

Entornos de desarrollo

Práctica 2: Lenguajes compilados e interpretados

Esta práctica tiene como objetivo conocer diferentes lenguajes de programación, y aprender de forma práctica su naturaleza ejecutando programas sencillos: si son compilados interpretados, qué compilador/intérprete se usa...


1. Crea un programa que muestre por pantalla la frase **“Madre mía, cómo me gusta la asignatura de ENTORNOS DE DESARROLLO :)”** en cada uno de los siguientes lenguajes.

C++



```
Open  Hola.cpp  Save  ~/Desktop/Practica 2
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Madre mía, cómo me gustas la asignatura de ENTORNOS DE
    DESARROLLO :)\\n";
6     return 0;
7 }
8
```

Código escrito en la sintaxis de C++

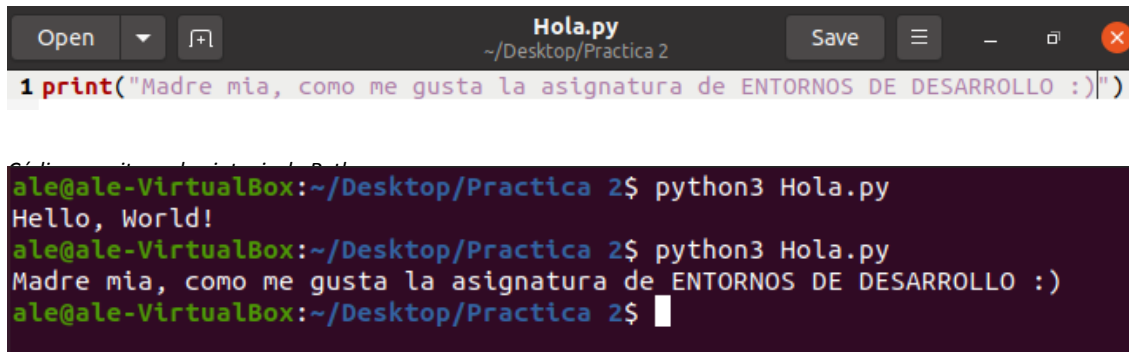


```
ale@ale-VirtualBox: ~/Desktop/Practica 2
ale@ale-VirtualBox:~/Desktop/Practica 2$ g++ -o ejecutable Hola.cpp
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
ejecutable  Hola.cpp
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./ejecutable
Madre mía, cómo me gustas la asignatura de ENTORNOS DE DESARROLLO :)
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Ejecución del programa.

C++ es un lenguaje compilado, como vemos con el ejecutable que ha creado al poner el comando “ls”, sus archivos usan la extensión **.cpp** y el nombre del compilador que he usado para compilar el código es **G++**.

PYTHON



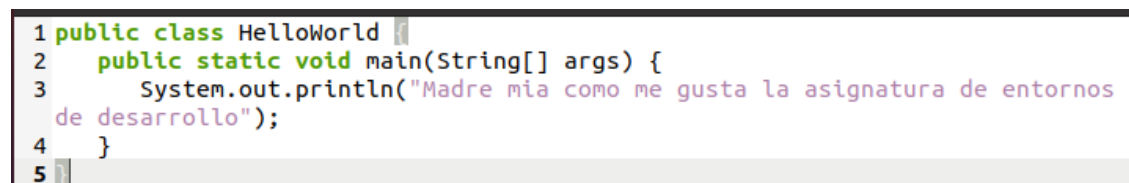
The image shows a code editor window titled 'Hola.py' with the file path '~/.Desktop/Practica 2'. The code in the editor is a single line: `1 print("Madre mia, como me gusta la asignatura de ENTORNOS DE DESARROLLO :)")`. Below the editor is a terminal window with the following commands and output:

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ python3 Hola.py
Hello, World!
ale@ale-VirtualBox:~/Desktop/Practica 2$ python3 Hola.py
Madre mia, como me gusta la asignatura de ENTORNOS DE DESARROLLO :)
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Ejecución del programa.

Python es un lenguaje interpretado, sus archivos usan la extensión **.py** y el nombre del compilador que he usado para compilar el código es **Python 3**.

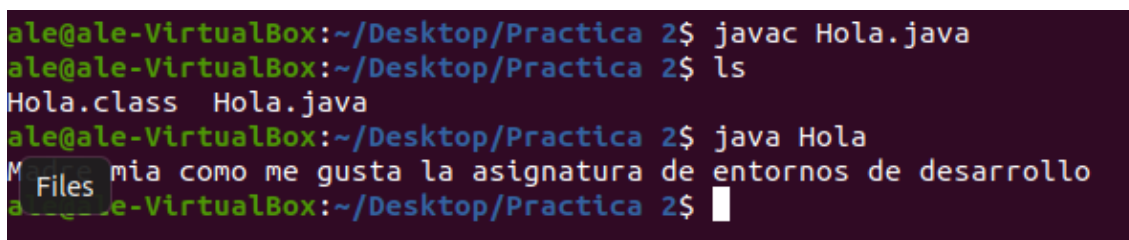
JAVA



The image shows a code editor window with the following Java code:

