Prólogo

Conocer la teoría y la práctica de los lenguajes de programación es básico para muchas de las tareas que un informático ha de afrontar a lo largo de su vida profesional. Casi a cualquier estudiante de informática le gustar programar, pero además debe plantearse algunas preguntas como las siguientes:

- ¿Qué lenguaje elegir para resolver un problema o desarrollar una aplicación determinada?
- ¿Con qué criterios descartar un lenguaje?
- ¿Qué ventajas en general y en concreto ofrece cada uno de ellos?

Pero no sólo en tareas de modelado y desarrollo de paquetes software o programas eficientes, también en otros casos, como por ejemplo si tuviera la oportunidad de diseñar un nuevo lenguaje, es necesario saber responder a preguntas como:

- ¿Qué características deberían tenerse en cuenta durante el diseño?
- ¿Qué abstracciones de datos o control incluir como mínimo?
- ¿Qué le distinguirán de otros lenguajes existentes?

Los autores quieren atender especialmente a la necesidad que tienen los profesionales, estudiantes o curiosos de la informática de conocer los diferentes paradigmas de la programación, con una orientación fundamentalmente práctica. Se espera que el lector del libro tenga al menos unas nociones mínimas sobre la programación imperativa y sobre la orientada a objetos, pero casi con toda seguridad desconoce la programación declarativa. Sin embargo en la historia de los lenguajes de programación, estos lenguajes ocupan una posición especial debida a sus especiales características tanto en lo relacionado a los tipos de datos y estructuras de control que aportan, como en su implementación. Por ello se han incluido dos capítulos dedicados a la programación declarativa y así mostrar cómo resolver y programar soluciones a problemas que, con otros paradigmas, es mas complejo o poco adecuado y viceversa.

El aprendizaje de un lenguaje imperativo u orientado a objetos y uno declarativo supone una riqueza en términos de conocimiento, pero como además, desde casi cualquier punto de vista y no solo el educativo, a un informático se le convence mejor con hechos que con palabras, se ha hecho especial hincapié en presentar casos prácticos y programas en estos lenguajes. Por ello, los ejercicios relacionados con el contenido de este libro, se deberían realizar tanto con un lenguaje declarativo como con uno orientado a objetos, y comprobar las diferencias y similitudes. Ahora bien, estos dos capítulos que incluyen con detalle los fundamentos, características especiales y técnicas de programación eficiente sobre un lenguaje funcional y lógico, pueden ser leídos de forma parcial, si el objetivo principal del lector es unicamente conocer los aspectos más básicos de los lenguajes de programación y su diseño.

Este libro ha sido concebido para los estudiantes de informática en general y en concreto para los de la ETSI de Informática de la Universidad Nacional de Educación a Distancia (UNED). Esperamos que disponer de un libro con el temario completo, ayude tanto a los estudiantes como a los profesores y tutores de la UNED (estos últimos distribuidos a lo largo de la geografía española y algunas ciudades del extranjero). Se hace una mención especial al libro de K. C. Louden [15] en su versión española del 2004 (actualmente descatalogado), uno de los libros clásicos para el estudio de los paradigmas y los lenguajes de programación. Los autores han utilizado en parte su línea de discurso (especialmente en los capítulos 1, 4 y 5), aunque no su organización general.

Además, se ha intentado cumplir con las recomendaciones para la asignatura de Teoría de los Lenguajes de Programación (semestral en un grado de Ingeniería Informática) incluidas, tanto en el libro blanco del título de grado en Ingeniería Informática de la Agencia Nacional de Evaluación de la Calidad y Acreditación (ANECA) española[1], como en las referencias internacionales del *Computer Science Curricula* de la ACM[2] y del *Computer Science Curricula* del IEEE[3].

Aunque en el siguiente apartado se indica con más detalle, el contenido se estructura en torno a seis capítulos. Todos ellos se desarrollan con ejemplos prácticos presentados con cierto detalle y además se incluye un índice analítico al final del libro, para permitir al lector encontrar el lugar concreto en el que revisar los conceptos que aparecen a lo largo del libro.

Los contenidos de cada capítulo se complementan con un conjunto balanceado de problemas con y sin solución, tanto de aplicación casi directa del contenido y de problemas representativos de aspectos específicos, como de resolución o programación de problemas mas complejos. Finalmente, en cada capítulo se incluyen algunas notas bibliográficas con referencias complementarias.

