From Basics to Bytecode

From Basics to Bytecode: A Guide to Computers and Programming

Set Lonnert
with AI Collaboration

Adobe and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Raspberry Pi is a trademark of Raspberry Pi Ltd.

All other trademarks are the property of their respective owners.

All images are in the public domain unless otherwise stated.

Cover illustration in: Journal for Manufactures and Household Management, Scheutz, Stockholm, 1825, third issue, between pp. 96–97. Uppsala University Library. https://urn.kb.se/resolve?urn=urn:nbn:se:alvin:portal:record-95006
An illustration of the Jacquard mechanism. A device that uses punched cards to control the lifting of individual warp threads on a loom, making it possible to automate the weaving of intricate patterns with precision.

This work is marked with CC0 1.0 Universal.
To view a copy of this license, visit:
https://creativecommons.org/publicdomain/zero/1.0/

Authors: Set Lonnert with AI Collaboration 2025
Publisher: BoD, Books on Demand, Stockholm, Sweden
Printer: Libri Plureos GmbH, Hamburg, Germany
ISBN: 978-91-XXXX-XXX-X

Contents

Introduction **i**

1 Foundations **1**

1.1 Prerequisites 1
1.2 Foundations of Programming 1
1.3 Simple data types 2
1.3.1 Integers in binary 2
1.3.2 Floating-point numbers 4
1.3.3 Characters and ASCII 4
1.3.4 Strings 5
1.3.5 Representations and types 5
1.3.6 Summary 6
1.4 Variables 7
1.4.1 Assignment 7
1.4.2 Mutable and immutable variables 9
1.4.3 Summary 12
1.5 Control structures 13
1.5.1 Conventional control structures 13
1.5.2 Control structures in computers 17
1.5.3 Summary 18
1.6 Functions 19
1.6.1 Calling functions 20
1.6.2 Summary 23
1.7 Practice 24

2 Understanding VMs **29**

2.1 Simple VMs 29
2.1.1 The stack 30
2.1.2 Interpreter technique 31
2.1.3 VM1 implementation 31

2.1.4 REGVM implementation . . . 33
2.1.5 Portability . . . 36
2.1.6 Summary . . . 38
2.2 Stack-based VM . . . 39
2.2.1 Comparisons . . . 41
2.2.2 Iterations . . . 42
2.2.3 Error handling . . . 43
2.2.4 Summary . . . 45
2.3 Memory and functions . . . 46
2.3.1 Frame pointer . . . 51
2.3.2 Local storage . . . 53
2.3.3 Memory management . . . 54
2.3.4 Frame stack . . . 55
2.3.5 Summary . . . 60
2.4 Practice . . . 62

3 Development Environment 67
3.1 Prerequisites . . . 67
3.2 Basic tools . . . 67
3.3 Debugging . . . 68
3.3.1 Process . . . 70
3.3.2 Tools . . . 71
3.3.3 Summary . . . 78
3.4 Optimisation . . . 78
3.4.1 Memory . . . 83
3.4.2 Time . . . 85
3.4.3 Confusion matrix . . . 87
3.4.4 Summary . . . 93
3.5 Tests and testing . . . 93
3.5.1 Automated testing and continuous integration . . . 97
3.5.2 Summary . . . 100
3.6 Practice . . . 101

4 Building and Experimenting 105
4.1 Prerequisites . . . 105
4.2 The computer as hardware . . . 105
4.2.1 Hardware . . . 108
4.2.2 The Pico . . . 110
4.2.3 Summary . . . 113
4.2.4 Practice . . . 114
4.3 Input/Output . . . 115