```
1 public class HelloWorld {
2     public static void main(String[] args) {
3         System.out.println("Madre mia como me gusta la asignatura de entornos
4         de desarrollo");
5     }
6 }
```

Código escrito en la sintaxis de JAVA.



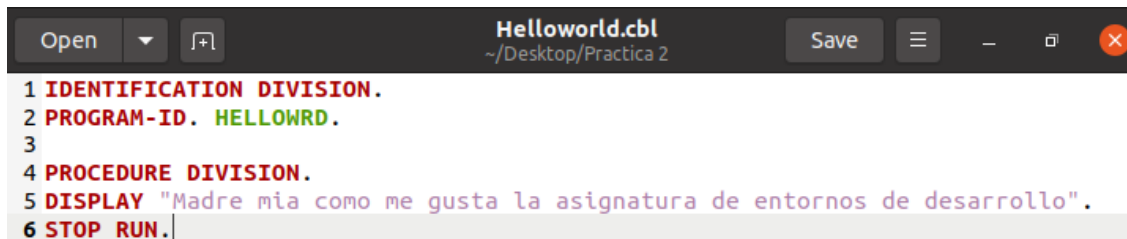
The image shows a terminal window with the following commands and output:

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ javac Hola.java
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
Hola.class  Hola.java
ale@ale-VirtualBox:~/Desktop/Practica 2$ java Hola
M
Files  mia como me gusta la asignatura de entornos de desarrollo
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Ejecución del programa.

Java es un caso peculiar porque es un lenguaje interpretado y compilado a la vez, sus archivos usan la extensión **.java** y el compilador/intérprete se llama **JDK**.

COBOL



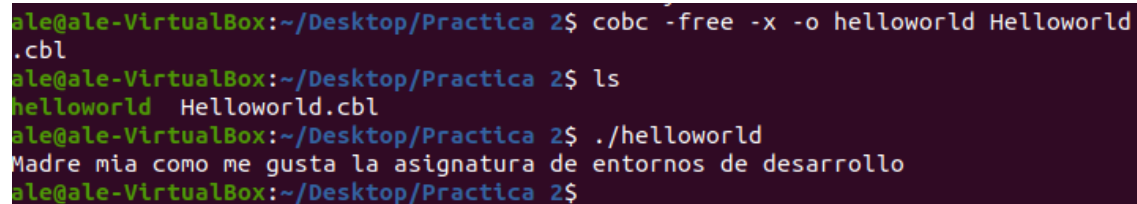
```
Open  ▾  [+]
```

Helloworld.cbl
~/Desktop/Practica 2

Save ≡ — □ ✕

