera especificarse en un programa. Es más, unas bibliotecas pueden requerir de otras para funcionar, pues el comportamiento que definen refina, o altera, el comportamiento de la biblioteca original; o bien la hace disponible para otra tecnología o lenguaje de programación.

La mayoría de los sistemas operativos modernos proporcionan bibliotecas que implementan los servicios del sistema. De esta manera, estos servicios se han convertido en una «materia prima» que cualquier aplicación moderna espera que el sistema operativo ofrezca. Como tal, la mayor parte del código utilizado por las aplicaciones modernas se ofrece en estas bibliotecas.

Existen dos tipos de librerías:

- Las librerías estáticas son aquellas que se incorporan directamente al código fuente del programa en el momento de la compilación. Esto hace que el tamaño del ejecutable final sea mayor, ya que se incluye todo el código de la librería en él. Sin embargo, esto también tiene la ventaja de que el programa resultante es autónomo y no necesita depender de ninguna otra librería externa para funcionar.
- Las librerías dinámicas, por otro lado, no se incorporan al código fuente del programa en el momento de la compilación. En su lugar, se cargan en tiempo de ejecución mediante un mecanismo de enlazado dinámico. Esto permite que el programa resultante sea más pequeño, ya que no incluye el código de la librería, y que se pueda utilizar la misma librería por varios programas al mismo tiempo. Sin embargo, requiere que la librería dinámica esté disponible en el sistema en el momento de la ejecución del programa.

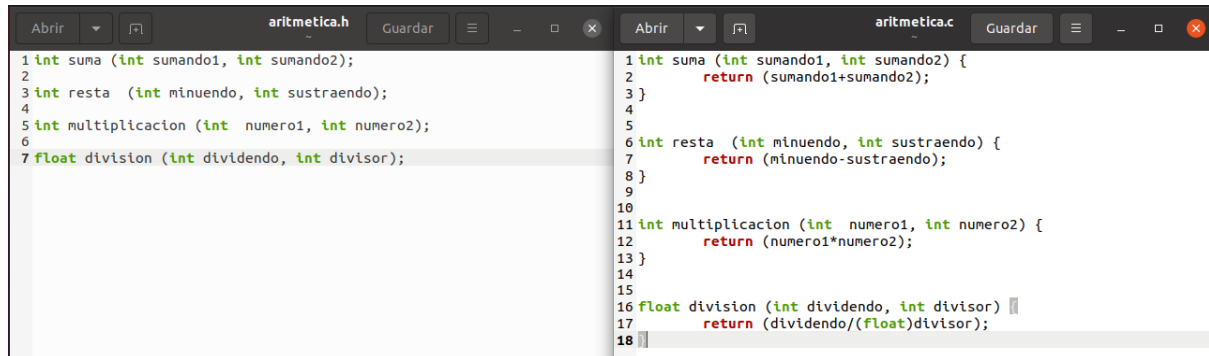
## 12. Bibliotecas. ¿Qué tipo es el más utilizado actualmente? ¿Por qué?

Lo normal es distribuir la funcionalidad básica de una aplicación en bibliotecas dinámicas y la funcionalidad opcional en forma de plugins.

El funcionamiento de los plugins es muy parecido al enlazado dinámico, con la salvedad que se carga la biblioteca en tiempo de ejecución. Esto permite, en un caso dado, comprobar si dicha biblioteca está disponible y hacer un uso de ella según el caso. Así prevenimos el error de carga del programa que se produce cuando no se encuentra la biblioteca enlazada dinámicamente.

## 13. Bibliotecas. Crea una biblioteca dinámica en C que proporcione las funciones para sumar, restar, multiplicar y dividir 2 números enteros. Crea un programa que haga uso de ella y comprueba que se ejecuta correctamente.

Creemos los archivos .h y .c



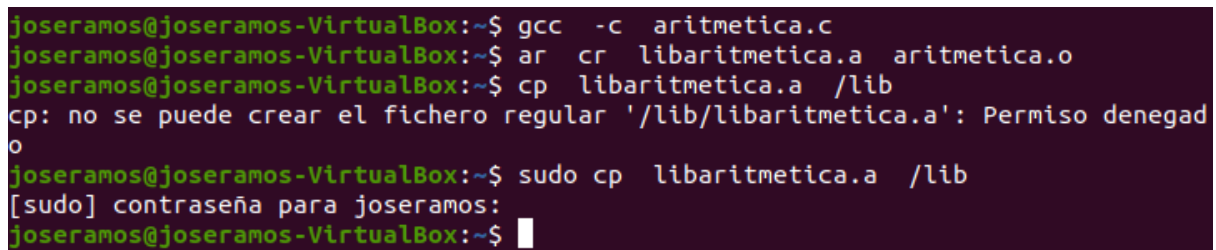
The image shows two side-by-side code editors. The left editor, titled 'aritmética.h', contains the following code:

```
1 int suma (int sumando1, int sumando2);
2
3 int resta (int minuendo, int sustraendo);
4
5 int multiplicacion (int numero1, int numero2);
6
7 float division (int dividendo, int divisor);
```

