DEVELOPMENT OF A REDUNDANT FIBONACCI NUMERAL SYSTEM (NFNS) WITH A DUAL BASIS FOR FAULT-TOLERANT COMPUTING ENVIRONMENTS

Andrey V. Voron (anvoron1@yandex.ru)

September 19, 2026

Abstract

Modern positional numeral systems suffer from zero code redundancy, making internal data verification impossible. While the classical Fibonacci numeral system features natural redundancy, its level varies drastically across different numbers. To optimize and stabilize this mathematical framework, this paper formulates the Parity Constraint Rule to restrict code redundancy. Based on this rule, we develop a New Fibonacci Numeral System (NFNS) with a dual basis designed specifically for fault-tolerant computing environments.

1 Introduction

The most widely used positional systems include binary, ternary, octal, decimal, duodecimal, hexadecimal, vigesimal, and sexagesimal systems. There also exists the mixed "Fibonacci" numeral system for integers based on Fibonacci numbers. It is underpinned by Zeckendorf's theorem: any non-negative integer can be uniquely represented as a sum of one or more distinct Fibonacci numbers (excluding F_1) such that the representation does not include any two consecutive Fibonacci numbers. A universal bit-sequence code for natural numbers is constructed using this system. The combination "11" is strictly forbidden in the classical Fibonacci numeral system and serves as an end-of-record marker. A similar property of multi-valued representation (ambiguity) is also shared by George Bergman’s positional numeral system, which uses an irrational base known as the golden ratio.

Table 1 – Comparative representation of bit capacities with examples of codes for natural numbers and code redundancy in various numeral systems

A) BINARY NUMERAL SYSTEM								
Radix: 2	128	64	32	16	8	4	2	1
155	1	0	0	1	1	0	1	1
B) BERGMAN’S NUMERAL SYSTEM								
Radix: Golden ratio	4,236	2,618	1,618	1	0,618	0,382	0,236	0,146
8	1	1	0	1	0	0	0	1
8	1	1	0	0	1	1	0	1
8	1	0	1	1	1	1	0	1
C) FIBONACCI NUMERAL SYSTEM								
Radix: Fibonacci sequence numbers	21	13	8	5	3	2	1	1
8			1	0	0	0	0	0
8				1	1	0	0	0
8				1	0	1	1	0
8				1	0	1	0	1

Doctor of Technical Sciences, Professor Alexey Petrovich Stakhov (Figure) developed Fibonacci computer arithmetic [7] and put forward the fundamental concept of "Fibonacci computers" in the mid-1970s [2, 3, 4, 5].

Figure – Alexey Petrovich Stakhov

Speaking about the disadvantages of the conventional binary system used in modern microprocessors, the scientist points out a critical vulnerability: "The 'Trojan horse' of the binary system used in modern microprocessors is its zero redundancy. ... The lack of redundancy implies that all binary code combinations within the system are 'valid,' which makes it impossible to detect any errors" [6].

2 The main part

MATHEMATICAL FORMULATION OF
THE NEW FIBONACCI NUMERAL SYSTEM (NFNS)

Unlike the standard binary code, the Fibonacci code possesses natural redundancy. This manifests as an algorithmic ambiguity where a single natural number can have multiple valid representations. However, this redundancy is not constant and varies significantly across different numerical values. To optimize and stabilize the underlying mathematical framework, the author formulates Voron’s parity constraint rule:\[\text{Bit\ positions\ with\ even\ indices\ can\ act\ as\ valid\ code\ equivalents\ only\ when\ combined\ with\ odd-indexed\ positions.}\]Based on this constraint, we build the New Fibonacci Numeral System (NFNS), also classified as a Position parity-based NFNS. This restriction can be formulated as an alternative numerical theorem (analogous to Zeckendorf's theorem).

Voron’s Dual-Basis Theorem:

– any natural number can be represented in exactly two interconnected valid ways as a sum of distinct Fibonacci numbers such that its binary representation contains no adjacent active bits ("1") of the same position parity. Consequently, even-positioned bits of the Fibonacci sequence can only be active simultaneously with odd-positioned bits of the same sequence.

