Podrá descargar algunos elementos de este libro en la página web de Ediciones ENI: **http://www.ediciones-eni.com**. Escriba la referencia ENI del libro **RITHS-43PYT** en la zona de búsqueda y valide. Haga clic en el título y después en el botón de descarga.

Prólogo

1. Introducción . . . 23
2. Contenido del libro . . . 23
3. Progresión del libro . . . 24
4. Destinado a profesores y alumnos . . . 25
5. Destinado a investigadores y doctores . . . 26
6. Destinado a aquellos que vienen de otro lenguaje . . . 27

Parte 1: Las fortalezas de Python

Capítulo 1-1
Claves teóricas

1. Breve historia de los lenguajes informáticos . . . 29
   - 1.1 Informática teórica . . . 29
   - 1.2 Cronología de la informática . . . 30
     - 1.2.1 Evolución de las problemáticas vinculadas a la informática . . . 30
     - 1.2.2 Cronología de los lenguajes informáticos . . . 31
2. Tipología de los lenguajes de programación . . . 35
   - 2.1 Paradigmas . . . 35
     - 2.1.1 Definición . . . 35
     - 2.1.2 Paradigma imperativo y derivados . . . 36
     - 2.1.3 Paradigma orientado a objetos y derivados . . . 37
     - 2.1.4 Programación orientada a aspectos . . . 37
     - 2.1.5 Paradigma funcional . . . 38
     - 2.1.6 Paradigma lógico . . . 38
     - 2.1.7 Programación concurrente . . . 38
     - 2.1.8 Síntesis . . . 39
   - 2.2 Interoperabilidad . . . 39
   - 2.3 Niveles de programación . . . 41
     - 2.3.1 Máquina . . . 41
     - 2.3.2 Bajo nivel . . . 41
     - 2.3.3 Alto nivel . . . 42

2.4 Tipado 43
2.4.1 Débil vs. fuerte 43
2.4.2 Estático vs dinámico 43
2.5 Gramática 43
2.5.1 Lenguajes formales 43
2.5.2 Sintaxis 44
3. Python y el resto del mundo 45
3.1 Posición estratégica del lenguaje Python 45
3.1.1 Segmentos de mercado 45
3.1.2 Nivel de complejidad 45
3.1.3 Fortalezas del lenguaje 45
3.1.4 Puntos débiles 46
3.2 Integración con otros lenguajes 46
3.2.1 Extensiones C 46
3.2.2 Integración de programas escritos en C 47
3.2.3 Integración de programas Python en C 47
3.2.4 Integración de programas escritos en Java 47
3.2.5 Integración de programas Python en Java 47
3.2.6 Otras integraciones 47

Capítulo 1-2
Presentación de Python

1. Filosofía 49
1.1 Python en pocas líneas 49
1.1.1 ¿De dónde proviene el nombre «Python»? 49
1.1.2 Presentación técnica 49
1.1.3 Presentación conceptual 50
1.2 Comparación con otros lenguajes 50
1.2.1 Shell 50
1.2.2 Perl 51
1.2.3 C, C++ 51
1.2.4 Java 52
1.2.5 PHP 54
1.3 Grandes principios 55
1.3.1 El zen de Python 55
1.3.2 El desarrollador no es estúpido 55
1.3.3 Documentación 56
1.3.4 Python viene con todo incluido 56
1.3.5 Duck Typing 56

1.3.6 Noción de código pythónico 57
2. Historia de Python. 57
2.1 La génesis. 57
2.2 Ampliación del perímetro funcional 58
2.3 Evolución de la licencia. 62
2.4 El futuro 62
3. Gobierno. 63
3.1 Desarrollo 63
3.1.1 Ramas 63
3.1.2 Comunidad 64
3.2 Modo de gobierno 64
3.2.1 Creador del lenguaje 64
3.2.2 PEP 65
3.2.3 Toma de decisiones 65
3.2.4 Contribuir a Python 65
4. ¿Qué contiene Python? 66
4.1 Una gramática y una sintaxis 66
4.2 Varias implementaciones 66
4.3 Una librería estándar 66
4.4 Librerías de terceros 67
4.5 Frameworks. 67
5. Fases de ejecución de un programa Python 67
5.1 Carga de la máquina virtual 67
5.2 Compilación 67
5.3 Interpretación 68

Capítulo 1-3
Por qué escoger Python

1. Cualidades del lenguaje 69
1.1 Puerta de entrada 69
1.2 Cualidades intrínsecas 71
1.3 Cobertura funcional 71
1.4 Dominios de excelencia 72
1.5 Garantías 73
2. Difusión 74
2.1 Empresas 74
2.2 El mundo de la investigación 75

2.3 El mundo de la educación 76
2.4 Comunidad 77
3. Referencias 78
3.1 Pesos pesados en la industria informática 78
3.1.1 Google 78
3.1.2 Mozilla 79
3.1.3 Microsoft 79
3.1.4 Canonical 79
3.1.5 Cisco 80
3.2 Empresas de innovación 80
3.2.1 Servicios de almacenamiento en línea 80
3.2.2 Cloud computing 80
3.2.3 Plataforma colaborativa (Forge) 80
3.2.4 Redes sociales 80
3.3 Editores de contenidos 81
3.3.1 Disney Animation Studio 81
3.3.2 YouTube 81
3.3.3 Box ADSL 81
3.3.4 Spotify 81
3.4 Fabricantes de software 81
4. Experiencia 82
4.1 Internet de las cosas 82
4.2 Sistemas y desarrollo web 83
4.3 Enseñanza 84
4.4 Informática embebida 84
4.5 Desarrollo web 85
4.6 ERP 85

Capítulo 1-4
Instalar el entorno de trabajo

1. Introducción 87
2. Instalar Python 87
2.1 Para Windows 87
2.2 Para Mac 90
2.3 Para GNU/Linux et BSD 90
2.4 Mediante compilación 91
2.5 Para un smartphone 92

3. Instalar una librería externa 92
3.1 A partir de Python 3.4 92
3.2 Para una versión inferior a Python 3.4 94
3.3 Para Linux 94
4. Crear un entorno virtual 95
4.1 ¿Para qué sirve un entorno virtual? 95
4.2 Para Python 3.3 o versiones superiores 95
4.3 Para cualquier versión de Python 96
4.4 Para Linux 97
5. Gestión de las dependencias 98
6. Instalar Anaconda 99
6.1 Para Windows 99
6.2 Para Linux 100
6.3 Para Mac 100
6.4 Actualizar Anaconda 100
6.5 Instalar una librería externa 100
6.6 Entornos virtuales 101
7. Docker 101
8. La consola Python 102
8.1 Arrancar la consola Python 102
8.2 BPython 102
8.3 IPython 103
8.4 IPython Notebook 103
9. Instalar un IDE 104
9.1 Lista de IDE 104
9.2 Presentación de PyCharm 104
9.3 Configuración de PyCharm 105
10. VSCode 110