The right editor, titled 'aritmética.c', contains the following code:

```
1 int suma (int sumando1, int sumando2) {
2     return (sumando1+sumando2);
3 }
4
5
6 int resta (int minuendo, int sustraendo) {
7     return (minuendo-sustraendo);
8 }
9
10
11 int multiplicacion (int numero1, int numero2) {
12     return (numero1*numero2);
13 }
14
15
16 float division (int dividendo, int divisor) {
17     return (dividendo/(float)divisor);
18 }
```

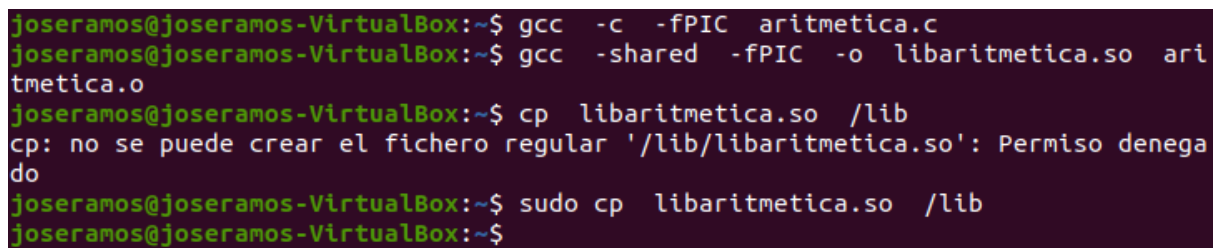
Pasamos a crear la biblioteca estática



The image shows a terminal window with the following commands and output:

```
joseramos@joseramos-VirtualBox:~$ gcc -c aritmetica.c
joseramos@joseramos-VirtualBox:~$ ar cr libaritmetica.a aritmetica.o
joseramos@joseramos-VirtualBox:~$ cp libaritmetica.a /lib
cp: no se puede crear el fichero regular '/lib/libaritmetica.a': Permiso denegado
joseramos@joseramos-VirtualBox:~$ sudo cp libaritmetica.a /lib
[sudo] contraseña para joseramos:
joseramos@joseramos-VirtualBox:~$
```

Y ahora la biblioteca dinámica



The image shows a terminal window with the following commands and output:

```
joseramos@joseramos-VirtualBox:~$ gcc -c -fPIC aritmetica.c
joseramos@joseramos-VirtualBox:~$ gcc -shared -fPIC -o libaritmetica.so aritmetica.o
joseramos@joseramos-VirtualBox:~$ cp libaritmetica.so /lib
cp: no se puede crear el fichero regular '/lib/libaritmetica.so': Permiso denegado
joseramos@joseramos-VirtualBox:~$ sudo cp libaritmetica.so /lib
joseramos@joseramos-VirtualBox:~$
```

Vamos a utilizar una biblioteca dinámica como plugin  
Primero creamos el archivo plug.c

```

1
2 #include <dlfcn.h>
3 #include <stdio.h>
4
5 #define NUM1    5
6 #define NUM2    2
7
8
9 int main ()
10     void *h = dlopen("./libaritmetica.so", RTLD_LAZY);
11
12     printf ("Dados los números %d y %d\n", NUM1, NUM2);
13
14     int (*test)() = dlsym (h, "suma");
15     printf ("La suma es %d\n", (*test)(NUM1, NUM2));
16
17     test = dlsym (h, "resta");
18     printf ("La resta es %d\n", (*test)(NUM1, NUM2));
19
20     test = dlsym (h, "multiplicacion");
21     printf ("La multiplicación es %d\n", (*test)(NUM1, NUM2));
22
23     float (*test2)() = dlsym (h, "division");
24     printf ("La división es %f\n", (*test2)(NUM1, NUM2));
25     return 0;
26

```

Compilamos, enlazamos y ejecutamos el programa

```

joseramos@joseramos-VirtualBox:~$ gcc -o plug plug.c -ldl
joseramos@joseramos-VirtualBox:~$ ./plug
Dados los números 5 y 2
La suma es 7
La resta es 3
La multiplicación es 10
La división es 2.500000
joseramos@joseramos-VirtualBox:~$

```

Vamos a crear un ejecutable con enlace estático.