This mathematical formulation introduces a strict syntactic prohibition on certain binary sequences. Prohibited patterns in NFNS include any Fibonacci code combinations containing an uninterrupted odd number of zeros between two ones (Table 2).

Table 2 – Prohibited code combinations in NFNS architectures

Forbidden code combinations	
101 и 101	1000000000001 и 1000000000001
10001 и 10001	100000000000001 и 100000000000001
1000001 и 1000001	10000000000000001 и 10000000000000001
100000001 и 100000001	1000000000000000001 и 1000000000000000001
10000000001 и 10000000001	…

Due to the parity constraint rule, all natural numbers receive exactly two valid dual representations (Table 3).

Table 3 – Representation of natural numbers in Fibonacci code under Voron's parity constraint rule

Natural numbers	Fibonacci numbers	Fibonacci code	Natural numbers	Fibonacci numbers	Fibonacci code
1	1 и 1	1 и 1	8	5, 3 и 8	11000 и 100000
2	1, 1 и 2	11 и 100	9	5, 3, 1 и 8, 1	11001 и 100001
3	2, 1 и 3	110 и 1000	10	5, 3, 2 и 8, 2	11100 и 100100
4	2, 1, 1 и 3, 1	111 и 1001	11	5, 3, 2, 1 и 8, 2, 1	11110 и 100110
5	3, 2 и 5	1100 и 10000	12	5, 3, 2, 1, 1 и 8, 2, 1, 1	11111 и 100111
6	3, 2, 1 и 5, 1	1110 и 10010	13	8, 5 и 13	110000 и 1000000
7	3, 2, 1, 1 и 5, 1, 1	1111 и 10011	14	8, 5, 1 и 13, 1	110010 и 1000010

An example of positional bitwise encoding for the numbers 8 and 25 in the NFNS system with a fixed-parity bit grid is presented in Table 4.

Table 4 – Positional encoding paradigm within the NFNS system

Position No. / Parity	F_8/ Even	F_7/ Odd	F_6/ Even	F_5/ Odd	F_4/ Even	F_3/ Odd	F_2/ Even	F_1/ Odd
'NEW FIBONACCI NUMERAL SYSTEM' (NFNS)								
Radix: Fibonacci sequence numbers	21	13	8	5	3	2	1	1
8			1	0	0	0	0	0
8				1	1	0	0	0
25	1	0	0	0	0	1	1	1
25		1	1	0	0	1	1	1

The positional system based on the formulated rule (the non-congruence of bits with mismatched positional parities) is designated as the New Fibonacci Numeral System (NFNS), utilizing the NFNS-code.

The mathematical distribution logic of NFNS code structures across the natural number line is illustrated by a "dialectical" numerical expansion algorithm (Table 5). Complete code tables for random integers—contrasting generation modes with and without Voron's parity constraint rule—are provided in APPENDICES P1–P29.

Table 5 – Schematic hierarchical map of NFNS code groups along the natural number sequence