Parte 2: Guía de Python

Capítulo 2-1
Primeros pasos

1. Antes de comenzar 111
1.1 Algunas nociones importantes 111
1.1.1 ¿Cómo funciona un ordenador? 111

1.1.2 ¿Qué es un programa informático?. 112
1.1.3 ¿Qué es el código fuente? 112
1.2 Algunas convenciones utilizadas en este libro 112
1.2.1 Código Python 112
1.2.2 Terminal 113
1.2.3 Formato 113
1.3 ¿Cuál es el mejor método para aprender? 114
2. Primer programa 114
2.1 Hello world!. 114
2.2 Asignación 116
2.3 Valor booleano 117
2.4 Tipo 118
2.5 Excepciones 119
2.6 Bloque condicional 121
2.7 Condiciones avanzadas. 123
2.8 Bloque iterativo 123
3. Primer juego: Adivine el número 125
3.1 Descripción del juego 125
3.2 Pistas 125
3.2.1 Gestión del azar 125
3.2.2 Etapas del desarrollo 126
3.3 Para ir más allá. 126

Capítulo 2-2
Funciones y módulos

1. Las funciones 127
1.1 ¿Por qué utilizar funciones? 127
1.2 Introducción a las funciones 129
1.2.1 Cómo declarar una función 129
1.2.2 Gestión de un parámetro 130
1.2.3 Cómo hacer la función más genérica 132
1.2.4 Parámetros por defecto 134
1.3 Problemáticas de acoplamiento y duplicación de código 135
1.3.1 Nivel de sus funciones 135
1.3.2 Noción de complejidad 137
1.3.3 Buenas prácticas 139

2. Los módulos 140
2.1 Introducción 140
2.1.1 ¿Qué es un módulo? 140
2.1.2 ¿Cómo crear un módulo en Python? 141
2.1.3 Organizar el código 141
2.2 Gestionar el código de los módulos 141
2.2.1 Ejecutar un módulo, importar un módulo 141
2.2.2 Gestionar un árbol de módulos 142
3. Terminar el juego 143
3.1 Crear niveles 143
3.2 Determinar un número máximo de intentos 144
3.3 Registrar las mejores puntuaciones 144
3.4 Inteligencia artificial 144

Capítulo 2-3
Los principales tipos

1. Cadenas de caracteres 145
1.1 Sintaxis 145
1.2 Formato de una cadena 146
1.3 Noción de tamaño de letra 146
1.4 Noción de longitud 147
1.5 Pertenencia 148
1.6 Noción de ocurrencia 148
1.7 Reemplazo 149
1.8 Noción de carácter 149
1.9 Tipología de los caracteres 150
1.10 Secuenciar una cadena de caracteres 151
2. Listas 152
2.1 Sintaxis 152
2.2 Índices 152
2.3 Valores 153
2.4 Azar 154
2.5 Técnicas de iteración 155
2.6 Ordenación 158
3. Diccionarios 159
3.1 Presentación de los diccionarios 159
3.2 Recorrer un diccionario 160

3.3 Ejemplo 160

Capítulo 2-4
Las clases

1. Sintaxis 163
2. Noción de instancia en curso 164
3. Operadores 167
4. Herencia 168
   4.1 Especialización 169
   4.2 Programación por composición 170

Parte 3: Los fundamentos del lengaje

Capítulo 3-1
Algoritmos básicos

1. Delimitadores 173
   1.1 Instrucción 173
   1.2 Una línea de código = una instrucción 173
   1.3 Comentario 174
   1.4 Una instrucción en varias líneas 174
   1.5 Palabras clave 174
   1.6 Palabras reservadas 175
   1.7 Indentación 176
   1.8 Símbolos 177
   1.9 Operadores 180
      1.9.1 Operador de expresión de asignación := 183
   1.10 Uso del carácter de subrayado 184
   1.11 PEP-8 185
   1.12 PEP-7 185
   1.13 PEP-257 185
2. Instrucciones 186
   2.1 Definiciones 186
      2.1.1 Variable 186
      2.1.2 Función 188
      2.1.3 Funciones lambda 189
      2.1.4 Clase 189
      2.1.5 Instrucción vacía 190

2.1.6 Borrado . . . 191
2.1.7 Devolver el resultado de la función . . . 191
2.2 Instrucciones condicionales . . . 192
2.2.1 Definición . . . 192
2.2.2 Condición . . . 193
2.2.3 Instrucción if . . . 193
2.2.4 Instrucción elif . . . 193
2.2.5 Instrucción else . . . 194
2.3 Instrucción de correspondencia . . . 195
2.4 Uso de una expresión de asignación . . . 198
2.4.1 Instrucción switch . . . 199
2.4.2 Interrupciones . . . 199
2.4.3 Profundizando en las condiciones . . . 199
2.4.4 Rendimiento . . . 200
2.5 Iteraciones . . . 201
2.5.1 Instrucción for . . . 201
2.5.2 Instrucción while . . . 202
2.5.3 ¿Cuál es la diferencia entre for y while? . . . 202
2.5.4 Instrucción break . . . 203
2.5.5 Instrucción return . . . 204
2.5.6 Instrucción continue . . . 205
2.5.7 Instrucción else . . . 205
2.5.8 Generadores . . . 206
2.6 Construcciones funcionales . . . 209
2.6.1 Construcción condicional . . . 209
2.6.2 Generadores . . . 209
2.6.3 Recorrido de listas . . . 209
2.6.4 Recorrido de conjuntos . . . 209
2.6.5 Recorrido de diccionarios . . . 210
2.7 Recorrido y expresión de asignación . . . 210
2.8 Gestión de excepciones . . . 210
2.8.1 Breve presentación de las excepciones . . . 210
2.8.2 Elevar una excepción . . . 211
2.8.3 ¿Por qué elevar una excepción? . . . 211
2.8.4 Aserciones . . . 212
2.8.5 Capturar una excepción . . . 213
2.8.6 Manejar una excepción . . . 214
2.8.7 Gestionar la salida del bloque de captura . . . 216
2.8.8 Gestionar que no se produzcan excepciones . . . 216