Aquí tenemos el archivo main.c

```
main.c
1 #include <stdio.h>
2 #include "aritmetica.h"
3
4 #define NUM1    5
5 #define NUM2    2
6
7
8 int main ()
9 {
10     printf ("Dados los números %d y %d\n", NUM1, NUM2);
11     printf ("La suma es %d\n", suma (NUM1, NUM2));
12     printf ("La resta es %d\n", resta (NUM1, NUM2));
13     printf ("La multiplicación es %d\n", multiplicacion (NUM1, NUM2));
14     printf ("La división es %f\n", division (NUM1, NUM2));
15     return 0;
}
```

Observamos que si aparece la biblioteca libaritmetica.so como enlace dinámico.

```
joseramos@joseramos-VirtualBox:~$ gcc -o main main.c libaritmetica.a
joseramos@joseramos-VirtualBox:~$ ldd main
linux-vdso.so.1 (0x00007ffe1f598000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fd787552000)
/lib64/ld-linux-x86-64.so.2 (0x00007fd78775c000)
```

Distribución de binario junto a biblioteca en la misma carpeta: en primer lugar compilamos y enlazamos, indicando el directorio donde está la biblioteca, para terminar comprobando vínculos dinámicos.

```
joseramos@joseramos-VirtualBox:~$ gcc -L. -Wl,-rpath=. -Wall -o main main.c -laritmetica
joseramos@joseramos-VirtualBox:~$ ldd main
linux-vdso.so.1 (0x00007ffe94bac000)
libaritmetica.so => ./libaritmetica.so (0x00007fc0f1a8d000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fc0f188a000)
/lib64/ld-linux-x86-64.so.2 (0x00007fc0f1a99000)
```

Distribución de binario junto a biblioteca en una subcarpeta: primero se crea subdirectorio y movemos biblioteca a él, compilamos y enlazamos, indicando el directorio donde está localizada la biblioteca, en este caso ./libs, para finalmente comprobar vínculos dinámicos.

```
joseramos@joseramos-VirtualBox:~$ mkdir libs
joseramos@joseramos-VirtualBox:~$ mv libaritmetica.so libs
joseramos@joseramos-VirtualBox:~$ gcc -L./libs -Wl,-rpath=libs -Wall -o main main.c -laritmetica
joseramos@joseramos-VirtualBox:~$ ldd main
linux-vdso.so.1 (0x00007fffeece9000)
libaritmetica.so => libs/libaritmetica.so (0x00007ffa11a7b000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007ffa11878000)
/lib64/ld-linux-x86-64.so.2 (0x00007ffa11a87000)
```

**14. Bibliotecas. Crea una biblioteca dinámica en Java que proporcione las funciones para sumar, restar, multiplicar y dividir 2 números enteros. Crea un programa que haga uso de ella y comprueba que se ejecuta correctamente.**

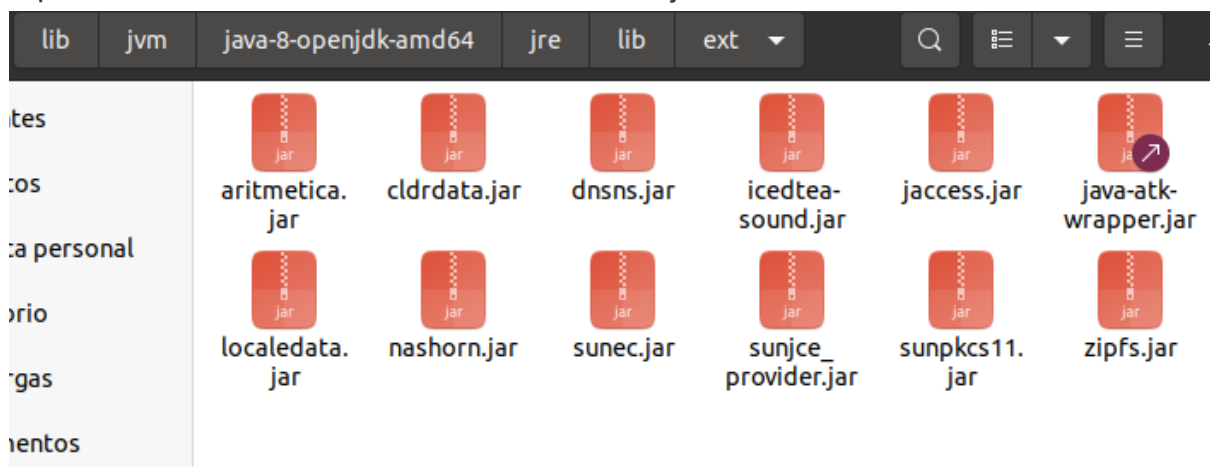
En primer lugar hay que asegurarse de instalar la versión 8 de jdk. Con versiones posteriores habría problemas al usar este método.

Lo instalaremos con el comando: `sudo apt install openjdk-8-jdk`.

