

## Enunciado

Se propone al estudiante instalar en su distribución de Linux el lenguaje programación C (gcc).

Los pasos que debe seguir son:

---

### *Parte 1: Instalar gcc sobre Ubuntu (obligatorio)*

---

Realice los pasos a continuación para instalar el compilador GCC Ubuntu 18.04. Comienza actualizando la lista de paquetes:

#### - **sudo apt-get update**

```
mcardenas@mcardenas:~$ sudo apt-get update
[sudo] password for mcardenas:
Obj:1 http://es.archive.ubuntu.com/ubuntu bionic InRelease
Des:2 http://es.archive.ubuntu.com/ubuntu bionic-updates InRelease [88,7 kB]
Des:3 http://es.archive.ubuntu.com/ubuntu bionic-backports InRelease [74,6 kB]
Des:4 http://es.archive.ubuntu.com/ubuntu bionic-security InRelease [88,7 kB]
Des:5 http://es.archive.ubuntu.com/ubuntu bionic/main Translation-es [364 kB]
Des:6 http://es.archive.ubuntu.com/ubuntu bionic/restricted Translation-es [1.960 B]
Des:7 http://es.archive.ubuntu.com/ubuntu bionic/universe Translation-es [1.259 kB]
Des:8 http://es.archive.ubuntu.com/ubuntu bionic/multiverse Translation-es [74,9 kB]
Descargados 1.952 kB en 2s (949 kB/s)
Leyendo lista de paquetes... Hecho
mcardenas@mcardenas:~$ _
```

Instala el paquete build-essential escribiendo los siguientes comandos:

#### - **sudo apt-get install build-essential**

```
mcardenas@mcardenas:~$ sudo apt-get install build-essential
```

Al final de la instalación tendrás que ver algo como lo siguiente:

```
Configurando libgcc-7-dev:amd64 (7.4.0-1ubuntu1~18.04.1) ...
Configurando cpp-7 (7.4.0-1ubuntu1~18.04.1) ...
Configurando libstdc++-7-dev:amd64 (7.4.0-1ubuntu1~18.04.1) ...
Configurando libalgorithm-merge-perl (0.08-3) ...
Configurando libalgorithm-diff-xs-perl (0.04-5) ...
Configurando binutils-x86-64-linux-gnu (2.30-21ubuntu1~18.04.2) ...
Configurando cpp (4:7.4.0-1ubuntu2.3) ...
Configurando binutils (2.30-21ubuntu1~18.04.2) ...
Configurando gcc-7 (7.4.0-1ubuntu1~18.04.1) ...
Configurando g++-7 (7.4.0-1ubuntu1~18.04.1) ...
Configurando gcc (4:7.4.0-1ubuntu2.3) ...
Configurando dpkg-dev (1.19.0.5ubuntu2.3) ...
Configurando g++ (4:7.4.0-1ubuntu2.3) ...
update-alternatives: utilizando /usr/bin/g++ para proveer /usr/bin/c++ (c++) en modo automático
Configurando build-essential (12.4ubuntu1) ...
Procesando disparadores para libc-bin (2.27-3ubuntu1) ...
mcardenas@mcardenas:~$ _
```

Este comando instala en tu ordenador todo un conjunto de paquetes nuevos que incluyen **gcc**, **g++** y **make**. Podrías de igual forma, instalar las páginas del manual sobre el uso de GNU/ Linux para el desarrollo, aunque lo más probable es que ya esté instalado.

#### - **sudo apt-get install manpages-dev**

```
mcardenas@mcardenas:~$ sudo apt-get install manpages-dev
```

Comprueba que has instalado correctamente el compilador GCC. Para ello, utiliza el comando `gcc --version` para visualizar la versión del compilador:

- `gcc --version`

```
mcardenas@mcardenas:~$ gcc --version
gcc (Ubuntu 7.4.0-1ubuntu1~18.04.1) 7.4.0
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
mcardenas@mcardenas:~$ _
```

---

*Crea tu primer programa en C*

---

```
mcardenas@mcardenas:~$ sudo nano hola.c_
```

Usa el editor por defecto de Linux (nano) para crear el siguiente programa:

```
GNU nano 2.9.3 hola.c
#include <stdio.h>

int main(){
printf ("Hola Pitter");
return 0;
}
```

Nano es un editor de Linux muy útil para editar ficheros. En este caso lo usarás para crear tu fichero **"hola.c"** y en el escribir el código anterior. En este editor, para guardar utiliza la combinación de teclas: **Ctrl + O** para escribir el fichero (guardar) y **Ctrl + X** para salir del editor.