<table>
<tr><th>Sequence of numbers</th><th colspan="5">Group of numbers 5</th><th colspan="3">Group of numbers 3</th><th colspan="2">Group of numbers 2</th><th>Group of numbers 16</th><th>Group of numbers 1a</th><th>Group of numbers 0</th><th>digit position in a numeral system</th><th>Digit index / Parity of the digit position</th></tr>
<tr><td rowspan="2">1a</td><td colspan="12"></td><td>1</td><td></td><td></td></tr>
<tr><td colspan="12"></td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
<tr><td rowspan="2">1b</td><td colspan="12"></td><td>1</td><td></td><td></td></tr>
<tr><td colspan="12"></td><td>1</td><td>1</td><td>1 / Even position</td></tr>
<tr><td rowspan="3">1–2</td><td colspan="12"></td><td>2</td><td></td><td></td></tr>
<tr><td colspan="12"></td><td>1</td><td>1</td><td>2 / Even position</td></tr>
<tr><td colspan="12"></td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
<tr><td rowspan="4">3–4</td><td colspan="11"></td><td>3</td><td>4</td><td></td><td></td></tr>
<tr><td colspan="11"></td><td colspan="2">1</td><td>2</td><td>3 / Odd position</td></tr>
<tr><td colspan="11"></td><td colspan="2">1</td><td>1</td><td>2 / Even position</td></tr>
<tr><td colspan="11"></td><td>0</td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
<tr><td rowspan="5">5–7</td><td colspan="10"></td><td>5</td><td>6</td><td>7</td><td></td><td></td></tr>
<tr><td colspan="10"></td><td colspan="3">1</td><td>3</td><td>4 / Even position</td></tr>
<tr><td colspan="10"></td><td colspan="3">1</td><td>2</td><td>3 / Odd position</td></tr>
<tr><td colspan="10"></td><td>0</td><td colspan="2">1</td><td>1</td><td>2 / Even position</td></tr>
<tr><td colspan="10"></td><td colspan="2">0</td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
<tr><td rowspan="6">8–12</td><td colspan="8"></td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td></td><td></td></tr>
<tr><td colspan="8"></td><td colspan="5">1</td><td>5</td><td>5 / Odd position</td></tr>
<tr><td colspan="8"></td><td colspan="5">1</td><td>3</td><td>3 / Even position</td></tr>
<tr><td colspan="8"></td><td colspan="2">0</td><td colspan="3">1</td><td>2</td><td>3 / Odd position</td></tr>
<tr><td colspan="8"></td><td colspan="3">0</td><td colspan="2">1</td><td>1</td><td>2 / Even position</td></tr>
<tr><td colspan="8"></td><td>0</td><td>1</td><td colspan="2">0</td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
<tr><td rowspan="7">13–20</td><td colspan="5"></td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td></td><td></td></tr>
<tr><td colspan="5"></td><td colspan="8">1</td><td>8</td><td>6 / Even position</td></tr>
<tr><td colspan="5"></td><td colspan="8">1</td><td>5</td><td>5 / Odd position</td></tr>
<tr><td colspan="5"></td><td colspan="3">0</td><td colspan="5">1</td><td>3</td><td>3 / Even position</td></tr>
<tr><td colspan="5"></td><td colspan="5">0</td><td colspan="3">1</td><td>2</td><td>3 / Odd position</td></tr>
<tr><td colspan="5"></td><td>0</td><td colspan="2">1</td><td colspan="3">0</td><td colspan="2">1</td><td>1</td><td>2 / Even position</td></tr>
<tr><td colspan="5"></td><td colspan="2">0</td><td>1</td><td>0</td><td>1</td><td colspan="2">0</td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
<tr><td rowspan="8">21–33</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td>32</td><td>33</td><td></td><td></td></tr>
<tr><td colspan="13">1</td><td>13</td><td>7 / Odd position</td></tr>
<tr><td colspan="13">1</td><td>8</td><td>6 / Even position</td></tr>
<tr><td colspan="5">0</td><td colspan="8">1</td><td>5</td><td>5 / Odd position</td></tr>
<tr><td colspan="8">0</td><td colspan="5">1</td><td>3</td><td>4 / Even position</td></tr>
<tr><td colspan="2">0</td><td colspan="3">1</td><td colspan="5">0</td><td colspan="3">1</td><td>2</td><td>3 / Odd position</td></tr>
<tr><td colspan="3">0</td><td colspan="2">1</td><td>0</td><td colspan="2">1</td><td colspan="3">0</td><td colspan="2">1</td><td>1</td><td>2 / Even position</td></tr>
<tr><td>0</td><td>1</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>1</td><td>0</td><td>1</td><td colspan="2">0</td><td>1</td><td>1</td><td>1 / Odd position</td></tr>
</table>

ARITHMETIC OPERATION ARCHITECTURE IN NFNS

The development of an optimal arithmetic pipeline for NFNS must rely on the fundamental principles of classical Fibonacci arithmetic designed by Stakhov, combined with strict hardware execution of the author's parity constraint rule. Since odd numbers of consecutive zeros are forbidden between active bits, the mathematical apparatus shifts toward a permanent dual representation.