Creamos directorio aritmetica con comando `mkdir`, para crear dentro de él la clase `Aritmetica.java`

```
Abrir  Aritmetica.java
~/aritmetica
1 package aritmetica;
2
3 public class Aritmetica {
4
5     public static int suma (int sumando1, int sumando2) {
6         return (sumando1+sumando2);
7     }
8
9     public static int resta (int minuendo, int sustraendo) {
10        return (minuendo-sustraendo);
11    }
12
13    public static int multiplicacion (int numero1, int numero2) {
14        return (numero1*numero2);
15    }
16
17    public static float division (int dividendo, int divisor) {
18        return (dividendo/(float)divisor);
19    }
20
21 }
```

Compilamos y obtenemos el `.class`, para posteriormente crear el paquete `jar`, el cual copiaremos al directorio de sistema donde se alojan las librerías de extensiones.





Ahora vamos a crear un programa que use la biblioteca.

Primero creamos el archivo Main.java el cual compilaremos para obtener el Main.class.

Finalmente ejecutamos

```
root@joseramos-VirtualBox:/home/joseramos# mv aritmetica.jar /usr/lib/jvm/java-8-openjdk-amd64/jre/lib/ext
root@joseramos-VirtualBox:/home/joseramos# javac Main.java
root@joseramos-VirtualBox:/home/joseramos# java Main
Dados los números 5 y 2
La suma es 7
La resta es 3
La multiplicación es 10
La división es 2.5
root@joseramos-VirtualBox:/home/joseramos#
```

### **15. Bibliotecas. Busca información y explica las ventajas y desventajas de usar bibliotecas estáticas.**

Las bibliotecas estáticas son conjuntos de código que se compilan con un programa, en lugar de cargarse dinámicamente en tiempo de ejecución.

Una de las principales ventajas de usar bibliotecas estáticas es que pueden mejorar el rendimiento del programa, ya que el código ya está compilado y listo para su uso en lugar de tener que cargarse y compilarse dinámicamente en tiempo de ejecución.

Otra ventaja es que las bibliotecas estáticas pueden ayudar a ahorrar espacio en disco, ya que el código de la biblioteca se incluye directamente en el ejecutable del programa en lugar de tener que almacenarse como un archivo separado.

Sin embargo, también hay algunas desventajas al usar bibliotecas estáticas. Una de ellas es que pueden hacer que el código sea más difícil de mantener y actualizar, ya que si se produce un cambio en la biblioteca, el código del programa completo debe recompilarse para reflejar ese cambio.

Además, las bibliotecas estáticas pueden dificultar el trabajo en equipo, ya que todos los desarrolladores deben tener la misma versión de la biblioteca para evitar conflictos de compilación. En general, las bibliotecas estáticas pueden ser útiles en algunos casos, pero también pueden presentar desafíos en términos de mantenimiento y colaboración en equipo.

### **16. Bibliotecas. Busca información y explica las ventajas y desventajas de usar bibliotecas dinámicas.**

Las bibliotecas dinámicas son conjuntos de código que se cargan en tiempo de ejecución en lugar de incluirse en el programa en tiempo de compilación.

Una de las principales ventajas de usar bibliotecas dinámicas es que permiten una mayor flexibilidad en términos de actualización y mantenimiento del código. Si se

produce un cambio en la biblioteca, solo es necesario actualizar el archivo de la biblioteca en lugar de tener que recompilar el programa completo.

Además, las bibliotecas dinámicas pueden facilitar el trabajo en equipo, ya que todos los desarrolladores pueden usar la misma versión de la biblioteca sin necesidad de preocuparse por conflictos de compilación.

Sin embargo, también hay algunas desventajas al usar bibliotecas dinámicas. Una de ellas es que pueden reducir el rendimiento del programa, ya que el código de la biblioteca debe cargarse y compilarse dinámicamente en tiempo de ejecución.

Además, las bibliotecas dinámicas pueden ocupar más espacio en disco, ya que el código de la biblioteca se almacena como un archivo separado en lugar de incluirse directamente en el ejecutable del programa. En general, las bibliotecas dinámicas pueden ser útiles en algunos casos, pero también pueden tener desventajas en términos de rendimiento y uso de espacio en disco.

## **17. Build. Automatiza el proceso de compilación de ejecutable y biblioteca, su enlazado y la generación del archivo ejecutable para código fuente en C con make. Haz uso de un buildfile.**

Primero instalamos el paquete make.