2.8.9 Gestor de contexto . . . 218
2.8.10 Programación asíncrona . . . 219
2.9 Otros . . . 220
2.9.1 Gestionar imports . . . 220
2.9.2 Compartir espacios de nombres . . . 221
2.9.3 Funciones print, help, eval y exec . . . 223

Capítulo 3-2
Declaraciones

1. Variable . . . 225
1.1 ¿Qué es una variable? . . . 225
1.1.1 Contenido . . . 225
1.1.2 Continente . . . 225
1.1.3 Formas de modificar una variable . . . 227
1.2 Tipado dinámico . . . 230
1.2.1 Asignación: recordatorio . . . 230
1.2.2 Primitiva type y naturaleza del tipo . . . 230
1.2.3 Características del tipado Python . . . 231
1.3 Visibilidad . . . 234
1.3.1 Espacio global . . . 234
1.3.2 Noción de bloque . . . 234
2. Función . . . 238
2.1 Declaración . . . 238
2.2 Parámetros . . . 239
2.2.1 Firma de una función . . . 239
2.2.2 Noción de argumento o de parámetro . . . 240
2.2.3 Valor por defecto . . . 240
2.2.4 Valor por defecto mutable . . . 242
2.2.5 Parámetros nombrados . . . 243
2.2.6 Declaración de parámetros extensibles . . . 244
2.2.7 Paso de parámetros con asterisco . . . 245
2.2.8 Firma universal . . . 246
2.2.9 Obligar a un parámetro a ser nombrado (keyword-only) . . . 246
2.3 Forzar un argumento a ser posicional (positional-only) . . . 248
2.3.1 Anotaciones/tipo hint/tipado estático . . . 249
3. Clase . . . 252
3.1 Declaración . . . 252
3.1.1 Firma . . . 252

3.1.2 Atributo 253
3.1.3 Método 253
3.1.4 Bloque local 254
3.2 Instanciación 255
3.2.1 Sintaxis 255
3.2.2 Relación entre la instancia y la clase 255
4. Módulo 256
4.1 ¿Para qué sirve un módulo? 256
4.2 Declaración 256
4.3 Instrucciones específicas 256
4.4 ¿Cómo conocer el contenido de un módulo? 257
4.5 Compilación de los módulos 258

Capítulo 3-3

Modelo de objetos

1. Todo es un objeto 261
1.1 Principios 261
1.1.1 Qué sentido dar a «objeto» 261
1.1.2 Adaptación de la teoría de objetos en Python 262
1.1.3 Generalidades 263
1.2 Clases 264
1.2.1 Introducción 264
1.2.2 Declaración imperativa de una clase 265
1.2.3 Instancia 265
1.2.4 Objeto en curso 267
1.2.5 Declaración por prototipo de una clase 267
1.2.6 Tuplas con nombre 269
1.3 Métodos 270
1.3.1 Declaración 270
1.3.2 Invocar al método 272
1.3.3 Métodos y atributos especiales 274
1.3.4 Constructor e inicializador 277
1.3.5 Gestión automática de atributos 278
1.3.6 Interés del paradigma orientado a objetos 279
1.3.7 Relación entre objetos 279
1.4 Herencia 280
1.4.1 Polimorfismo por subtipado 280
1.4.2 Sobrecarga de métodos 281

1.4.3 Sobrecarga de operadores . . . 283
1.4.4 Polimorfismo paramétrico . . . 284
1.4.5 Herencia múltiple . . . 285
2. Otras herramientas de la programación orientada a objetos . . . 287
2.1 Principios . . . 287
2.2 Interfaces . . . 288
2.3 Atributos . . . 290
2.4 Propiedades . . . 293
2.5 Ubicaciones . . . 295
2.6 Metaclases . . . 296
2.7 Clases abstractas . . . 299
2.8 Zope Component Architecture . . . 301
2.8.1 Presentación . . . 301
2.8.2 Instalación . . . 302
2.8.3 Definir una interfaz y un componente . . . 302
2.8.4 Otras funcionalidades . . . 303
2.8.5 Ventajas de la ZCA . . . 303
3. Funciones principales y primitivas asociadas . . . 304
3.1 Personalización . . . 304
3.1.1 Clases . . . 304
3.1.2 Instancias . . . 306
3.1.3 Comparación . . . 307
3.1.4 Evaluación booleana . . . 307
3.1.5 Relaciones de herencia o de clase a instancia . . . 307
3.2 Clases particulares . . . 308
3.2.1 Iterador . . . 308
3.2.2 Contenedores . . . 310
3.2.3 Instancias similares a funciones . . . 311
3.2.4 Recursos que hay que proteger . . . 312
3.2.5 Tipos . . . 312
3.2.6 Clases de datos . . . 313

Capítulo 3-4

Números, booleanos y algoritmos aplicados

1. Números . . . 315
1.1 Tipos . . . 315
1.1.1 Enteros . . . 315
1.1.2 Reales . . . 316

1.1.3 Cosas en común entre números enteros y reales 317
1.1.4 Métodos dedicados a los números enteros 318
1.1.5 Métodos dedicados a los números reales 319
1.1.6 Complejos 319
1.2 La consola Python, la calculadora por excelencia 320
1.2.1 Operadores matemáticos binarios 320
1.2.2 Operadores binarios particulares 321
1.2.3 Operadores matemáticos unarios 322
1.2.4 Redondeo 323
1.2.5 Operadores de comparación 325
1.2.6 Operaciones matemáticas n-arias 326
1.2.7 Funciones matemáticas usuales 328
1.3 Representaciones de un número 333
1.3.1 Representación decimal 333
1.3.2 Representación por un exponente 333
1.3.3 Representación por una fracción 333
1.3.4 Representación hexadecimal 334
1.3.5 Representación octal 335
1.3.6 Representación binaria 336
1.3.7 Operaciones binarias 336
1.3.8 Longitud de la representación en memoria de un entero 338
1.4 Conversiones 339
1.4.1 Conversión entre enteros y reales 339
1.4.2 Conversión entre reales y complejos 340
1.4.3 Conversión en un booleano 340
1.5 Trabajar con variables 341
1.5.1 Un número es inmutable 341
1.5.2 Modificar el valor de una variable 342
1.5.3 Operadores incrementales 342
1.6 Estadísticas 343
2. Booleanos 344
2.1 El tipo booleano 344
2.1.1 Clase bool 344
2.1.2 Los dos objetos True y False 345
2.1.3 Diferencia entre el operador de igualdad y de identidad 345
2.2 Evaluación booleana 346
2.2.1 Método genérico 346
2.2.2 Objetos clásicos 346