[1] http://www.aneca.es/var/media/150388/libroblanco_jun05_informatica.pdf
[2] http://www.acm.org/education/CS2013-final-report.pdf
[3] http://ai.stanford.edu/users/sahami/CS2013/ironman-draft/cs2013-ironman-v0.8.pdf

Organización del contenido

El contenido del libro está organizado en cuatro partes:

- Una introducción a los principios de diseño de los lenguajes de programación (capítulo 1).
- Los lenguajes de programación declarativos (capítulos 2 y 3).
- La sintaxis y la semántica de los lenguajes de programación no declarativos y los tipos de datos (capítulos 4 y 5).
- La abstracción, las estructuras de control y la organización de los entornos de ejecución (capítulo 6).

El desarrollo de los contenidos se dedica a la presentación de los fundamentos y a la práctica basada en ejemplos. Se detallan a continuación, aunque brevemente, los contenidos que se incluyen en cada capítulo:

- **Capítulo 1. Paradigmas de la computación.** Se presentan y definen conceptos que pueden no ser nuevos para los estudiantes, pero de los que conviene hacer una recapitulación y recordarlos de cara a su comprensión y profundización en capítulos posteriores. Son conceptos relacionados con las abstracciones de datos y de las estructuras de control, que se presentan enmarcados en el estudio de los paradigmas de la computación, aunque se verán con detalle en el capítulo cinco y seis respectivamente. A continuación se introducen diferentes paradigmas de computación y una panorámica de los principios de diseño de los lenguajes de programación.
- **Capítulo 2. Programación Funcional.** En este capítulo se presentan los principios de la programación funcional. Se repasa el concepto matemático de función y la forma en que pueden considerarse los programas como funciones, partiendo del modelo matemático que subyace a la programación funcional. Se da una introducción al modelo de evaluación perezosa, propio de este tipo de lenguajes y, a continuación, se presenta una introducción al lenguaje funcional Haskell, analizando algunas de sus propiedades.
- **Capítulo 3. Programación Lógica.** Tras una una breve introducción a la forma en que puede utilizarse la lógica como un lenguaje de programación, se describe el lenguaje concreto Prolog y algunas técnicas para escribir programas en este lenguaje. A continuación se presentan con ejemplos, algunos de los puntos fuertes y débiles de este lenguaje. El capítulo finaliza con la presentación de la programación lógica en problemas complejos.

- **Capítulo 4. Sintaxis y Semántica Básica.** En primer lugar se presenta con cierto detalle la sintaxis de los lenguajes de programación, mostrando el uso de diferentes herramientas, como BNF, EBNF y los diagramas sintácticos, para la definición formal de las gramáticas de los lenguajes. A continuación se incluyen aspectos semánticos básicos de los lenguajes de programación como son: la declaración, la asignación y evaluación de variables y constantes, el uso de la tabla de símbolos y finalmente aspectos relacionados con el entorno de los símbolos definidos en los programas.

- **Capítulo 5. Tipos de Datos.** Este capítulo incluye una descripción de los tipos de datos y los principales tipos y constructores de tipos disponibles en la mayoría de los lenguajes. Después de analizar los principios básicos, se estudian los conceptos de verificación y equivalencia de tipos de datos, analizando los principales problemas que surgen.

- **Capítulo 6. Control de la Ejecución.** El objetivo global de este capítulo es el conocimiento teórico y práctico de los principios relacionados con las estructuras de control y los entornos de ejecución de los lenguajes de programación. En primer lugar se estudian las expresiones, cuya evaluación permitirá que el programa pueda variar el flujo de ejecución mediante las estructuras de control, aunque pueden provocar diversos problemas de control. Después se tratan los mecanismos de control estructurados como las sentencias condicionales y los bucles. Posteriormente se analiza el manejo de excepciones, un mecanismo de control preventivo, que hace que se altere la estructura de control explícito de un programa. A continuación, se describen los subprogramas, que son bloques de código cuya ejecución se pospone y que cuentan con una interfaz claramente especificada. Finalmente se estudian las principales técnicas de paso de parámetros que llevan a un comportamiento semántico distinto, y se esboza la estructura de los registros de activación y de las tres variedades básicas de los ambientes en tiempo de ejecución.

Como ya se ha indicado en la primera parte de esta introducción, los capítulos tienen contenidos que se pueden leer y estudiar independientemente y casi en cualquier orden. Sin embargo, se sugiere estudiar el capítulo cuarto (la sintaxis y la semántica de los lenguajes de programación) tras el primero (paradigmas de la computación), para afianzar algunos de los conceptos abstractos presentados en aquel. La misma dependencia ocurre entre el capítulo sexto (control de la ejecución de programas) y el primero. El capítulo sexto exige un conocimiento básico de tipos de datos, por lo que es adecuado leer o estudiar primero dicho capítulo (el quinto). En el cuarto capítulo se comparan algunas características de los lenguajes declarativos y no declarativos, por lo que es conveniente haber estudiado, al menos, la primera parte tanto del capítulo segundo como del tercero.

Madrid, enero de 2014.