```
1 IDENTIFICATION DIVISION.  
2 PROGRAM-ID. HELLOWRD.  
3  
4 PROCEDURE DIVISION.  
5 DISPLAY "Madre mia como me gusta la asignatura de entornos de desarrollo".  
6 STOP RUN.
```

Código escrito en la sintaxis de COBOL. (madre mía)

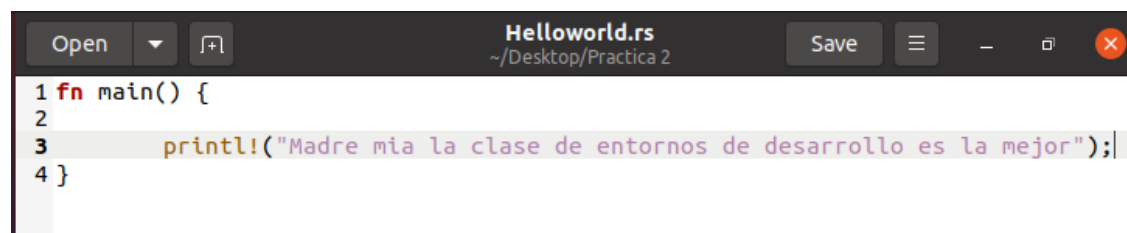


```
ale@ale-VirtualBox:~/Desktop/Practica 2$ cobc -free -x -o helloworld Helloworld  
.cbl  
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls  
helloworld  Helloworld.cbl  
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./helloworld  
Madre mia como me gusta la asignatura de entornos de desarrollo  
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Ejecución del programa.

COBOL es un lenguaje compilado, sus archivos usan la extensión **.cbl** y el compilador es **Open Cobol**.

RUST



```
Open  ▾  [+]
```

Helloworld.rs
~/Desktop/Practica 2

Save ≡ — □ ✕

```
1 fn main() {  
2  
3     println!("Madre mia la clase de entornos de desarrollo es la mejor");  
4 }
```

Código escrito en la sintaxis de RUST.

```

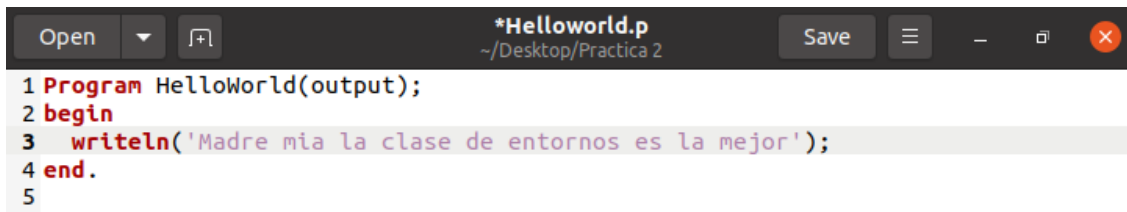
ale@ale-VirtualBox:~/Desktop/Practica 2$ rustc Helloworld.rs
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
Helloworld  Helloworld.rs
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./Helloworld
Madre mia la clase de entornos de desarrollo es la mejor
ale@ale-VirtualBox:~/Desktop/Practica 2$

```

Ejecución del programa.

Rust es un lenguaje compilado cuya extensión es **.rs** y el compilador usado se llama **rustc**.

PASCAL



```

1 Program HelloWorld(output);
2 begin
3   writeln('Madre mia la clase de entornos es la mejor');
4 end.
5

```

Código escrito en la sintaxis de PASCAL.

```

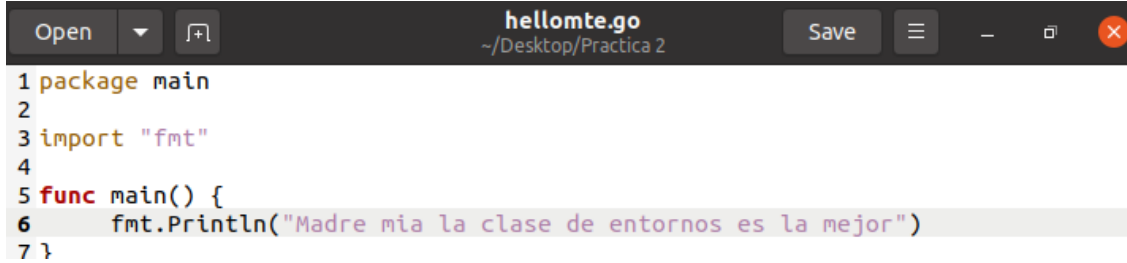
ale@ale-VirtualBox:~/Desktop/Practica 2$ pc Helloworld.p
Free Pascal Compiler version 3.0.4+dfsg-23 [2019/11/25] for x86_64
Copyright (c) 1993-2017 by Florian Klaempfl and others
Target OS: Linux for x86-64
Compiling Helloworld.p
Linking HelloWorld
/usr/bin/ld.bfd: warning: link.res contains output sections; did you forget -T?
5 lines compiled, 0.1 sec
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
Helloworld  Helloworld.o  Helloworld.p
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./Helloworld
Madre mia la clase de entornos es la mejor
ale@ale-VirtualBox:~/Desktop/Practica 2$