Capítulo 3-5
Secuencias y algoritmos aplicados

1. Presentación de los distintos tipos de secuencias. 347
   1.1 Generalidades 347
   1.2 Las listas. 348
   1.3 Las n-tuplas 349
   1.4 Conversión entre listas y n-tuplas 351
   1.5 Cosas en común entre una lista y una n-tupla 351
2. Noción de iterador 352
3. Uso de índices y tramos. 355
   3.1 Definición de índice de un objeto y sus ocurrencias. 355
   3.2 Utilizar el índice para recorrer la secuencia 356
   3.3 Encontrar las ocurrencias de un objeto y sus índices 357
   3.4 Tamaño de una lista, contar ocurrencias. 358
   3.5 Utilizar el índice para modificar o eliminar. 359
   3.6 Iteración simple. 361
   3.7 Presentación de la noción de tramos (slices) 364
   3.8 Caso particular de la rama 2.x de Python 373
   3.9 Uso básico de tramos 374
   3.10 Uso avanzado de tramos. 375
4. Uso de operadores 377
   4.1 Operador + ... 377
   4.2 Operador * 378
   4.3 Operador += 381
   4.4 Operador *= 382
   4.5 Operador in 383
   4.6 Operadores de comparación 384
5. Métodos de modificación 385
   5.1 Agregar elementos a una lista y a una n-tupla 385
   5.2 Eliminar un objeto de una lista y de una n-tupla 387
   5.3 Soluciones alternativas para la modificación de n-tuplas. 391
   5.4 Invertir una lista o una tupla 392
   5.5 Ordenar una lista 393
6. Uso avanzado de listas 396
   6.1 Operaciones de conjunto 396
   6.2 Pivotar una secuencia 397

6.3 Iterar correctamente 398
6.4 Programación funcional 399
6.5 Recorrido de listas 401
6.6 Iteraciones avanzadas 403
6.7 Combinatoria 407
7. Adaptar las listas a necesidades específicas 410
7.1 Lista de enteros 410
7.2 Presentación del tipo array 411
7.3 Utilizar una lista como una pila 413
7.4 Utilizar una lista como una fila de espera 414
7.5 Contenedor con mejor rendimiento 414
7.6 Utilizar las listas para representar matrices 415
7.7 Lista sin duplicados 416
8. Otros tipos de datos 419

Capítulo 3-6

Conjuntos y algoritmos aplicados

1. Presentación 423
1.1 Definición de un conjunto 423
1.2 Diferencias entre set y frozenset 424
1.3 Uso para eliminar valores duplicados de las listas 425
1.4 Agregar una relación de orden 425
2. Operaciones sobre conjuntos 426
2.1 Operadores para un conjunto a partir de otros dos 426
2.2 Operadores para modificar un conjunto a partir de otro 427
2.3 Métodos equivalentes a la creación o modificación de conjuntos . . . 428
2.4 Métodos de comparación de conjuntos 428
2.5 Ejemplos de uso poco clásicos 430
3. Métodos de modificación de un conjunto 433
3.1 Agregar un elemento 433
3.2 Eliminar un elemento 434
3.3 Vaciar un conjunto 434
3.4 Duplicar un elemento 434
3.5 Sacar un valor de un conjunto 435
3.6 Utilizar un conjunto como un almacén de objetos 436
3.7 Algorítmica avanzada: resolución del problema de las n reinas 439

Capítulo 3-7
Cadenas de caracteres y algoritmos aplicados

1. Presentación 443
   1.1 Definición 443
   1.2 Vocabulario 444
   1.3 Especificidades de la rama 2.x 445
   1.4 Cambios aportados por la rama 3.x 446
   1.5 Cadena de caracteres como secuencia de caracteres 448
   1.6 Caracteres 450
   1.7 Operadores de comparación 452
2. Dar formato a cadenas de caracteres 454
   2.1 Operador módulo 454
   2.2 Métodos para dar formato al conjunto de la cadena 459
   2.3 Nuevo método para dar formato a variables en una cadena 462
   2.4 Literales formateados 465
3. Operaciones de conjunto 466
   3.1 Secuenciación de cadenas 466
   3.2 Operaciones sobre mayúsculas y minúsculas 468
   3.3 Búsqueda en una cadena de caracteres 469
   3.4 Información sobre los caracteres 470
4. Problemáticas relativas a la codificación 472
   4.1 Codificación por defecto 472
   4.2 Codificación del sistema 472
   4.3 Unicode, referencia absoluta 472
   4.4 Otras codificaciones 473
   4.5 Puntos entre el Unicode y el resto del mundo 474
   4.6 Volver a Unicode 475
5. Manipulaciones de bajo nivel avanzadas 476
   5.1 Operaciones para contar 476
   5.2 Una cadena de caracteres vista como una lista 477
   5.3 Una cadena de caracteres vista como un conjunto de caracteres 478
6. Representación en memoria 478
   6.1 Presentación del tipo bytes 478
   6.2 Vínculo con las cadenas de caracteres 479
   6.3 Presentación del tipo bytearray 480
   6.4 Gestión de un juego de caracteres 482

Capítulo 3-8
Diccionarios y algoritmos aplicados

1. Presentación 489
   - 1.1 Definición 489
   - 1.2 Evolución y diferencias entre las ramas 2.x y 3.x 490
   - 1.3 Vistas de diccionarios 491
   - 1.4 Instanciación 493
   - 1.5 Recorrer un diccionario 494
2. Manipular un diccionario 494
   - 2.1 Recuperar un valor de un diccionario 494
   - 2.2 Modificar los valores de un diccionario 495
   - 2.3 Eliminar una entrada de un diccionario 496
   - 2.4 Duplicar un diccionario 497
   - 2.5 Utilizar un diccionario como un agregador de datos 497
   - 2.6 Métodos de iteración 498
3. Uso avanzado de diccionarios 499
   - 3.1 Agregar una relación de orden 499
   - 3.2 Algorítmicas clásicas 500
   - 3.3 Adaptar los diccionarios a necesidades específicas 502
   - 3.4 Representación universal de datos 504

