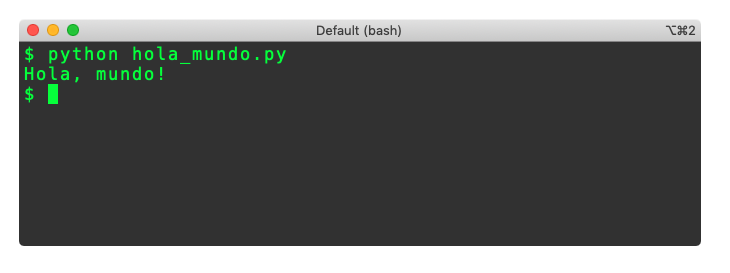
# Qué significa y qué se necesita para programar

Programar es la acción de escribir instrucciones para que un computador las ejecute. Por ejemplo, el siguiente es el primer *programa* que la mayoría de programadores conoce:

print('Hola, mundo!')

Aunque sencillo, este programa nos permite ilustrar varias ideas importantes. En primer lugar, este programa se escribió usando el *lenguaje de programación* Python. Así como en un lenguaje como el español tenemos reglas de gramática y un conjunto de palabras para combinar (vocabulario), un lenguaje de programación tiene reglas de sintaxis, una colección de librerías con programas que se pueden reutilizar, y unas herramientas que permiten ejecutar los programas que se escriban con ese lenguaje. En el caso de nuestro ejemplo, estamos siguiendo las reglas de sintaxis de Python, usamos usando una función llamada print, y usaremos el intérprete [[1]](#footnote-1) de Python para ejecutar el programa [[2]](#footnote-2).

Cuando alguien decide *ejecutar* (o correr) este programa, el computador va a mostrar en la pantalla un mensaje que dice Hola, mundo! (ver imagen). Sin embargo, con algo de práctica y estudio podremos construir programas mucho más avanzados que, por ejemplo, calculen estadísticas a partir de los datos de un censo, analicen una imagen de una resonancia magnética para encontrar un tumor, reproduzcan un video, o sean capaces de sugerir el mejor movimiento para hacer en un juego de ajedrez.



Programar no es una actividad difícil, pero sí es una actividad compleja que involucra varios elementos y habilidades complementarias. La más importante posiblemente sea la habilidad para *resolver problemas*, que en realidad es una combinación de otras habilidades como comprensión de lectura, abstracción, descomposición, razonamiento abstracto, y creatividad. Con esto queremos decir que, para programar, es necesario ser capaz de resolver problemas independientemente de que haya tecnología y computadores involucrados.

El segundo componente para resaltar es el *pensamiento algorítmico*, el cual hace referencia a una cierta forma de aproximarse a la resolución de problemas. El término *algoritmo* hace referencia a las instrucciones para resolver un determinado problema. Dos ejemplos comunes son las instrucciones (el algoritmo) para fritar un huevo y las instrucciones que se encuentran en el envase de un Shampoo. Pensamiento algorítmico hace referencia a la habilidad para describir cómo se debe llegar a la solución de un problema en términos de una serie de instrucciones que alguien más pueda repetir.

En nuestras vidas resolvemos problemas todo el tiempo (manejar un carro, entrar a nuestra oficina, pagar los recibos de servicios públicos, organizar nuestra casa), pero no necesariamente somos capaces de explicarle a alguien más cada uno de estos procesos con el nivel de detalle suficiente para que lo pueda hacer exactamente igual. Cuando empecemos a programar nos daremos cuenta que uno de los obstáculos más grandes va a ser vencer la tentación de darle al computador órdenes demasiado gruesas y en cambio darle órdenes mucho más pequeñas que permitan ir resolviendo el problema (en lugar de decirle a alguien 'frita un huevo', le vamos a tener que decir 'prende el fogón de la estufa, pon una cacerola con mantequilla, espera a que se derrita la mantequilla, ...' )

Programar requiere también conocer al menos un *lenguaje de programación* para poder escribir las instrucciones para el computador. Posiblemente aprender un lenguaje de programación sea la parte más sencilla de todo el proceso, pero es la que fácilmente se confunde con la parte central de aprender a programar. Esto es análogo a pensar que para pintar al óleo lo más importante es aprender los nombres de los colores y conocer la técnica para usar los pinceles: seguramente alguien que no tenga esos conocimientos básicos encontrará dificultades para pintar, pero con seguridad otras habilidades y actitudes son más importantes para producir verdaderas obras de arte.

