Entornos de desarrollo  
Práctica 2: Lenguajes compilados e interpretados

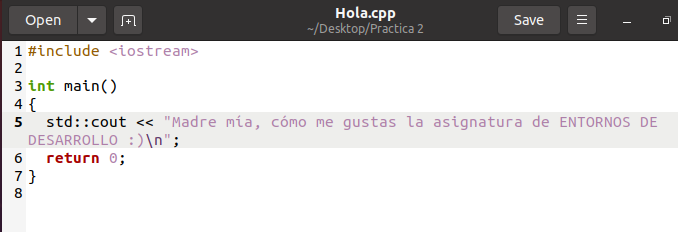
línea horizontal

###### 

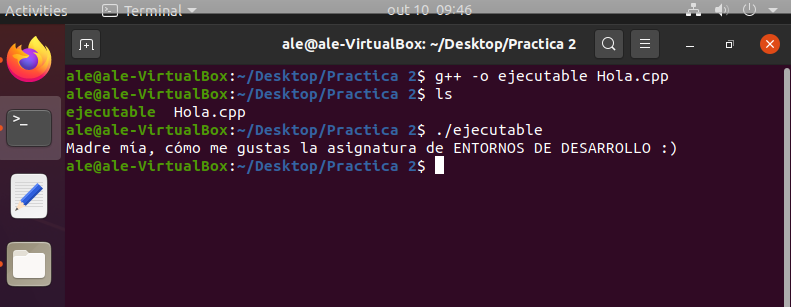
###### Esta práctica tiene como objetivo conocer diferentes lenguajes de programación, y aprender de forma práctica su naturaleza ejecutando programas sencillos: si son compilados interpretados, qué compilador/intérprete se usa…

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++



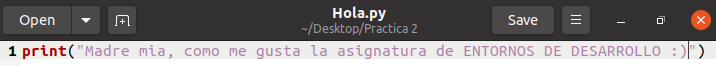
*Código escrito en la sintaxis de C++*

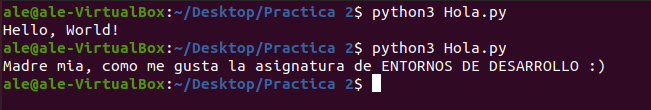


*Ejecución del programa.*

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

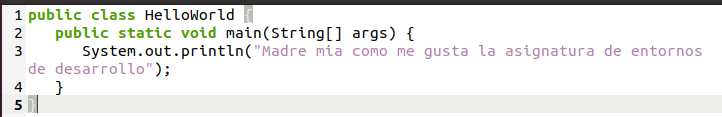
**

*Código escrito en la sintaxis de Python.*

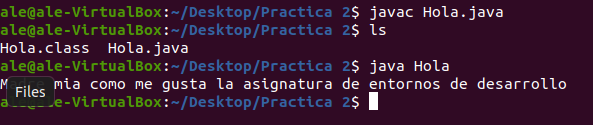
*Ejecución del programa.*

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



*Código escrito en la sintaxis de JAVA.*



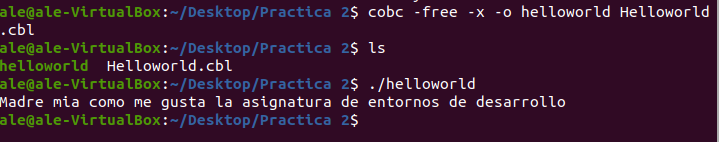
*Código escrito en la sintaxis de JAVA.*

*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

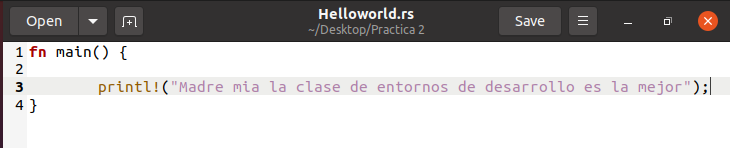
*Código escrito en la sintaxis de COBOL. (madre mía)*



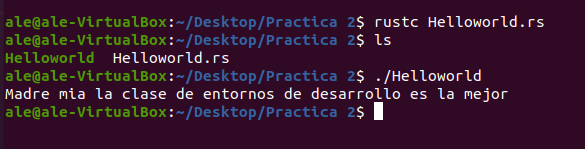
*Ejecución del programa.*

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

###### RUST



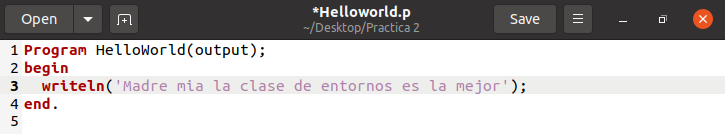
*Código escrito en la sintaxis de RUST.*