Capítulo 3-9
Datos temporales y algoritmos aplicados

1. Gestionar una fecha del calendario 507
   - 1.1 Noción de fecha del calendario 507
   - 1.2 Trabajar con una fecha 508
   - 1.3 Consideraciones astronómicas 509
   - 1.4 Consideraciones históricas 509
   - 1.5 Consideraciones técnicas 509
   - 1.6 Representación textual 510
2. Gestionar un horario o un momento de la jornada 511
   - 2.1 Noción de instante 511
   - 2.2 Noción de huso horario 513
   - 2.3 Representación textual 513

3. Gestionar un instante absoluto. 514
3.1 Noción de instante absoluto 514
3.2 Relación con las nociones anteriores 514
3.3 Representación textual 516
3.4 Gestión de los husos horarios 517
3.5 Crear una fecha a partir de una representación textual 517
4. Gestionar una diferencia entre dos fechas o instantes 517
4.1 Noción de diferencia y de resolución 517
4.2 Consideraciones técnicas 519
4.3 Uso con fechas del calendario. 520
4.4 Uso con horarios 520
4.5 Uso con fechas absolutas 520
4.6 El segundo como unidad básica 520
4.7 Precisión al nanosegundo 521
5. Especificidades de los husos horarios 521
6. Problemáticas de bajo nivel 522
6.1 Timestamp y struct_time. 522
6.2 Medidas de rendimiento 523
7. Uso del calendario 526
7.1 Presentación del módulo calendar 526
7.2 Funciones esenciales del calendario 530

Parte 4: Características

Capítulo 4-1
Manipulación de datos

1. Manipular los archivos 533
1.1 Abrir un archivo 533
1.2 Leer un archivo 534
1.3 Escribir un archivo 535
1.4 Comparar dos archivos 536
2. Herramienta de copia de seguridad. 538
3. Leer un archivo de configuración 538
4. Formato de exportación/importación 539
4.1 CSV 539
4.1.1 Explotar un archivo CSV 540
4.1.2 Generación de un archivo CSV 544

4.2 JSON 546
4.3 Base64 549
4.4 Pickle 549
5. Comprimir y descomprimir un archivo 552
5.1 Tarfile 552
5.2 Gzip 554
5.3 BZ2 554
5.4 Zipfile 555
5.5 Interfaz de alto nivel. 557
6. Herramientas de manipulación de datos 558
6.1 Generar números aleatorios 558
6.2 Expresiones regulares 559
7. Criptografía ligera 563
7.1 Número aleatorio seguro 563
7.2 Funciones de cifrado 563
7.3 Código de autenticación del mensaje 565
7.4 Huella de archivo 566
7.5 Esteganografía 567
7.6 Comunicación segura entre aplicaciones 570

Capítulo 4-2
Bases de datos

1. Introducción 573
2. Acceso a una base de datos relacional 573
2.1 Punto de entrada. 573
2.2 MySQL 574
2.3 PostgreSQL 579
2.4 SQLite 581
2.5 Oracle 582
3. Uso de un ORM. 582
3.1 ¿Qué es un ORM? 582
3.2 ORM propuestos por Python. 582
3.3 SQLAlchemy 583
3.3.1 Introspección en una tabla existente 583
3.3.2 Manipular datos en una tabla existente. 586
3.3.3 Describir una base de datos mediante el código 590

4. Otras bases de datos 591
4.1 CSV 591
4.2 NoSQL 597
4.3 Base de datos orientada a objetos: ZODB 597
4.4 Base de datos orientada al grafo: Neo4j 602
4.5 Base de datos de tipo clave-valor: Redis 603
4.6 Bases de datos orientadas a documentos: CouchDB y MongoDB . . . 605
4.7 Bases de datos nativas XML: BaseX, eXist 606
4.8 Cassandra 607
4.9 Bases de datos orientadas a columnas: HBase 607
4.10 Big Data: el ecosistema Hadoop 609
5. LDAP 611
5.1 Protocolo 611
5.2 Servidores 612
5.3 Terminología 612
5.4 Instalación 612
5.5 Abrir una conexión a un servidor 613
5.6 Realizar una búsqueda 614
5.7 Síncrono vs asíncrono 615
5.8 Conexiones seguras 616

Parte 5: Puesta en práctica

Capítulo 5-1
Crear un entorno de trabajo en 10 minutos

1. Descripción de la aplicación para construir 617
2. Contenedores 618
2.1 Portainer 618
2.2 Base de datos 619
3. Crear su contenedor Docker 621
4. Instalar sus bibliotecas Python 623

Capítulo 5-2
Crear una aplicación web en 30 minutos

1. Descripción de la aplicación que se va a construir. 625
2. Implementación. 626
   2.1 Aislar el entorno 626
   2.2 Creación del proyecto. 627
   2.3 Configuración 627
   2.4 Primeros ensayos. 629
3. Realizar la aplicación 630
   3.1 Modelos 630
   3.2 Plantillas 632
   3.3 Vistas 634
4. Para ir más allá 638

Capítulo 5-3
Crear una aplicación de consola en 10 minutos

1. Objetivo 639
2. Registrar el script 640
3. Creación de los datos. 640
4. Parser de argumentos 641

Capítulo 5-4
Crear una aplicación gráfica en 20 minutos

1. Objetivo 643
   1.1 Funcional. 643
   1.2 Técnico 643
2. Breve presentación de Gtk y algunos trucos 644
   2.1 Presentación 644
   2.2 Trucos 644
3. Iniciar el programa. 645
4. Interfaz gráfica con Glade. 648
5. Crear el componente gráfico. 650
6. Controlador 652

7. Otras librerías gráficas 653
7.1 TkInter 653
7.2 wxPython 653
7.3 PyQt 653
7.4 PySide 654
7.5 Otras 654

Capítulo 5-5
Crear un juego en 30 minutos con PyGame

1. Presentación de PyGame 655
2. Construcción de un juego Tetris 656
2.1 Presentación del juego 656
2.2 Presentación de la problemática 657
2.3 Creación de constantes 657

Anexos

1. Objetos mutables y no mutables 669
2. Tabla Unicode 672
2.1 Script 672
3. Bytes 672
3.1 Script 672
3.2 Resultado 673
4. Guía de migración a Python 3 676
5. Cómo depurar 677
6. Cómo probar el rendimiento 679

Índice 681

Contenido

Puede solicitar los archivos complementarios de este libro escribiendo a **comercial@ediciones-eni.com**.

