Beneath and Beyond the Code

ANA MARQUES DA SILVA
CLP | Universidade de Coimbra
Bolseira da FCT

Nick Monfort *et al.*, *10 PRINT CHR$(205.5+RND(1)); : GOTO 10*. Cambridge, MA: The MIT Press, 2013, 309 pp.
ISBN 978-0-262-01846-3

Although software **is** deeply embedded in contemporary post-industrial societies, it remains opaque to most of its users. That is why emerging disciplines such as critical code, software and platform studies largely contribute to the construction of knowledge about the many ways in which computer code participates in culture, shedding light on the importance of literacy in the context of the digital instruments and processes which are increasingly present in human activities. As is clearly stated early on, "code is not merely functional", it has "significant social, political, and aesthetic dimensions" (3) which may be revealed with a hermeneutic approach, by a close reading of the code itself. That is the aim of this book: an attentive and deep analysis of a program coded in a single line – *10 PRINT CHR$(205.5+RND(1)); : GOTO 10* – that unfolds into three hundred pages, telling the story of this particular computer program, of the BASIC language on which it is written, and of the Commodore 64 computer on which it ran. At the same time, this analysis explores randomness and repetition in computing, as well as in the arts and sciences, while reflecting upon the ways in which mazes are imagined in culture. Through this approach, the work brings forth the physical and socio-cultural materialities of computational processes, the value of creative computing, and the epistemological relevance of code, radially establishing many relationships with a myriad of points usually perceived as distant from computation. From the history to the specificities of

computer processes, and from the memory rooted in aesthetic artifacts to the relationship between scientific endeavors and their historical background, this one-line program reveals itself as a text whose analysis sheds light on the materialities of culture and human imagination.

The set of instructions coded in this extremely concise program (10 PRINT CHR$(205.5+RND(1)); : GOTO 10) produces the image of a maze pattern, as represented in the book's cover: "when the program runs, the characters appear one at a time, left to right and then top to bottom, and the image scrolls up by two lines each time the screen is filled" (8). This is a non-utilitarian program, its output produces an image that unfolds indefinitely through the screen. But besides causing something to happen on a computer screen, this line of code is also related to many cultural objects, raising questions about the conditions under which it was written and performed. Just as code is dialogically adapted by many programmers, this book is collaboratively written by ten renowned authors (Nick Montfort, Patsy Baudoin, John Bell, Ian Bogost, Jeremy Douglass, Mark C. Marino, Michael Mateas, Casey Reas, Mark Sample, Noah Vawter) in the fields of literature, media studies, computer science and art. Despite all the different hands at work, *10PRINT* has an homogeneous style and type of approach: straightforward and clear (like code). After introducing the scope of the work, the authors describe, step by step, the functioning of 10PRINT, raising issues like the historical context of each command and relating, for example, the evolution of computers with typewriters and other media.

The pattern produced by 10PRINT is a never-ending maze. The authors argue that although an adult could "easily trivialize and dismiss it as simply a childish amusement", that posture would disregard "the cultural resonance and historical depth of the maze" as well as the "culture of creative, exploratory computing" (32). The first chapter addresses the meaning and the long history of mazes in western cultures, from the labyrinth of Knossos to the contemporary explorations of these structures in arts, sciences or computer games. A multi-perspectival approach highlights the scope of computer programming but, at the same time (and more importantly), acknowledges that the mastery of a maze reflects the path from confusion to solution, which is created by the explorer, and not by the maze's programmer. Thus, the use of any program (or any other technical tool) should be accompanied by the knowledge of its nature, premises and possibilities, so that users may be producers and not merely consumers. The two dynamics that give 10PRINT its visual impact are regularity and randomness. The second chapter is dedicated to regularity. Although intricate and complex, 10PRINT is a regular procedure developing in space and time, producing patterns in the process. Regularity is explored in human cultures as a key element to produce a sense of orientation and control over disorder. The authors consider many examples taken from art history, such as patterns and grids used in ancient tapestry, tiling or modern art movements inspired by constructivism.

Randomness is explored in the third chapter. Games have been played throughout all history, and uncertainty is a key element in them. Through the study of games, anthropologist Thomas Malaby came to distinguish four categories of indeterminacy: formal, or chance (understood through statistical analysis); social, or the impossibility of knowing another subject's perspective; performative, or the unreliability of another's actions; and cosmological, or the "skepticism" (123) about the rules of the game. In this context, the authors criticize the tendency in players, scholars, and programmers to privilege stochastic principles of chance, dismissing social and performative forms of indeterminacy. In the sciences, randomness is necessary for simulating anything with unknown variables. It was John von Neumann, while working on the hydrogen bomb, who first proposed the idea of using computers to generate random numbers, in 1946. In the case of 10PRINT, randomness is necessary to create the impression of a maze, producing diagonal lines that alternate unpredictably between left and right. As the authors state, "the RND command acts as the algorithmic heart of 10PRINT, its flip-flopping beat powering the construction of the maze" (120). But randomness is not fully random: computers are "deterministic devices", since "the next state of the machine is determined entirely by the current state" (130), and so their processes are in fact "pseudo-random", which means that with long enough sequences, and in time, an underlying 'order' may be unveiled.