Compilar y enlazar todo, generando el ejecutable y la biblioteca asociada.

```
joseramos@joseramos-VirtualBox:~$ gcc -O -c main.c
joseramos@joseramos-VirtualBox:~$ gcc -O -c -fPIC aritmetica.c
joseramos@joseramos-VirtualBox:~$ gcc -O -shared -fPIC -o libaritmetica.so
aritmetica.o
joseramos@joseramos-VirtualBox:~$ gcc -O -Wl,-rpath=/usr/local/lib main.o lib
aritmetica.so -o programa
joseramos@joseramos-VirtualBox:~$
```

```
joseramos@joseramos-VirtualBox:~$ make help
Objetivos válidos para make:

all (el objetivo por defecto si no se indica nada)
clean
install
help
```

```
joseramos@joseramos-VirtualBox:~$ sudo make install
gcc -O -Wl,-rpath=/usr/local/lib main.o libaritmetica.so -o programa
[ -d /usr/local ] || mkdir /usr/local
[ -d /usr/local/bin ] || mkdir /usr/local/bin
[ -d /usr/local/lib ] || mkdir /usr/local/lib
install -m 0755 programa /usr/local/bin
install -m 0644 libaritmetica.so /usr/local/lib
```

**18. Build. Automatiza el proceso de compilación de ejecutable y biblioteca, su enlazado y la generación del archivo .jar para código fuente en Java con Ant. Haz uso de un buildfile.**

Creamos los archivos Aritmetica.java, Main.java y build.xml

Para poder ejecutar ant, deberemos primero instalar el paquete ant.

```
joseramos@joseramos-VirtualBox:~$ ant

No se ha encontrado la orden «ant», pero se puede instalar con:

sudo snap install ant # version 1.10.12, or
sudo apt install ant # version 1.10.7-1

Consulte «snap info ant» para ver más versiones.

joseramos@joseramos-VirtualBox:~$ ^C
joseramos@joseramos-VirtualBox:~$ sudo apt install ant # version 1.10.7-1
[sudo] contraseña para joseramos:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 ant-optional
Paquetes sugeridos:
 ant-doc antlr javacc junit junit4 jython libactivation-java libbccl-java
```

Ahora sí, al ejecutar lo hace correctamente.

```
joseramos@joseramos-VirtualBox:~$ ant
Buildfile: /home/joseramos/build.xml

init:
 [mkdir] Created dir: /home/joseramos/build/classes
 [mkdir] Created dir: /home/joseramos/build/jar

compile:
 [javac] Compiling 2 source files to /home/joseramos/build/classes

jar:
 [jar] Building jar: /home/joseramos/build/jar/programa.jar

BUILD SUCCESSFUL
Total time: 0 seconds
```

```
joseramos@joseramos-VirtualBox:~$ ant run
Buildfile: /home/joseramos/build.xml

init:

compile:

jar:

run:
    [java] Dados los números 5 y 2
    [java] La suma es 7
    [java] La resta es 3
    [java] La multiplicación es 10
    [java] La división es 2.5

BUILD SUCCESSFUL
Total time: 0 seconds
joseramos@joseramos-VirtualBox:~$
```

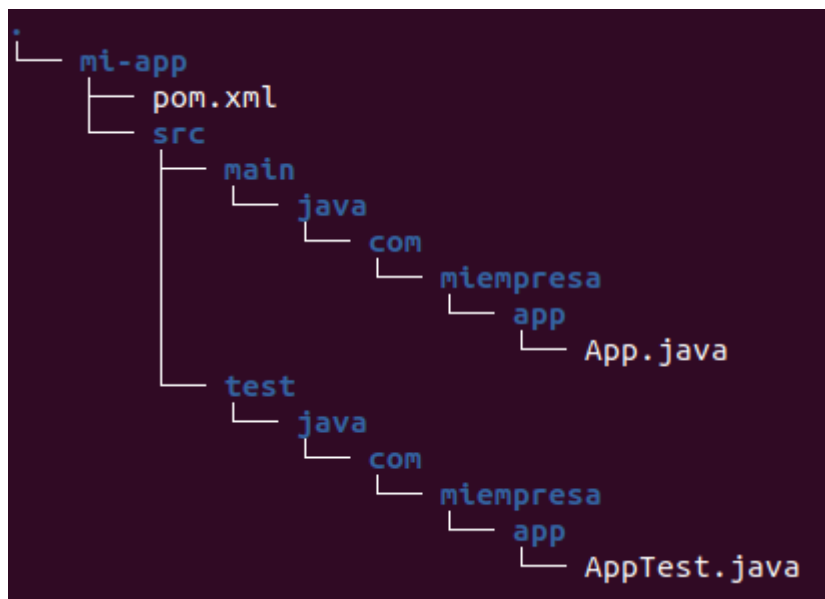
**19. Build.** Automatiza el proceso de compilación de ejecutable y biblioteca, su enlazado y la generación del archivo .jar para código fuente en Java con Maven. Haz uso de un buildfile.