Prólogo

1. Introducción 17
2. ¿A quién va dirigido este libro? 17
3. Objetivos y enfoque 18
4. Lo que no es Business Intelligence con Python. 18
5. Estructura y progresión 19
6. Lo que aprenderá. 19
7. En conclusión 20

Capítulo 1
Python como principal herramienta de BI

1. ¿Qué es la Business Intelligence?. 21
   - 1.1 Definición y desarrollo 21
   - 1.2 Componentes clave de la Business Intelligence 22
   - 1.3 Beneficios y evolución de la Business Intelligence 23
   - 1.4 Enfoque programático de la BI. 23
2. ¿Por qué Python? 24
   - 2.1 Preámbulo 24
   - 2.2 Tendencias 25
3. Python para Business Intelligence 28
4. ¿Cuáles son las ventajas de utilizar Python para Business Intelligence?. 30
5. Configuración del entorno 32
   - 5.1 Instalación de Python. 32
     - 5.1.1 ¿Qué versión de Python elegir? 32
     - 5.1.2 Instalación en Ubuntu 34

5.1.3 Instalación en Windows 11 34
5.1.4 Instalación en Windows 11 sin derechos de administrador 35
5.2 Elección de un IDE 35
5.2.1 Exploración de datos con Jupyter Notebooks 36
5.2.2 Entornos de desarrollo integrados (IDE) para producción 36
6. Algunas buenas prácticas antes de empezar 39
6.1 Control de versiones 40
6.1.1 Cree su repositorio 40
6.1.2 Clonación del repositorio 43
6.1.3 Añadir archivos y commit 44
6.1.4 Creación de ramas 44
6.1.5 Pull requests 45
6.2 Estructure su proyecto 47
6.3 Escriba código limpio y fácil de mantener 51
6.3.1 Buenas prácticas de formato y sintaxis 51
6.3.2 Documentación automática 57
6.3.3 Entornos virtuales 59
6.3.4 Logging con Python 61
6.3.5 Pruebe su código 62
6.3.6 Depuración 65
6.3.7 Integración continua con GitHub Actions 66

Capítulo 2
Extraer sus datos de cualquier fuente

1. Presentación de la biblioteca pandas 71
1.1 ¿Qué es un DataFrame? 71
1.2 Estructura de un DataFrame 72
1.3 Principales atributos de un DataFrame 72
1.3.1 Atributo shape 72
1.3.2 Atributo columns 73

1.3.3 Atributo index 73
1.3.4 Atributo dtypes. 73
1.3.5 Métodos head() y tail() 73
2. Archivos planos u otros formatos estructurados 74
2.1 CSV, TXT y TSV 74
2.2 XLS, XLSX. 78
2.3 JSON 79
2.4 XML. 80
2.5 PDF 81
2.6 Caso especial de datos tabulares almacenados como imágenes. 85
3. Bases de datos relacionales 86
3.1 ¿Qué es una base de datos relacional? 86
3.2 ¿Cuáles son las bases de datos más comunes? 87
3.3 Bases de datos PostgreSQL. 88
3.4 Bases de datos MySQL 92
3.5 Bases de datos SQLite. 93
4. Bases de datos NoSQL 94
4.1 ¿Qué es una base de datos NoSQL? 94
4.2 Conexión a una base de datos MongoDB 95
5. Almacenes de datos 98
5.1 ¿Qué es un almacén de datos? 98
5.2 Elección de soluciones NoSQL. 99
5.3 Conexión a bases de datos con Python 100
5.3.1 BigQuery 100
5.3.2 Snowflake 101
5.3.3 Buenas prácticas 103
6. Servidores FTP/SFTP 103
6.1 ¿Qué es FTP? 103
6.2 ¿Qué es SSH? 104
6.3 Conexión con Python. 104

7. Interfaz de programación de aplicaciones (API) 106
7.1 ¿Por qué utilizar una interfaz de programación de aplicaciones? 106
7.2 Algunas definiciones y un poco de teoría 107
7.2.1 Tipos de peticiones 108
7.2.2 Puntos finales 108
7.3 Códigos de respuesta 109
7.4 El resultado de la petición 111
7.4.1 Atributo text 111
7.4.2 Atributo raw 111
7.4.3 Atributo content 112
7.4.4 Método json 112
7.5 Las bibliotecas requests y json 112
7.5.1 Presentación de la biblioteca requests 112
7.5.2 Presentación de la biblioteca json 113
7.5.3 Instalación 113
7.5.4 Utilización de la API sin autenticación 114
7.5.5 Llamadas concurrentes a la API con multithreading . . . 115
7.5.6 Peticiones API con autenticación 117
7.6 Buenas prácticas 120
7.7 Ilustración práctica 121
7.8 Para practicar 125
8. Web scraping 125
8.1 HTML/CSS básico 126
8.1.1 HTML (HyperText Markup Language) 126
8.1.2 CSS (Cascading Style Sheets) 126
8.2 Utilización de requests y BeautifulSoup 126
8.3 Metodología 127
8.4 Web Scraping con selenium 131

Capítulo 3

Preparar sus datos para sacarles todo su potencial

1. La calidad de los datos: un recordatorio 135
   - 1.1 ¿Qué es la calidad de los datos? 135
   - 1.2 ¿Por qué es importante la DQD? 137
   - 1.3 Los principales criterios de la DQD 139
     - 1.3.1 Precisión (accuracy) 139
     - 1.3.2 Integridad (completeness) 140
     - 1.3.3 Coherencia (consistency) 140
     - 1.3.4 Vigencia (timeliness) 141
     - 1.3.5 Validez (validity) 141
     - 1.3.6 Singularidad (uniqueness) 142
2. Depuración de datos 142
   - 2.1 Primeros pasos con la biblioteca pandas 142
   - 2.2 Presentación de nuestro conjunto de datos 143
   - 2.3 Manipulación básica de un conjunto de datos 144
     - 2.3.1 Método head() 145
     - 2.3.2 Atributo shape 147
     - 2.3.3 Atributo columns 147
   - 2.4 Creación de subconjuntos 148
     - 2.4.1 Selección de columnas 149
     - 2.4.2 Selecciones mediante los métodos loc e iloc 150
     - 2.4.3 Selecciones condicionales 151
   - 2.5 Limpieza del conjunto de datos 154
     - 2.5.1 Gestión de duplicados 154
     - 2.5.2 Valores perdidos 157
     - 2.5.3 Modificación de elementos 166
   - 2.6 Procesamientos avanzados 174
     - 2.6.1 Valores atípicos 174
     - 2.6.2 Valores aproximados 180
     - 2.6.3 Series temporales 185