Una vez creado y guardado el fichero, compila y genera un ejecutable con la siguiente instrucción:

```
mcardenas@mcardenas:~$ gcc hola.c -o hola_
```



Con **"sudo loadkeys es"** puedes cargar el teclado en español. Sino funciona instala el paquete de utilidades de teclado:  
**"sudo apt-get install x11-xkb-utils"**

Una vez compilado, verifica la existencia del nuevo fichero llamado **"hola"**. Utiliza el comando **"ls -al"** para listar todos los ficheros existentes en el directorio actual.

Ejecuta el comando nuevo con la siguiente instrucción: **./hola**

```
mcardenas@mcardenas:~$ ./hola
Hola Pitter
mcardenas@mcardenas:~$ _
```

## Parte 2: Instalar varias versiones de gcc (Opcional)

Añade el siguiente repositorio a la PPA **ubuntu-toolchain-r/test PPA**

- **sudo apt-get install software-properties-common**
- **sudo add-apt-repository ppa:ubuntu-toolchain-r/test**

```
mcardenas@mcardenas:~$ sudo apt-get install software-properties-common
[sudo] password for mcardenas: _
```

```
mcardenas@mcardenas:~$ sudo add-apt-repository ppa:ubuntu-toolchain-r/test
```

El resultado que obtienes será algo como lo que se muestra en la siguiente imagen:

```
mcardenas@mcardenas:~$ sudo add-apt-repository ppa:ubuntu-toolchain-r/test
Toolchain test builds; see https://wiki.ubuntu.com/ToolChain

More info: https://launchpad.net/~ubuntu-toolchain-r/+archive/ubuntu/test
Press [ENTER] to continue or Ctrl-c to cancel adding it.

Des:1 http://ppa.launchpad.net/ubuntu-toolchain-r/test/ubuntu bionic InRelease [15,4 kB]
Obj:2 http://es.archive.ubuntu.com/ubuntu bionic InRelease
Des:3 http://es.archive.ubuntu.com/ubuntu bionic-updates InRelease [88,7 kB]
Des:4 http://es.archive.ubuntu.com/ubuntu bionic-backports InRelease [74,6 kB]
Des:5 http://ppa.launchpad.net/ubuntu-toolchain-r/test/ubuntu bionic/main amd64 Packages [23,1 kB]
Des:6 http://es.archive.ubuntu.com/ubuntu bionic-security InRelease [88,7 kB]
Des:7 http://ppa.launchpad.net/ubuntu-toolchain-r/test/ubuntu bionic/main Translation-en [8.980 B]
Descargados 300 kB en 2s (127 kB/s)
Leyendo lista de paquetes... Hecho
mcardenas@mcardenas:~$ _
```

Instala varias versiones del compilador, por ejemplo:

- **sudo apt-get install gcc-7 g++-7 gcc-8 g++-8 gcc-9 g++-9**

```
mcardenas@mcardenas:~$ sudo apt-get install gcc-7 g++-7 gcc-8 g++-8 gcc-9 g++-9
```

Utiliza los siguientes comandos para definir las alternativas disponibles del compilador:

- **sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90 --slave /usr/bin/g++ g++ /usr/bin/g++-9 --slave /usr/bin/gcov gcov /usr/bin/gcov-9**
- **sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-8 80 --slave /usr/bin/g++ g++ /usr/bin/g++-8 --slave /usr/bin/gcov gcov /usr/bin/gcov-8**
- **sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 70 --slave /usr/bin/g++ g++ /usr/bin/g++-7 --slave /usr/bin/gcov gcov /usr/bin/gcov-7**

```
mcardenas@mcardenas:~$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90 --slave /usr/bin/g++ g++ /usr/bin/g++-9 --slave /usr/bin/gcov gcov /usr/bin/gcov-9
[sudo] password for mcardenas:
update-alternatives: utilizando /usr/bin/gcc-9 para proveer /usr/bin/gcc (gcc) en modo automático
mcardenas@mcardenas:~$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-8 80 --slave /usr/bin/g++ g++ /usr/bin/g++-8 --slave /usr/bin/gcov gcov /usr/bin/gcov-8
mcardenas@mcardenas:~$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 70 --slave /usr/bin/g++ g++ /usr/bin/g++-7 --slave /usr/bin/gcov gcov /usr/bin/gcov-7
mcardenas@mcardenas:~$
```

Finalmente, configura las alternativas disponibles: **sudo update-alternatives --config gcc**

```
mcardenas@mcardenas:~$ sudo update-alternatives --config gcc
Existen 3 opciones para la alternativa gcc (que provee /usr/bin/gcc).
```

Selección	Ruta	Prioridad	Estado
* 0	/usr/bin/gcc-9	90	modo automático
1	/usr/bin/gcc-7	70	modo manual
2	/usr/bin/gcc-8	80	modo manual
3	/usr/bin/gcc-9	90	modo manual

Pulse <Intro> para mantener el valor por omisión [\*] o pulse un número de selección: █

### Parte 3: Crea otro programa en C (obligatorio)

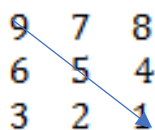
En este PDF tenéis los pasos que debéis seguir para **instalar gcc** en vuestra distribución de Linux. El objetivo es que sepáis **instalar el compilador y que compiléis vuestro primer programa en C**. En este documento os propongo un primer programa muy sencillo con un **print "Hola Pitter"** para que **probéis vuestra instalación**.

Cuando comprobéis que todo va bien, y como no tengo idea de quién es *Pitter*, por favor, cread otro programa en C que haga lo siguiente:

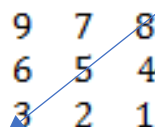
- Que pida al usuario el número de filas de la matriz.
- Que pida al usuario el número columnas de la matriz.
- Que genere valores aleatorios para la matriz en función del número de filas y columnas que indique el usuario.
- Que imprima por pantalla la matriz con los valores generados, usando la forma de matriz (A) (tal cual se aprecia con forma de matriz).
- **Además**, si la matriz es cuadrada, debe imprimirla de la forma 1 a 6.
- **Sólo si** la matriz no es cuadrada, debe imprimirla de la forma 2 y 3.

Impresión de la matriz:

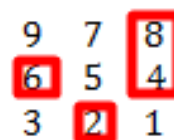
1. Los valores de la diagonal principal.
2. Los valores de la diagonal secundaria.
3. Los valores pares.
4. Los valores impares.
5. Los valores de la diagonal superior y
6. Los valores de la diagonal inferior



(1) Output: 9, 5, 1

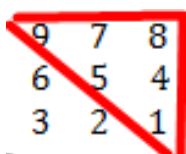


(2) Output: 8, 5, 3

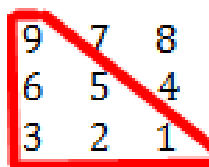


(3) Output: 8, 6, 4, 2

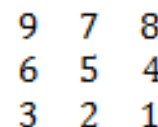
(4) Output: 9, 7, 5, 3, 1



(5) Output: 9, 7, 8, 5, 4, 1



(6) Output: 9, 6, 5, 3, 2, 1



(A) Imprimir la matriz

Os recuerdo, las matrices se pueden crear así:

```
int a[8][8] = { {1,2,3,4,5,6,7,8}, {9,8,7,6,5,4,3,2}, {2,4,4,6,2,7,9,2}, ...};
```



Retos propuestos en esta práctica:

1. Instalar el compilador
2. Crear un programa en C, compilarlo y darle al profesor un código válido que haga lo descrito en el enunciado.
3. Que cada integrante de la práctica con su distribución de Linux haga las pruebas pertinentes.

Entrega:

- Subir lo siguiente como entrega:
  - El código en un fichero “.C” compilado y validado **(si falla la compilación y/o ejecución se suspende el laboratorio)**.