Vamos a crear un proyecto Java con Maven. En este caso, una aplicación llamada mi-app.

En primer lugar tendremos que instalar el paquete maven

```
joseramos@joseramos-VirtualBox:~$ sudo apt install maven
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java
  libcommons-parent-java libgeronimo-annotation-1.3-spec-java
  libgeronimo-interceptor-3.0-spec-java libguava-java libguice-java
  libhawtjni-runtime-java libjansi-java libjansi-native-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java
  libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java
  libplexus-component-annotations-java libplexus-interpolation-java
  libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java
  libsisu-plexus-java libslf4j-java libwagon-file-java
  libwagon-http-shaded-java libwagon-provider-api-java
Paquetes sugeridos:
  libaopalliance-java-doc libatinject-jsr330-api-java-doc libservlet3.1-java
  libcdi-api-java-doc libcommons-cli-java-doc libcommons-io-java-doc libcommons-lang3-java-doc
  libcommons-parent-java-doc libgeronimo-annotation-1.3-spec-java-doc libgeronimo-interceptor-3.0-spec-java-doc
  libguava-java-doc libguice-java-doc libhawtjni-runtime-java-doc libjansi-java-doc libjansi-native-java-doc libjsr305-java-doc
  libmaven-parent-java-doc libmaven-resolver-java-doc libmaven-shared-utils-java-doc libmaven3-core-java-doc libplexus-cipher-java-doc
  libplexus-classworlds-java-doc libplexus-component-annotations-java-doc libplexus-interpolation-java-doc libplexus-sec-dispatcher-java-doc
  libplexus-utils2-java-doc libsisu-inject-java-doc libsisu-plexus-java-doc libslf4j-java-doc libwagon-file-java-doc libwagon-http-shaded-java-doc
  libwagon-provider-api-java-doc
```

Seguidamente creamos la estructura de carpetas.



Aquí vemos el pom.xml que hemos creado

```
joseramos@joseramos-VirtualBox:~/Proyectos/mi-app$ cat pom.xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.miempresa.app</groupId>
  <artifactId>mi-app</artifactId>
  <packaging>jar</packaging>
  <version>1.0.0</version>
  <name>mi-app</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Creamos las siguientes clases en src/main/java/com/miempresa/app/ y src/test/java/com/miempresa/app/ respectivamente



App.java



App.java.  
save



AppTest.  
java



Aritmetica.  
java



AritmeticaT  
est.java

Editamos el pom.xml

```
<project>

  <modelVersion>4.0.0</modelVersion>
  <groupId>com.miempresa.app</groupId>
  <artifactId>mi-app</artifactId>
  <version>1.0.0</version>
  <name>mi-app</name>

  <build>
    <plugins>
      <plugin>
        <!-- Para construir un JAR ejecutable -->
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-jar-plugin</artifactId>
        <version>3.0.2</version>
        <configuration>
          <archive>
            <manifest>
              <addClasspath>true</addClasspath>
              <classpathPrefix>./</classpathPrefix>
              <mainClass>com.miempresa.app.App</mainClass>
            </manifest>
          </archive>
        </configuration>
      </plugin>

      <plugin>
        <!-- Para ejecutar el JAR creado -->
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>exec-maven-plugin</artifactId>
        <version>1.2.1</version>
        <configuration>
          <mainClass>com.miempresa.app.App</mainClass>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

## Compilamos

```
joseramos@joseramos-VirtualBox:~/Proyectos/mi-app$ mvn compile
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.miestra.app:mi-app >-----
[INFO] Building mi-app 1.0.0
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ mi-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/joseramos/Proyectos/mi-app/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ mi-app ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 2 source files to /home/joseramos/Proyectos/mi-app/target/classes
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 1.012 s
[INFO] Finished at: 2022-12-15T18:08:20+01:00
[INFO]
joseramos@joseramos-VirtualBox:~/Proyectos/mi-app$
```

## Ejecutamos el bytecode y ya lo tenemos

```
joseramos@joseramos-VirtualBox:~/Proyectos/mi-app$ cd target/classes && java com.miestra.app.App && cd ../..
Dados los números 5 y 2
La suma es 7
La resta es 3
La multiplicación es 10
La división es 2.5
```