3. Los cuatro pilares del manejo de datos con pandas 187
3.1 Filtrado avanzado de un DataFrame con operadores binarios 188
3.2 Unir DataFrames con concat y merge. 191
3.3 Fusión de DataFrames con el método merge 192
3.4 Clasificación y ordenación de los valores de un DataFrame: métodos sort_values y sort_index 197
3.5 Agrupación de elementos en un DataFrame: métodos groupby, agg y crosstab 198

Capítulo 4

Analizar y comprender sus datos

1. Introducción 203
1.1 Tipos de variables 204
1.2 Nociones de población y muestra 205
1.3 Leyes estadísticas de probabilidad 207
2. Estadística descriptiva 213
2.1 Análisis univariante 214
2.1.1 Indicadores de posición 214
2.1.2 Indicadores de dispersión 216
2.2 Análisis bivariante 223
2.2.1 Correlación entre variables cuantitativas 223
2.2.2 Asociación entre variables cualitativas 230
2.2.3 Relación entre las variables cualitativas y cuantitativas 235
3. Inferencia estadística 237
3.1 Noción de intervalo de confianza 237
3.2 Principios de las pruebas de hipótesis 239
3.3 Pruebas paramétricas 242
3.3.1 Prueba de normalidad. 242
3.3.2 Prueba t de Student 244
3.3.3 ANOVA unidireccional 245

3.4 Pruebas no paramétricas. 247
3.4.1 Prueba de Mann-Whitney 247
3.4.2 Prueba de Kruskal-Wallis 248
4. Técnicas avanzadas de análisis estadístico 250
4.1 Regresión lineal simple y múltiple. 250
4.1.1 Regresión lineal simple. 250
4.1.2 Interpretación de los coeficientes 251
4.2 Regresión lineal múltiple 253
4.2.1 Estimación de parámetros 254
4.2.2 Interpretación de los coeficientes 254
4.3 Evaluación en profundidad del modelo. 255
4.4 Límites y consideraciones. 257
4.5 Predictive Power Score 258
5. Caso de estudio: A/B testing en marketing 261
5.1 Presentación del contexto y de los objetivos 261
5.2 Diseño del experimento A/B 261
5.3 Recogida y preparación de datos 262
5.4 Análisis estadístico de los resultados. 263
5.4.1 Estadísticas descriptivas. 263
5.4.2 Pruebas de hipótesis para comparaciones de grupos . . . 264
5.4.3 Cálculo e interpretación del tamaño del efecto 266
5.5 Visualización de los resultados 267
5.6 Interpretación y toma de decisiones basadas en datos. 268
5.7 Limitaciones y consideraciones para futuras pruebas 269

Capítulo 5
Crear un Data Warehouse

1. Introducción 271
   1.1 Definiciones preliminares. 272
      1.1.1 OLTP (Online Transaction Processing) 273
      1.1.2 OLAP (Online Analytical Processing) 273
   1.2 ¿Qué es un Data Warehouse? 273
2. Las características y ventajas de un Data Warehouse 276
   2.1 Las principales características 276
   2.2 Ventajas de un Data Warehouse 277
3. Los componentes de una arquitectura analítica 278
4. Los diferentes tipos de arquitectura de un proyecto analítico. 281
   4.1 Arquitectura Single Tier. 281
   4.2 Arquitectura Two Tier. 284
   4.3 Arquitectura Three Tier. 286
   4.4 Conclusión 288
5. Normalización/desnormalización 289
6. Diferentes métodos de diseño de DWH 290
   6.1 Metodología de diseño Inmon. 290
   6.2 Metodología de diseño Kimball 292
   6.3 Metodología de diseño de OBT simple 294
   6.4 Metodología de diseño Data Vault 296
   6.5 El ciclo de vida de un proyecto de Data Warehouse. 298
7. Los distintos tipos de tablas en un Data Warehouse 301
   7.1 Las tablas de hechos (fact tables). 301
   7.2 Las tablas de dimensiones (dimension tables) 302
8. Esquemas. 303
   8.1 Star Schema (esquema en estrella) 303
   8.2 Snowflake Schema (esquema de copo de nieve). 304

9. Proyecto Data Warehouse 306
9.1 Requisitos previos 307
9.1.1 Ubuntu 307
9.1.2 Windows 308
9.1.3 Windows 309
9.2 Script en Python 310
9.3 Creación del diagrama ERD 316
9.4 Solicitudes 318

Capítulo 6

Automatizar su pipeline

1. Introducción a los ETL y la automatización de pipelines de datos 321
1.1 Definición e importancia de los ETL 321
1.2 Ventajas de la automatización de los pipelines de datos 322
1.3 Visión general de las herramientas de automatización: Airflow y Luigi 323
2. Apache Airflow: una potente herramienta para orquestar workflows 324
2.1 Presentación de Airflow 324
2.2 Conceptos clave: DAG, Tasks, Operators 325
2.3 Instalación y configuración básica (Linux) 326
2.4 Cree un pipeline simple con Airflow 327
2.5 Ventajas y casos de uso 329
3. Luigi: una alternativa ligera para la automatización de tareas 330
3.1 Introducción a Luigi 330
3.2 Conceptos fundamentales: Tasks, Targets, Parameters 330
3.3 Instalación y configuración 331
3.4 Creación de un pipeline básico con Luigi 332
3.5 Principales ventajas y casos de uso 336

4. Comparación entre Airflow y Luigi. 337
4.1 Arquitectura y diseño. 338
4.2 Definición y gestión de workflows 339
4.3 Planificación y ejecución 340
4.4 Integración y escalabilidad. 341
4.5 Escalabilidad y rendimiento. 342
4.6 Comunidad y soporte. 343
5. Buenas prácticas para diseñar pipelines de datos con Python 344
5.1 Modularidad y reutilización del código. 344
5.2 Gestión de errores y recuperación en caso de fallo 345
5.3 Logging y monitoring. 347
5.4 Control de versiones de pipelines 349
5.5 Pruebas y validación de datos 350
5.6 Documentación de códigos y procesos 354
6. Caso práctico: creación de un pipeline ETL completo 357
6.1 Definición de requisitos y flujo de datos. 358
6.1.1 Estructura del código 358
6.1.2 Funciones ETL (etl_functions.py) 358
6.1.3 DAG Airflow. 359
6.1.4 Ventajas de este enfoque 359
6.2 Implementación con Airflow. 360
6.3 Implementación con Luigi 365
6.4 Comparación de enfoques y debate. 368
7. Conclusión y perspectivas 370
7.1 Resumen de los puntos claves 370
7.2 Tendencias futuras en la automatización de pipelines de datos 371