Este libro pretende hacer explícito que aprender a programar no es lo mismo que aprender un lenguaje de programación. Inevitablemente tuvimos que escoger un lenguaje para nuestros ejemplos (Python), pero casi todos los conceptos que veremos podrían aplicarse a otros lenguajes de programación. En lo posible, haremos explícito cuando algo de lo que expliquemos sea exclusivo de Python.

Finalmente, programar requiere una buena disposición hacia la *tecnología* y hacia el *auto-aprendizaje*. En este libro usaremos un conjunto de herramientas y librerías relativamente sencillo que además será explicado en detalle. Estas deberían ser suficientes para construir desde programas triviales hasta programas muy interesantes y útiles. Pero un buen programador debería ser capaz (¡y debería tener muchas ganas!) de aprender a usar nuevas tecnologías por su propia cuenta.

# Prácticas fundamentales

La experiencia de varios años enseñando a programar nos muestra que con una adecuada actitud y compromiso cualquiera puede aprender a programar. Los estudiantes que se aproximan a la programación esperando cumplir con un requisito, sin ver el enorme potencial que podría tener para su desempeño profesional, usualmente tienen menos éxito que aquellos que tienen una mente abierta y curiosa y que se enfrenten a cada reto esperando aprender algo de él.

Además de esto, la experiencia también nos ha mostrado que los estudiantes que realizan las siguientes 3 acciones usualmente tienen mucho más éxito en su proceso de aprendizaje, disfrutan más la experiencia, y siguen aprendiendo a programar más allá de su primer curso.

## Lectura con atención

Para poder programar es necesario leer con atención todo lo relacionado con el problema que se esté solucionando. Es casi imposible construir una buena solución si no se han leído con cuidado las condiciones de lo que se está pidiendo ni las restricciones para la solución.

Muchas veces los problemas a los que se enfrentan los estudiantes que empiezan a programar no tienen que ver con la programación en sí misma, sino que tienen que ver con que el estudiante no entendió el problema que tenía que resolver.

¡Nunca empiece a programar sin entender antes lo que le están preguntando!

## Práctica deliberada y reflexiva

Al igual que cualquier otra actividad basada en habilidades, programar requiere practicar. Así como no se puede tocar aprender a tocar violín o a montar bicicleta leyendo todos los libros disponibles sobre el tema, para aprender a programar se necesita practicar programando.

Más aún, para hacer más eficiente su proceso de aprendizaje, un estudiante de programación debería esforzarse por hacer una práctica *deliberada* y *reflexiva*. Práctica deliberada hace referencia a tener un objetivo específico cuando se practica. Por ejemplo, cuando un futbolista practica no se limita a *jugar futbol*, sino que hace en cada sesión repite ejercicios diseñados para ayudarlo a desarrollar una determinada habilidad. De igual forma, en cada sesión de práctica los pianistas más exitosos definen un objetivo particular (practicar un tipo de técnica, resolver un fragmento de una pieza) en lugar de simplemente sentarse a *tocar piano*.

Por otro lado, práctica *reflexiva* hace referencia al proceso que debería hacer un estudiante al terminar una práctica. En lugar de simplemente dar el ejercicio por terminado, el estudiante debería tomarse al menos un momento para reflexionar sobre lo que hizo, lo que aprendió, los problemas que enfrentó y las conclusiones que se podrían sacar de la experiencia. Se ha visto en muy diversas situaciones que el esfuerzo invertido en este proceso de reflexión hace que la práctica sea mucho más efectiva y termina reduciendo el esfuerzo total que se debe hacer.

Este libro incluye numerosos ejercicios para cada uno de los temas, los cuales han sido seleccionados para ejercitar habilidades particulares. A medida que vaya avanzando, procure resolver los ejercicios haciendo una reflexión sobre lo que haya aprendido al final de ellos.

## Lectura de código

Al igual que un pintor no podría pintar sus propias obras sin haber visto las de otros, o un escritor no podría escribir una novela sin haber leído las de muchos otros, para escribir programas es necesario poder leer programas escritos por otros. Sin embargo, no se trata de hacer una lectura superficial de esos programas, sino de hacer una lectura cuidadosa que nos permita identificar las características y objetivos de cada uno y nos permita confrontar nuestras propias dudas y vacíos en nuestro conocimiento.