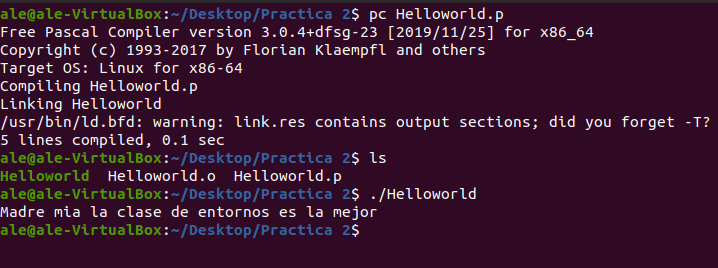
*Ejecución del programa.*

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

###### PASCAL



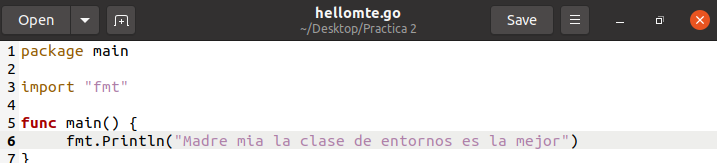
*Código escrito en la sintaxis de PASCAL.*



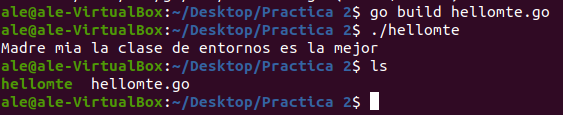
*Ejecución del programa. (mucho texto)*

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

###### GO



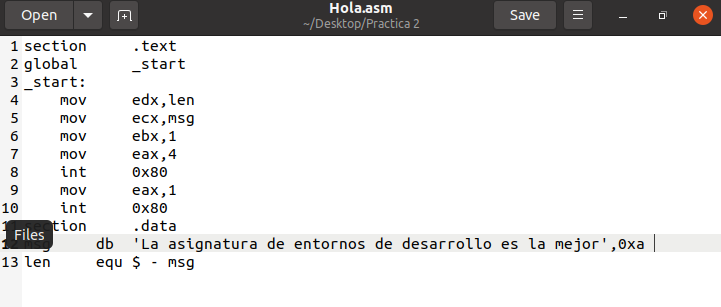
*Código escrito en la sintaxis de GO.*



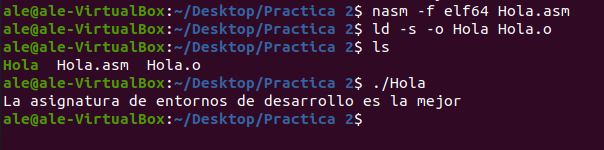
*Ejecución del programa.*

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)



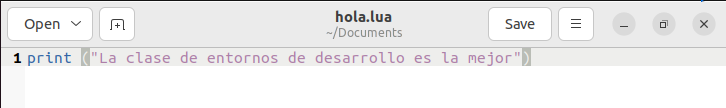
*Código escrito en la sintaxis de NASM.*



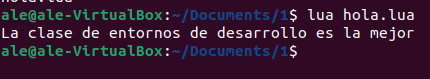
*Ejecución del programa.*

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



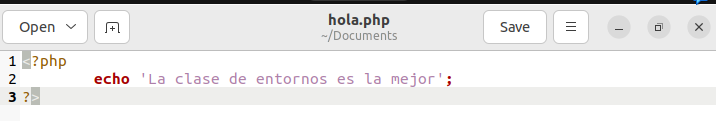
*Código escrito en la sintaxis de LUA.*

**

*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…?**

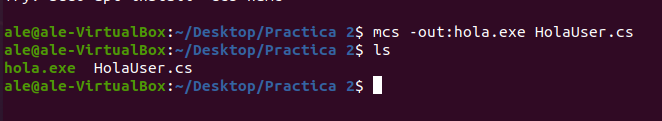
No veo demasiada diferencia entre unos y otros (quitando dos excepciones claras), al final la mayoría simplemente usa su propia sintaxis y ya está, algunos son más complejos que otros, o al menos requieren escribir más código para lo mismo, pero en general todos siguen las mismas estructuras. La excepción 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.

1. **Práctica en C#.**

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

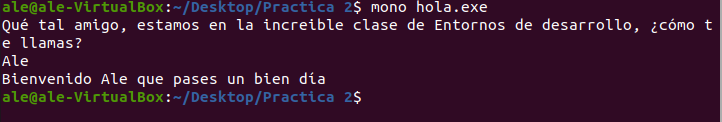
Usamos sudo apt install mono-complete para instalar el compilador

2. Vuelve a compilar el programa



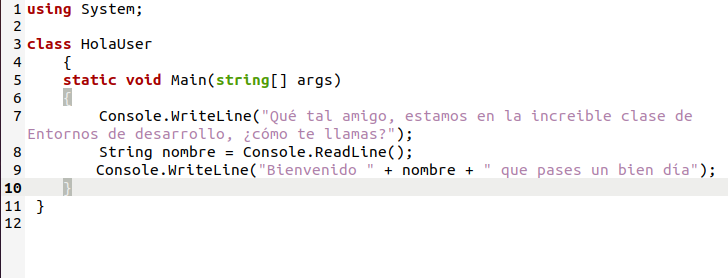
*Comprobación con ls para ver el ejecutable.*

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

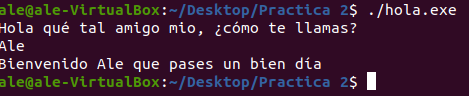
******

*(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?”