In the arts, the authors find an area filled with examples of chance operations: while Marcel Duchamp stated that "pure chance interested me as a way of going against logical reality", John Cage used randomness to "remove individual bias from creation" (126). The work of computer scientist Christopher Strachey, who in 1952 programmed a computer to produce a series of love letters, is referred to as the first encounter between literature and computation, although Strachey dismissed the experiment as literary since the texts served "simply to illustrate his point that simple rules can generate diverse and unexpected results" (134). In the 1960s, the public image of the computer shifted from being "a machine that supported technocracies to a tool for self-empowerment and creativity" (135), favoring a transdisciplinary culture of intense experimentation with randomness. From the 1970's on, a negative stigma towards computation in the arts began to emerge, as the words of conceptual artist Manfred Mohr show: "people accused me of degrading art, because I was employing capitalistic instruments of war – 'computer' was a word *non grata!*" (139). But today this trend is considered to be fading away, as new generations of artists have started programming their works.

The way 10PRINT functions depends, in part, on the language in which it is written: BASIC, whose history is told in the fourth chapter. Ada Byron, the Countess of Lovelace (1815–1852), is generally considered to be the first computer programmer. Although computers didn't exist at her time, Lovelace created "an algorithm described in mathematical notation" that was

more intelligible than computer programs that came later. The next stage emerged with the creation of machine language, which is nothing more than a sequence of numbers that causes the computer to operate. Greater legibility was brought by the second generation of programming languages, called assembly languages, which resorted to the use of mnemonics. The authors consider that "during the first decade of electronic computing, programming was still crawling toward the ideal of Lovelace's 'program' that specified an algorithm in high-level, human-readable form" (161). Readability improved later, during the 1950's and 60's, with FORTRAN and COBOL, designed for mathematics and business, respectively. Finally, in 1964, BASIC (or the 'Beginner's All-purpose Symbolic Instruction Code') was developed at Dartmouth College by two professors "who freely shared a working version of the language, leading to its widespread adoption at the high school and college level" (158).

BASIC programs circulated in print. In the 1970's and 80's, computer magazines and books featured programs that a user could load on a computer, so these materials highlight how "BASIC facilitated exploration, play, modification, and learning" (186) and how computer culture emphasized sharing. Another way for users to share programs was through memorization: "like a software virus whose host is a human rather than a machine" (186), code was memorized and distributed throughout communities of users, in a cultural economy of exchange. This happened with short programs, often one-liners like 10PRINT, and here is a curious fact: a version of 10PRINT gave rise to the discussions that originated this book, when Nick Montfort (one of its authors) presented it in 2010. That version was "corrupt" mainly because it was memorized. So memory can produce errors but that gives way to adaptations, leaving a space for creativity. After the era of Commodore 64 and other computers featuring BASIC, the use of this language declined but its lineage continues: "the principles behind BASIC remain strong, though, and continue to make programming languages easier, more transparent, and more freely distributed" (193). Through their reading of BASIC's history, the authors synthesize the conflicting dynamics between open source and corporative property, and they underline the contribution of higher education institutions, corporations and hackers to the history of programming.

The fifth chapter situates 10PRINT in the cultural ground of the Commodore 64 computer (the platform on which 10PRINT ran), and analyzes variations on BASIC, one-line programs comparable to 10PRINT, as well as 10PRINT's porting to other languages and computers, revealing what is particular to BASIC and its first platform. The Commodore 64 has been considered the best-selling computer ever: given that home computing was costly, Commodore positioned itself as a name for an economical yet very powerful computer. Due to the diversity of computers, "the experience of home computing was in many ways stratified by platform" (216). These and

other social and economical aspects are considered in this chapter. Relevant to the way 10PRINT operates and looks is the eight-bit Commodore PET, which included BASIC, and therefore was ready to be programmed. Also unique to the Commodore computer is its extended graphical character set, PETSCII. These extensions define 10PRINT's output, each one producing a different visual aspect. Furthermore, they had a SID chip, which "made the Commodore a formidable music maker and game machine" (230). But, interestingly enough, this chip could also be used for number generation, as is the case of a similar program discussed later in the book. Finally, the KERNAL (a misspelling of 'kernel') is the Commodore 64's operating system, and it is intended to facilitate machine language coding. Friedrich Kittler argued that high-level languages "*obscure* the operations of the hardware" (233). However, if programmers do not interpret high-level languages like BASIC, they must work with machine language which, being nothing but a series of numbers, is not suitable for programming. The last section of this chapter consists of a software studies approach, writing programs to interpret other programs, testing BASIC and 10PRINT's possibilities, "just as literary scholars study a text by generating more texts" (244). Finally, the Appendix presents the code and the output images of all the variations of 10PRINT.

Despite some detailed technical analyses of programming functions, particularly useful for those interested in computing, the interdisciplinary approach towards computation fuels the reader's curiosity. *10 PRINT CHR$(205.5+RND(1)); : GOTO 10* also represents the shift we are witnessing in the way books are conceived and distributed, since it is available for free as a PDF. In its print version (MIT Press, 2013), *10 PRINT* is a visually appealing volume (designed by co-author Casey Reas), displaying the program's exquisite white lines in a blue background. Like a bonus track, the print edition includes the '10 PRINT Companion Disk' (compiled by Martin Schemitsch), containing BASIC and assembly programs. This book proves that code may have as much to tell as any text and, in its rich and broad analysis, this collective work may well become a canonical example of the importance of a critical approach towards code and digital media in this particular moment of culture where computer users are often alienated from the knowledge of the systems on which they rely.