```

Ejecución del programa. (mucho texto)

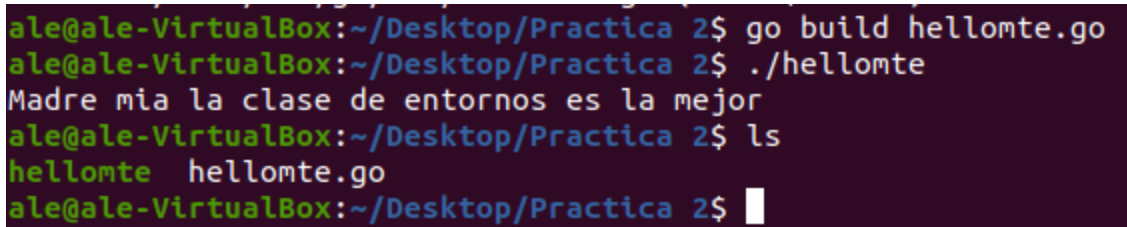
Pascal es un lenguaje compilado, su extensión es **.p** y el compilador usado se llama **FP compiler**.

GO



```
1 package main
2
3 import "fmt"
4
5 func main() {
6     fmt.Println("Madre mia la clase de entornos es la mejor")
7 }
```

Código escrito en la sintaxis de GO.

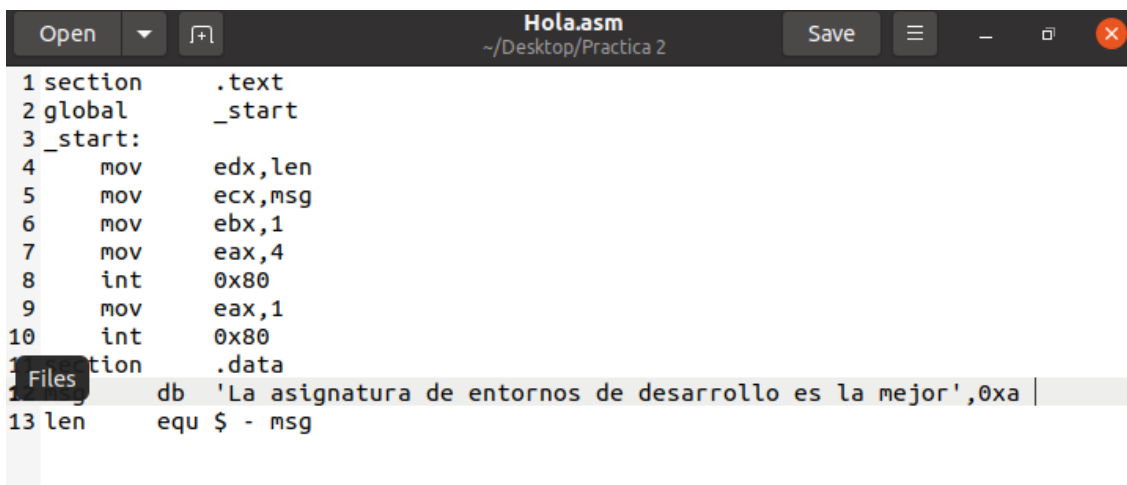


```
ale@ale-VirtualBox:~/Desktop/Practica 2$ go build hellomte.go
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./hellomte
Madre mia la clase de entornos es la mejor
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
hellomte  hellomte.go
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Ejecución del programa.

GO es un lenguaje compilado, la extensión de sus archivos es **.go** y el compilador que he usado en este caso es **golang**.

ENSAMBLADOR (NASM)