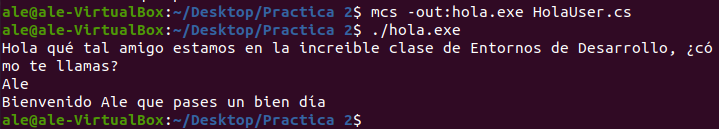


5. Ejecuta el programa sin compilar: ¿Qué sucede?



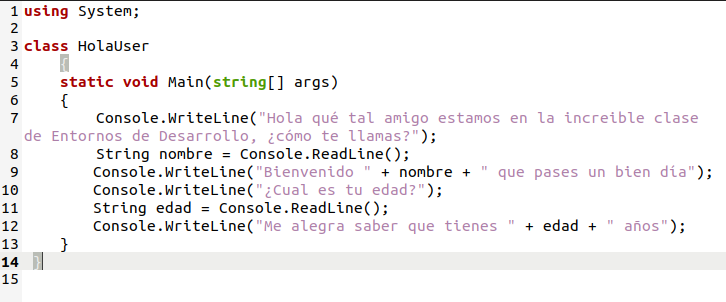
Al ejecutarlo sin compilar, se ejecuta el .exe que teníamos creado anteriormente sin que los cambios que hemos hecho en el código tengan ningún efecto.

6. ¿Cómo lo solucionamos?

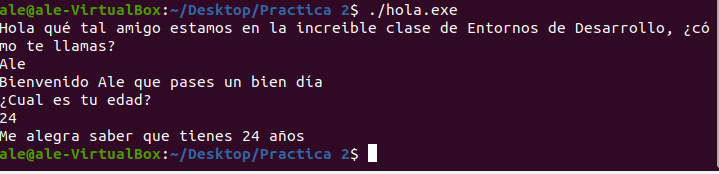


Compilando el código de nuevo y ejecutando el programa.

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



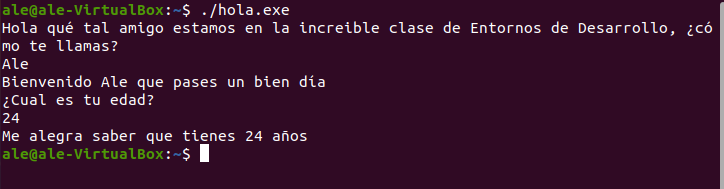
8. Compílalo y ejecuta.



9. Borra los nuevos archivos que generó la compilación del paso anterior.

10. Compílalo 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.

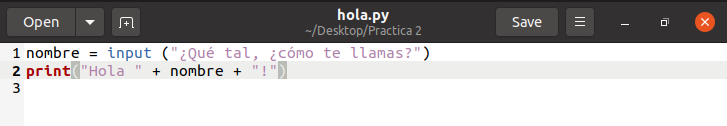
11. Ejecútalo desde la ruta donde lo generaste.



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.

1. **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



## 2. Utiliza el intérprete de Phyton para ejecutar el programa.

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

## 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)](https://hetpro-store.com/TUTORIALES/compilar-cpp-g-linux-en-terminal-leccion-1/#:~:text=Para%20Compilar%20C%2B%2B%20en%20G%2B%2B%20Linux%20en%20terminal,para%20compilar%20se%20usa%20la%20l%C3%ADnea%20de%20comandos.)

[command line - How can I compile, run and decompile C# code in Ubuntu terminal? - Ask Ubuntu](https://askubuntu.com/questions/1100537/how-can-i-compile-run-and-decompile-c-code-in-ubuntu-terminal)

[Lua Hello World Example: How To Write and Execute Lua Program on Linux OS (thegeekstuff.com)](https://www.thegeekstuff.com/2009/12/lua-hello-world-example-how-to-write-and-execute-lua-program-on-linux-os/)

[Learn Python - Free Interactive Python Tutorial](https://www.learnpython.org/)

[12.04 - Where is the GNU Pascal Compiler? - Ask Ubuntu](https://askubuntu.com/questions/236706/where-is-the-gnu-pascal-compiler)

[Cobol Hello World Example: How To Write, Compile and Execute Cobol Program on Linux OS (thegeekstuff.com)](https://www.thegeekstuff.com/2010/02/cobol-hello-world-example-how-to-write-compile-and-execute-cobol-program-on-linux-os/)

[Simple PHP 'Hello, World!' Program (scriptverse.academy)](https://scriptverse.academy/tutorials/php-hello-world.html)