## Ahora vamos a empaquetar el .jar, el cual ejecutamos y realizamos las pruebas unitarias

```
joseramos@joseramos-VirtualBox:~/Proyectos/mi-app$ mvn exec:java
[INFO] Scanning for projects...
Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/mojo/exec-maven-plugin/1.2.1/exec-maven-plugin-1.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/mojo/exec-maven-plugin/1.2.1/exec-maven-plugin-1.2.1.pom (7.7 kB at 18 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/mojo/mojo-parent/28/mojo-parent-28.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/mojo/mojo-parent/28/mojo-parent-28.pom (26 kB at 343 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/codehaus-parent/3/codehaus-parent-3.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/codehaus-parent/3/codehaus-parent-3.pom (4.1 kB at 142 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/mojo/exec-maven-plugin/1.2.1/exec-maven-plugin-1.2.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/mojo/exec-maven-plugin/1.2.1/exec-maven-plugin-1.2.1.jar (38 kB at 923 kB/s)
[INFO]
[INFO] -----< com.miestra.app:mi-app >-----
[INFO] Building mi-app 1.0.0
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> exec-maven-plugin:1.2.1:java (default-cli) > validate @ mi-app >>>
```



```

-----
T E S T S
-----
Running com.miempresa.app.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.001 sec
Running com.miempresa.app.AritmeticaTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 1.164 s
[INFO] Finished at: 2022-12-15T18:12:44+01:00
[INFO] -----
joseramos@joseramos-VirtualBox:~/Proyectos/mi-app$

```

Como vemos ha ido correctamente.

## 20. Build. Automatiza el proceso de compilación de ejecutable y biblioteca, su enlazado y la generación del archivo .jar para código fuente en Java con Gradle. Haz uso de un buildfile.

En primer lugar instalamos el paquete Grandle y creamos los directorios.

```

joseramos@joseramos-VirtualBox:~$ mkdir nombre-proyecto && cd nombre-proyecto
joseramos@joseramos-VirtualBox:~/nombre-proyecto$ gradle init --type java-application
Support for nested build without a settings file was deprecated and will be removed in Gradle 5.0. You should create a empty settings file in /home/joseramos/nombre-proyecto

BUILD SUCCESSFUL in 0s
2 actionable tasks: 2 executed
joseramos@joseramos-VirtualBox:~/nombre-proyecto$

```

Ahora procedemos a crear una aplicación llamada miapp.

Primero crearemos los directorios.

```
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$ tree
.
├── build.gradle
├── gradle
│   └── wrapper
│       ├── gradle-wrapper.jar
│       └── gradle-wrapper.properties
├── gradlew
├── gradlew.bat
├── settings.gradle
└── src
    ├── main
    │   └── java
    │       └── App.java
    └── test
        └── java
            └── AppTest.java

7 directories, 8 files
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$
```

Ahora crearemos las clases Main.java, Aritmetica.java, MainTest.java y AritmeticaTest.java, como hicimos en el ejercicio anterior.

Editamos el build.gradle.

```
Abrir ▼ [+]
```

**build.gradle**  
~/Proyectos/miapp

```
1 apply plugin: 'java'
2 apply plugin: 'application'
3
4 repositories {
5     jcenter()
6 }
7
8 dependencies {
9     compile 'com.google.guava:guava:23.0'
10    testCompile 'junit:junit:4.12'
11 }
12
13 jar {
14     manifest {
15         attributes ('Main-Class': 'Main')
16     }
17 }
18
19 mainClassName = 'Main'
```

## Compilamos

```
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$ rm src/main/java/App.java src/test/java/AppTest.java
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$ ./gradlew assemble
Downloading https://services.gradle.org/distributions/gradle-4.4.1-bin.zip
.....
Unzipping /home/joseramos/.gradle/wrapper/dists/gradle-4.4.1-bin/46gopw3g8i1v3zqqx4q949t2x/gradle-4.4.1-bin.zip to /home/joseramos/.gradle/wrapper/dists/gradle-4.4.1-bin/46gopw3g8i1v3zqqx4q949t2x
Set executable permissions for: /home/joseramos/.gradle/wrapper/dists/gradle-4.4.1-bin/46gopw3g8i1v3zqqx4q949t2x/gradle-4.4.1/bin/gradle
Download https://jcenter.bintray.com/com/google/guava/guava/23.0/guava-23.0.pom
Download https://jcenter.bintray.com/com/google/guava/guava-parent/23.0/guava-parent-23.0.pom
Download https://jcenter.bintray.com/org/sonatype/oss/oss-parent/7/oss-parent-7.pom
Download https://jcenter.bintray.com/com/google/j2objc/j2objc-annotations/1.1/j2objc-annotations-1.1.pom
Download https://jcenter.bintray.com/com/google/code/findbugs/jsr305/1.3.9/jsr305-1.3.9.pom
Download https://jcenter.bintray.com/org/codehaus/mojo/animal-sniffer-annotations/1.14/animal-sniffer-annotations-1.14.pom
Download https://jcenter.bintray.com/com/google/errorprone/error_prone_annotations/2.0.18/error_prone_annotations-2.0.18.pom
Download https://jcenter.bintray.com/com/google/errorprone/error_prone_parent/2.0.18/error_prone_parent-2.0.18.pom
Download https://jcenter.bintray.com/org/codehaus/mojo/animal-sniffer-parent/1.14/animal-sniffer-parent-1.14.pom
Download https://jcenter.bintray.com/org/codehaus/mojo/mojo-parent/34/mojo-parent-34.pom
Download https://jcenter.bintray.com/org/codehaus/codehaus-parent/4/codehaus-parent-4.pom
Download https://jcenter.bintray.com/com/google/errorprone/error_prone_annotations/2.0.18/error_prone_annotations-2.0.18.jar
Download https://jcenter.bintray.com/com/google/j2objc/j2objc-annotations/1.1/j2objc-annotations-1.1.jar
Download https://jcenter.bintray.com/com/google/code/findbugs/jsr305/1.3.9/jsr305-1.3.9.jar
Download https://jcenter.bintray.com/org/codehaus/mojo/animal-sniffer-annotations/1.14/animal-sniffer-annotations-1.14.jar
Download https://jcenter.bintray.com/com/google/guava/guava/23.0/guava-23.0.jar

BUILD SUCCESSFUL in 7s
5 actionable tasks: 5 executed
```