A lo largo de este libro encontrará numerosos fragmentos de código que ilustran conceptos particulares. Además, encontrará también programas más largos y complejos que retarán sus habilidades de lectura. ¡Haga el esfuerzo de leer estos programas, así no los entienda completamente en un primer momento! En el nivel 1 le daremos algunas recomendaciones adicionales sobre cómo leer código.

# Sobre este libro

Este libro está pensado para acompañar el curso de **Introducción a la Programación** del [Departamento de Ingeniería de Sistemas](http://sistemas.uniandes.edu.co) de la Universidad de los Andes, el cual está dirigido a estudiantes que nunca hayan programado. Pensando en esto, el libro tiene las siguientes características.

* **Es un libro básico para aprender a programar:** A diferencia de otros libros que le enseñan Python a programadores que ya conozcan otros lenguajes, este libro no supone ningún conocimiento previo de programación. Si usted ya sabe programar en algún otro lenguaje, es posible que este libro sea *demasiado* básico para usted.
* **Libro soporte:** El libro está pensado para *soportar* y aclarar cualquier duda que haya quedado en el aula. Un estudiante del curso debería poder encontrar acá aclaraciones y ejercicios adicionales sobre cualquier tema. Sin embargo, el libro no siempre seguirá el mismo orden que se lleve en el aula de clase: en algunas secciones el libro presentará temas de forma completa desde el inicio, entrando en un mayor nivel de detalle del que normalmente se estudiaría la primera vez que se encontrara ese tema.
* **Python:** El curso está basado en Python, así que el libro también está basado en Python. Sin embargo, el libro hace un esfuerzo por identificar claramente los conceptos centrales que son aplicables a otros lenguajes de programación en lugar de presentarlos como características del lenguaje.
* **Organización:** El libro está organizado siguiendo la estructura del curso y el curso está organizado teniendo en cuenta restricciones y necesidades que son propias de nuestros programas académicos: la cantidad de tiempo disponible, el contenido de los cursos siguientes, los programas en los que están inscritos nuestros estudiantes, y la necesidad de desarrollar competencias algorítmicas por encima de competencias para la estructuración de programas. Esto hace que la organización de este libro sea muy diferente a la de muchos otros libros de programación con Python. Por ejemplo, el tema de programación orientada a objetos, que aparece en los primeros capítulos de otros libros, está completamente ausente en este. Algo similar pasa con el tema de manejo y creación de excepciones.

# Estructura del libro

El libro está organizado en 4 grandes partes que corresponden a los 4 niveles en los que está organizado nuestro curso **Introducción a la Programación**. Los 4 niveles y sus temas principales son los siguientes:

1. **Descubriendo el mundo de la programación**. En esta parte se introducen los conceptos básicos de la programación y se empieza a trabajar el tema del uso de funciones y módulos para descomponer y estructurar adecuadamente los programas.
2. **Tomando decisiones**. En esta parte se introducen las instrucciones condicionales y una estructura de datos que nos permite manejar información un poco más compleja (diccionarios).
3. **Repetir acciones y manejar estructuras de datos de una dimensión**. A partir de esta sección los programas que se pueden construir se vuelven mucho más poderosos porque se estudian las instrucciones repetitivas y se introducen las estructuras de datos lineales.
4. **Solucionar problemas con matrices y librerías.** La última sección del libro se concentra en el uso de una librería muy utilizada para el procesamiento de datos (*Pandas*) y en el uso de matrices.

Los 4 niveles son incrementales: lo que se aprende en un nivel se utiliza y se refuerza en el siguiente, de tal forma que al final todos los temas se hayan estudiado y practicado varias veces.

1. Más adelante vamos a explicar qué significa el intérprete. Por ahora es suficiente con saber que un intérprete es un programa capaz de ejecutar otros programas. En este caso, el intérprete será capaz de ejecutar programas escritos usando Python. [↑](#footnote-ref-1)
2. Si en lugar de Python hubiéramos usado el lenguaje de programación C, el programa que habríamos tenido que escribir sería el siguiente: main( ) { printf("hello, world\n"); }. Además, el programa no lo ejecutaríamos usando el intérprete de Python. [↑](#footnote-ref-2)