```
1 section      .text
2 global      _start
3 _start:
4     mov     edx,len
5     mov     ecx,msg
6     mov     ebx,1
7     mov     eax,4
8     int     0x80
9     mov     eax,1
10    int     0x80
11    section   .data
12    msg      db 'La asignatura de entornos de desarrollo es la mejor',0xa
13 len        equ $ - msg
```

Código escrito en la sintaxis de NASM.

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ nasm -f elf64 Hola.asm
ale@ale-VirtualBox:~/Desktop/Practica 2$ ld -s -o Hola Hola.o
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
Hola  Hola.asm  Hola.o
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./Hola
La asignatura de entornos de desarrollo es la mejor
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Ejecución del programa.

Los archivos usan la extensión **.asm**, es lo más compilado que puede ser algo y el programa necesario para ejecutar sus archivos correctamente es **NASM** e **ID**.

LUA



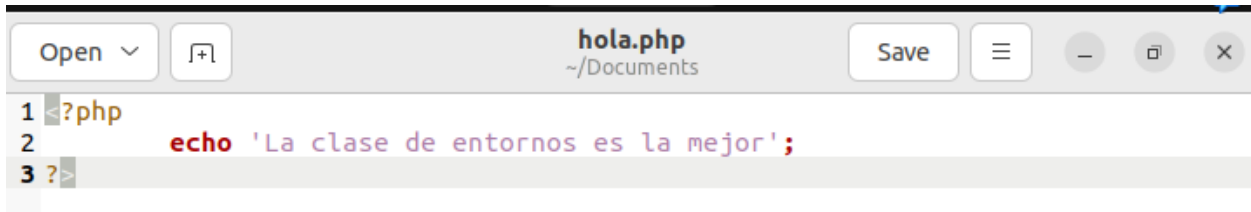
Código escrito en la sintaxis de LUA.

```
ale@ale-VirtualBox:~/Documents/1$ lua hola.lua
La clase de entornos de desarrollo es la mejor
ale@ale-VirtualBox:~/Documents/1$
```

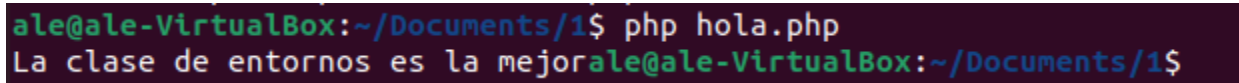
Ejecución del programa.

Lua es un lenguaje interpretado, sus archivos usan la extensión **.lua** y el intérprete que he usado es **Lua5.1**.

PHP



Código escrito en la sintaxis de PHP.



Ejecución del programa.

PHP es un lenguaje interpretado, la extensión de sus archivos es **.php** y el intérprete es **php-cli**.

1.1 Una vez acabes, respóndeme a estas preguntas: ¿Ves mucha diferencia entre los lenguajes de programación que has investigado? ¿Hay alguno que te haya parecido diferente al resto? ¿Hay alguno cuya sintaxis prefieras: ¿ya sea porque es más fácil de entender, más corto...?

No veo demasiada diferencia entre unos y otros (quitando dos excepciones claras), al final la mayoría simplemente usa su propia sintaxis y ya está, algunos son más complejos que otros, o al menos requieren escribir más código para lo mismo, pero en general todos siguen las mismas estructuras. La excepción a eso es Python, que usa un código extremadamente simple pero efectivo, y su contraparte (quitando el ensamblador) para mi ha sido COBOL, que sin que necesariamente sea muchísimo código, me hace sentir como que estoy literalmente escribiendo como un Robot y se me hace un poco antinatural. Así que creo que justo esos dos serían el que más y menos prefiero, al final Python cumple todos los requisitos para ser el favorito, una sintaxis corta, que a su vez, al ser corta, es generalmente más fácil de entender y de comprender, tanto a simple vista como a nivel general. Teniendo esto en cuenta, para mi sería el ganador si tuviese que elegir uno.

2. Práctica en C#.

1. Usaremos la línea de comandos para compilar el programa **mcs HolaUser.cs** (instala el compilador si no existe)

Usamos `sudo apt install mono-complete` para instalar el compilador

2. Vuelve a compilar el programa

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ mcs -out:hola.exe HolaUser.cs
ale@ale-VirtualBox:~/Desktop/Practica 2$ ls
hola.exe  HolaUser.cs
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Comprobación con ls para ver el ejecutable.

3. Ahora si ejecuta el programa: **mono HolaUser** o **./HolaUser.exe**

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ mono hola.exe
Qué tal amigo, estamos en la increíble clase de Entornos de desarrollo, ¿cómo t
e llamas?
Ale
Bienvenido Ale que pases un bien día
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

(Está puesto con la frase del ejercicio siguiente porque antes se me olvidó escribir mi nombre)

4. Cambia el mensaje que muestra tu programa a “¿Qué tal amigo, estamos en la **increíble** clase de Entornos de Desarrollo, ¿cómo te llamas?”