## Ejecutamos el bytecode

```
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$ cd build/classes/main && java Main && cd ../../..
bash: cd: build/classes/main: No existe el archivo o el directorio
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$ java -jar build/libs/miapp.jar
Dados los números 5 y 2
La suma es 7
La resta es 3
La multiplicación es 10
La división es 2.5
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$
```

## Realizamos las pruebas unitarias

```
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$ ./gradlew test

BUILD SUCCESSFUL in 1s
3 actionable tasks: 2 executed, 1 up-to-date
joseramos@joseramos-VirtualBox:~/Proyectos/miapp$
```

Por último abrimos en firefox el informe de las pruebas

Default package

all > default-package

2 tests, 0 failures, 0 ignored, 0.001s duration

100% successful

Classes

Class	Tests	Failures	Ignored	Duration	Success rate
<a href="#">AritmeticaTest</a>	1	0	0	0.001s	100%
<a href="#">MainTest</a>	1	0	0	0s	100%

Generated by [Gradle 4.4.1](#) at 15-dic-2022 18:37:02

## 21. CMake. Automatiza el proceso de compilación de ejecutable y bibliotecas, su enlazado y la generación del archivo ejecutable para código fuente en C++. Crea un buildfile con CMake.

Descargamos el código fuente desde GitHub

```
joseramos@joseramos-VirtualBox:~$ git clone https://github.com/jamj2000/DAW1-ED-Bibliotecas.git
Clonando en 'DAW1-ED-Bibliotecas'...
remote: Enumerating objects: 335, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (42/42), done.
remote: Total 335 (delta 24), reused 0 (delta 0), pack-reused 293
Recibiendo objetos: 100% (335/335), 105.64 KiB | 1.24 MiB/s, listo.
Resolviendo deltas: 100% (173/173), listo.
```

Después entraremos desde consola en el directorio DAW1-ED-Bibliotecas/cpp y crearemos un directorio de construcción.

Instalamos el paquete cmake

```
joseramos@joseramos-VirtualBox:~$ sudo snap install cmake --classic
Montar snap "cmake" (1210)
```

Generamos archivo Makefile y comprobamos que lo ha creado

```
joseramos@joseramos-VirtualBox:~$ cat Makefile
#####
#
# Cliente de correo Thunderbird Makefile      (cc0) jamj2000
#
#####

##### MACROS

# Compilador de C
CC      = gcc

# Opciones del compilador, en este caso Optimización
CFLAGS = -O

# Directorio de instalación
PREFIX = /usr/local

# Ejecutable resultante
OUTPUT = programa

##### REGLAS

# .PHONY indica objetivos especiales, que no corresponden a archivos
#
# Las reglas tienen la forma
# objetivo(target) : dependencias
# <TAB> comando1
# <TAB> comando2
# <TAB> ...
#
# La primera regla es la regla por defecto, puede invocarse simplemente con la orden make
```

Construimos con comando make e instalamos los archivos generados

```
joseramos@joseramos-VirtualBox:~$ make
gcc -O -Wl,-rpath=/usr/local/lib main.o libaritmetica.so -o programa
joseramos@joseramos-VirtualBox:~$ sudo make install
[sudo] contraseña para joseramos:
gcc -O -Wl,-rpath=/usr/local/lib main.o libaritmetica.so -o programa
[ -d /usr/local ] || mkdir /usr/local
[ -d /usr/local/bin ] || mkdir /usr/local/bin
[ -d /usr/local/lib ] || mkdir /usr/local/lib
install -m 0755 programa /usr/local/bin
install -m 0644 libaritmetica.so /usr/local/lib
joseramos@joseramos-VirtualBox:~$
```