4.3.1 The Pico pins 116
4.3.2 Light switching circuit 117
4.3.3 Programmable I/O 118
4.3.4 State machines 121
4.3.5 Circuit for traffic lights 124
4.3.6 Pedestrian crossing 126
4.3.7 Temperature measurements 128
4.3.8 Temperature indicator 129
4.3.9 Summary 132
4.3.10 Practice 134
4.4 Secondary memory 138
4.4.1 Save and load temperatures 140
4.4.2 Summary 140
4.4.3 Practice 140
4.5 External communication 140
4.5.1 UART 140
4.5.2 Connect Two Picos 142
4.5.3 Client-Server 145
4.5.4 Wireless 148
4.6 Security 149
4.6.1 Encryption/Decryption 149
4.6.2 Example 150
4.6.3 Program 1: Encrypting Program 152
4.6.4 Program 2: Decrypting Program 153
4.6.5 Secure communication 155
4.6.6 Data integrity 156
4.6.7 Hashing for Security 157
4.6.8 Symmetric/Asymmetric Encryption 159
4.6.9 Summary 161
4.6.10 Practice 162
4.7 Direct memory access 163
4.7.1 Pico and DMA 164
4.7.2 Pico and PicoDisplay 165
4.7.3 Text 168
4.7.4 Pixel graphics 168
4.7.5 Vector graphics 168
4.7.6 Summary 169
4.7.7 Practice 169
4.8 Combining I/O 169
4.8.1 Simplified web server 169
4.8.2 Summary 169

4.8.3 Practice 169
4.9 Handling errors 169
4.9.1 Types 170
4.9.2 Approaches 170
4.9.3 Summary 173
4.10 Timing and timers 174
4.10.1 Timing and Timers in the Pico 175
4.10.2 Interrupt handling 177
4.10.3 Summary 177
4.10.4 Practice 177
4.11 Concurrency and multithreading 178
4.12 Power management 183
4.13 Practice 188

5 The Compiler Pipeline 189
5.1 Prerequisites 189
5.2 Compilers, Interpreters, Assembler, and VMs 189
5.3 Syntax 192
5.3.1 Tokenisation or Lexical Analysis 193
5.3.2 Parsing 194
5.3.3 Production Rules and Grammars 196
5.3.4 Bottom-up 198
5.3.5 Top-down 204
5.3.6 Building Abstract Syntax Trees (ASTs) 208
5.3.7 Summary 210
5.4 Semantic Analysis 211
5.4.1 Symbol Table 214
5.4.2 Type System 214
5.4.3 Use of Abstract Syntax Trees (ASTs) 214
5.4.4 Rules and Constraints 214
5.4.5 Semantic Analysis Techniques 215
5.5 Intermediate Representations (IR) 215
5.5.1 Three-Address Code (TAC) 220
5.5.2 Role of IR in Optimisation 221
5.5.3 Summary 224
5.6 Code Generation Techniques 224
5.7 Loader and Linker 225
5.8 Practice 230

6 Philosophy and Methodology 237
6.1 Philosophy and style 237

6.1.1 Programming as craft 238
6.1.2 Building process 239
6.1.3 Blending philosophies 242
6.1.4 Methodologies 245
6.1.5 References 247
6.1.6 A personal note 249
6.1.7 Risks 251
6.1.8 Summary 252
6.2 Problems and methods 253
6.2.1 Example: Low customer satisfaction 253
6.2.2 Example: Thermostat 254
6.2.3 Generalisations 255
6.2.4 Methodology in the craft philosophy 259
6.3 Reflections on the history and future 261
6.3.1 AI example: LLMs 263
6.3.2 Summary 266
6.4 Method examples 266
6.4.1 Mock-ups and prototypes 267
6.4.2 Code review 271
6.4.3 Null hypothesis in programming 279
6.4.4 Summary 283
6.5 Practice 284

7 Case Studies 285
7.1 JVM 287
7.2 PostScript 287
7.3 Language and Virtual Machine Projects 292
7.4 Systems Programming Projects 292
7.5 Tools and Frameworks 292
7.6 Simulators and Emulators 293
7.7 Data Science and AI Projects 293
7.8 Creative and Visual Projects 293

8 Advanced Topics 295
8.1 Abstract data structures 295
8.2 Object-orientation 295
8.3 Functional languages 295
8.3.1 Design patterns 295
8.4 Design patterns 305
8.4.1 Factory 307
8.4.2 Observer 309

8.4.3 Command 311
8.4.4 Strategy 314
8.4.5 Adapter 316
8.4.6 Composite 319
8.4.7 Decorator 320
8.5 Security 322
8.6 Garbage collection 322
8.7 Memory management 323
8.8 Graphics 323
8.9 Different VMs 323

Register **326**