```
1 using System;
2
3 class HolaUser
4 {
5     static void Main(string[] args)
6     {
7         Console.WriteLine("¿Qué tal amigo, estamos en la increíble clase de
Entornos de desarrollo, ¿cómo te llamas?");
8         String nombre = Console.ReadLine();
9         Console.WriteLine("Bienvenido " + nombre + " que pases un bien día");
10    }
11 }
12
```


5. Ejecuta el programa sin compilar: ¿Qué sucede?

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./hola.exe
Hola qué tal amigo mio, ¿cómo te llamas?
Ale
Bienvenido Ale que pases un bien día
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Al ejecutarlo sin compilar, se ejecuta el .exe que teníamos creado anteriormente sin que los cambios que hemos hecho en el código tengan ningún efecto.

6. ¿Cómo lo solucionamos?

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ mcs -out:hola.exe HolaUser.cs
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./hola.exe
Hola qué tal amigo estamos en la increíble clase de Entornos de Desarrollo, ¿cómo te llamas?
Ale
Bienvenido Ale que pases un bien día
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

Compilando el código de nuevo y ejecutando el programa.

7. Ahora modifica el programa anterior, añadiendo las instrucciones necesarias para que le pida también su edad, y la muestre por pantalla.

```
1 using System;
2
3 class HolaUser
4 {
5     static void Main(string[] args)
6     {
7         Console.WriteLine("Hola qué tal amigo estamos en la increíble clase
de Entornos de Desarrollo, ¿cómo te llamas?");
8         String nombre = Console.ReadLine();
9         Console.WriteLine("Bienvenido " + nombre + " que pases un bien día");
10        Console.WriteLine("¿Cuál es tu edad?");
11        String edad = Console.ReadLine();
12        Console.WriteLine("Me alegra saber que tienes " + edad + " años");
13    }
14 }
```

8. Compíllalo y ejecuta.

```
ale@ale-VirtualBox:~/Desktop/Practica 2$ ./hola.exe
Hola qué tal amigo estamos en la increíble clase de Entornos de Desarrollo, ¿cómo te llamas?
Ale
Bienvenido Ale que pases un bien día
¿Cual es tu edad?
24
Me alegra saber que tienes 24 años
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