Capítulo 7
Visualizar sus datos

1. Introducción a la visualización de datos 373
2. ¿Por qué visualizar los datos?. 379
   2.1 Más allá de las cifras: la importancia de la visualización 379
   2.2 Python: una herramienta sin límites. 380
3. Visión general de las diferentes bibliotecas gráficas con Python . . . 380
   3.1 Un ecosistema rico y diverso 380
   3.2 El trío ganador Matplotlib Seaborn y Plotly 382
   3.3 matplotlib 383
      3.3.1 Instalación. 383
      3.3.2 Presentación del conjunto de datos 383
      3.3.3 Gráficos de línea 385
      3.3.4 Diagrama de barras. 389
      3.3.5 Nube de puntos 393
   3.4 Seaborn 396
      3.4.1 Instalación. 397
      3.4.2 Conjunto de datos 397
      3.4.3 Histogramas 398
      3.4.4 Diagrama de cajas y bigotes. 403
      3.4.5 Nube de puntos 404
      3.4.6 Matriz de correlación 408
      3.4.7 Pairplot 410
   3.5 Plotly 412
      3.5.1 Introducción 412
      3.5.2 Instalación. 413
      3.5.3 Gráfico en cascada 413
      3.5.4 Indicadores 415
      3.5.5 Manómetros 416
      3.5.6 Embudo 417
      3.5.7 Mapas 418
      3.5.8 Treemaps. 419

3.6 Gráficos a medida 420
3.6.1 Gráficos combinados 421
3.6.2 Gráficos de cinta 425
3.6.3 Area chart 428
4. Buenas prácticas de diseño 430
5. Caso práctico 439

Capítulo 8
Paneles de control e informes

1. De los datos a las decisiones: sacar el máximo partido del panel de control 441
2. Storytelling: el arte de hacer hablar a los datos 443
3. Domine los paneles de control BI con Python 447
3.1 Streamlit 447
3.1.1 Instalación 448
3.1.2 Los componentes 448
3.1.3 Optimizaciones 457
3.1.4 Nuestro primer panel de control Streamlit 459
3.2 Taipy 468
3.2.1 Instalación 468
3.2.2 Los componentes básicos 468
3.2.3 Primer panel de control Taipy 472
3.2.4 Creación de interfaces con taipy.gui.builder 481
3.3 Dash 486
3.3.1 Instalación 486
3.3.2 Los componentes básicos 486
3.3.3 Elementos de la estructura de página 491
3.3.4 Control e interactividad 497
3.3.5 Creación de paneles de control con Dash 503

4. Cree impactantes informes de BI (y rápido) 512
4.1 Jupyter Notebooks 513
4.2 Quarto 515
5. Difunda y comparta sus análisis 518
5.1 Desarrollo local 518
5.2 Preparativos para la implementación 519
5.3 Opciones de despliegue 519
5.3.1 Plataformas de despliegue específicas para cada framework 519
5.3.2 Plataformas cloud versátiles 520
5.3.3 Alojamiento estático con generación del lado del cliente 520
5.3.4 Servidores privados virtuales (VPS) 520
5.4 Seguridad y acceso 520
5.5 Mantenimiento y actualización 521

Capítulo 9
Ética, seguridad y RGPD

1. Introducción 523
1.1 Importancia de la ética la seguridad y el RGPD en la Business Intelligence 523
1.2 Desafíos actuales en el procesamiento de datos empresariales 524
2. Ética en la Business Intelligence 525
2.1 Principios éticos fundamentales en BI 525
2.1.1 Transparencia 525
2.1.2 Equidad 526
2.1.3 Responsabilidad 526
2.2 Sesgo en los datos y el análisis 526
2.2.1 Tipos comunes de sesgo 526
2.2.2 Consecuencias del sesgo en las decisiones empresariales 527

2.3 Toma de decisiones éticas basadas en datos 527
2.4 Gobernanza ética de datos 527
3. Seguridad de los datos en la Business Intelligence 528
3.1 La importancia de la seguridad de los datos en las empresas 528
3.2 Amenazas comunes a la seguridad de los datos 529
3.2.1 Ciberataques 529
3.2.2 Filtraciones internas de datos 529
3.2.3 El error humano 529
3.3 Mejores prácticas para la seguridad de los datos 530
3.3.1 Control de acceso y autenticación. 530
3.3.2 Cifrado de datos 530
3.3.3 Copias de seguridad y planes de recuperación en caso de siniestro. 530
3.4 Capacitación y concienciación de los empleados sobre la seguridad 531
4. RGPD y cumplimiento en Business Intelligence 531
4.1 Visión general del RGPD 532
4.2 Principios claves del RGPD aplicables a la BI 532
4.2.1 Consentimiento y base jurídica del procesamiento 532
4.2.2 Minimización de datos. 532
4.2.3 Limitación de la finalidad. 533
4.3 Derechos individuales en virtud del RGPD. 533
4.3.1 Derecho de acceso. 533
4.3.2 Derecho de supresión 533
4.3.3 Derecho a la portabilidad de los datos. 533
4.4 Cumplimiento del RGPD en proyectos de BI 534
4.4.1 Evaluación de impacto de la protección de datos (EIPD 534
4.4.2 Privacy by Design y Privacy by Default 534
4.5 Gestión de la violación de datos y notificación 534

5. Integración de la ética la seguridad y el RGPD en los procesos de BI 535
5.1 Creación de una cultura corporativa centrada en la ética y la protección de datos 535
5.2 Integración de consideraciones éticas y de confidencialidad en el ciclo de vida del proyecto de BI 536
5.3 Auditorías y evaluaciones periódicas 536
5.4 Colaboración entre equipos (BI, jurídico, seguridad, cumplimiento) 537
6. Futuros desafíos y oportunidades 538
6.1 Evolución en la normativa sobre protección de datos 538
6.2 Innovaciones tecnológicas y sus implicaciones éticas 539
6.3 Equilibrio entre innovación y protección de datos 539
7. Conclusión 540
8. Recursos complementarios 541
8.1 Directrices y marcos éticos 541
8.2 Herramientas y recursos para la seguridad de datos 542

Índice 545