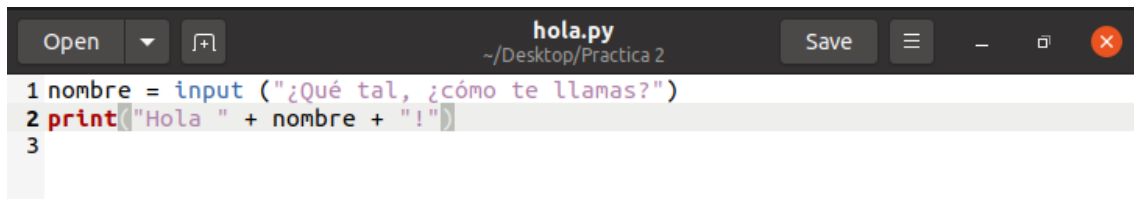
9. Borra los nuevos archivos que generó la compilación del paso anterior.
10. Compíllalo y haz que el nuevo programa compilado lo guarde en otra ruta diferente. Puedes hacerlo desde la terminal, o moviendo el ejecutable cuando lo hayas compilado.
11. Ejecútalo desde la ruta donde lo generaste.

```
ale@ale-VirtualBox:~$ ./hola.exe
Hola qué tal amigo estamos en la increíble clase de Entornos de Desarrollo, ¿cómo te llamas?
Ale
Bienvenido Ale que pases un bien día
¿Cual es tu edad?
24
Me alegra saber que tienes 24 años
ale@ale-VirtualBox:~$
```

A pesar de mover el .exe a otra carpeta, el programa sigue funcionando perfectamente, ya que a diferencia de un lenguaje interpretado, lo único que necesita es el ejecutable, da igual si el código es modificado posteriormente, o si están en carpetas o directorios diferentes.

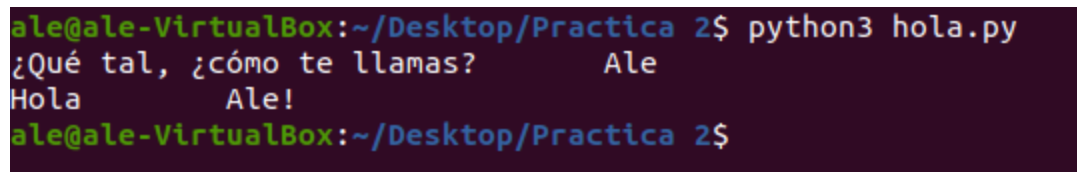
3. Práctica en PYTHON.

1. Crearemos un primer programa que escribirá por pantalla “¿Qué tal, ¿cómo te llamas?” y recuperará del teclado el nombre del usuario, mostrándolo por pantalla y devolviendo un “Hola! *usuario*” siendo el nombre de archivo HolaUser.py



```
hola.py
~/Desktop/Practica 2
1 nombre = input ("¿Qué tal, ¿cómo te llamas?")
2 print("Hola " + nombre + "!")
3
```

2. Utiliza el intérprete de Python para ejecutar el programa.



```
ale@ale-VirtualBox:~/Desktop/Practica 2$ python3 hola.py
¿Qué tal, ¿cómo te llamas?      Ale
Hola      Ale!
ale@ale-VirtualBox:~/Desktop/Practica 2$
```

3. Ahora modifica el programa para que, además, pida la edad del usuario y la muestre por pantalla.

```
1 nombre = input ("¿Qué tal, ¿cómo te llamas?")
2 print("Hola " + nombre + "!")
3 edad = input ("Dime también tu edad")
4 print("TU edad es" + edad)
```

4. Vuelve a interpretar el código, ¿Qué diferencia hay con C#?

La principal diferencia es que lo único que tienes que hacer para que los cambios se apliquen es guardar el archivo de texto donde lo has escrito, es decir, no tienes que realizar el proceso de compilación cada vez que haces un cambio como en C# y simplemente puedes guardar y ejecutar directamente y podrás ver reflejados todos los cambios en el código que hayas hecho.

Bibliografía:

[Compilar C++ en G++ Linux en terminal - lección-1 - HeTPro/Tutoriales \(hetpro-store.com\)](#)

[command line - How can I compile, run and decompile C# code in Ubuntu terminal? - Ask Ubuntu](#)

[Lua Hello World Example: How To Write and Execute Lua Program on Linux OS \(thegeekstuff.com\)](#)

[Learn Python - Free Interactive Python Tutorial](#)

[12.04 - Where is the GNU Pascal Compiler? - Ask Ubuntu](#)

[Cobol Hello World Example: How To Write, Compile and Execute Cobol Program on Linux OS \(thegeekstuff.com\)](#)

[Simple PHP 'Hello, World!' Program \(scriptverse.academy\)](#)