The implementation of arithmetic operations in NFNS is structured into four consecutive stages:

[BITWISE ADDITION]

↓

[ELIMINATION OF "TWOS" (CARRIES)]

↓

[REDUCTION TO THE NFNS CANONICAL FORM (RECODING)]

↓

[SELECTION OF ONE OF THE TWO ALLOWED REPRESENTATION OPTIONS]

1. Bitwise Addition (Abacus-style addition): Addition of two operands starts identically to classical redundant systems. A bitwise summation is performed without immediate carries or constraints. The resulting intermediate vector may temporarily contain the digit 2 within its alphabet (overflown binary alphabet).

2. Local Overflow Elimination (Stakhov's carry rules): To eliminate overflown digits (2), we apply the fundamental recurrence relation of Fibonacci numbers $(F_n + F_n = F_{n+1} + F_{n-2})$. Each occurrence of a two is iteratively replaced using the rule: $2F_i = F_{i+1} + F_{i-2}$. This process continues until the vector contains only 0 and 1 digits.

3. Macro-generation and Parity Filtering (Voron's Parity Filtering): Because classical Stakhov convolutions (011 → 100) can violate the parity constraint rule, the optimal method to standardize the vector is an automated re-encoding process (Look-Ahead Constraint Automata). Instead of infinite local convolutions/deconvolutions, the redundant Fibonacci number from step 2 is processed through a finite-state machine. The algorithm prunes all branches of the representation tree that contain forbidden combinations (odd sequences of zeros between ones).

4. Dual-form Assignment: According to the formulated theorem, exactly two valid code variants will remain at the output of the filter. The system assigns one as the Main Form (e.g., maximizing the highest active bit to accelerate numerical comparison operations) and stores the second as the Check Form in an isolated buffer for hardware error detection.

The efficiency of this approach lies in using mathematical redundancy for concurrent error detection during the computation process itself. If the finite-state machine detects that an intermediate number does not yield exactly two valid dual representations, the processor instantly flags a hardware exception.

MATHEMATICAL DESCRIPTION OF THE ADDITION ALGORITHM

Let numbers A and B be represented in NFNS as binary vectors:

$$A = \sum_{i=2}^{n} a_i F_{i+1}, \quad B = \sum_{i=2}^{n} b_i F_{i+1}, \quad a_i, b_i \in \{0, 1\}$$

Step 1: Form the intermediate sum vector S* = (s^*_n…., s^*_2) via bitwise summation:

$$s_i^* = a_i + b_i, \quad s_i^* \in \{0, 1, 2\}$$

Step 2: Apply carry formulas to resolve overflows (s*i = 2). To minimize parity fluctuations in the NFNS grid, the dynamic splitting of a two at index i is split into two distinct branches based on index parity.

If the bit index i is even (i = 2k):

$$2F_{2k} \longrightarrow F_{2k+1} + F_{2k-2}$$

The overflow from the even bit splits, sending a carry pulse (+1) to the higher adjacent odd bit (2k+1) and the lower even bit (2k-2).

If the bit index i is odd (i = 2k+1):

$$2F_{2k+1} \longrightarrow F_{2k+2} + F_{2k-1}$$

The overflow splits into the higher adjacent even bit (2k+2) and the lower odd bit (2k-1).

This cellular-automata step runs in parallel until all twos vanish. Because splitting strictly preserves the arithmetic invariant of the sum $(\sum s_i^* F_i = inv)$, the scalar numerical value remains unchanged.

Step 3: Eliminating forbidden structures (such as a 101 chain where ones occupy bits of different parities like F_5 and F_3) via local operations can cause infinite processor oscillations. Instead, a hardware Look-Ahead Constraint Automata executes a macro-expansion using the Fibonacci recurrence:

$$F_j + F_{j-1} \longleftrightarrow F_{j+1}$$

This chain expansion continues until the lengths of zero sequences between active units become strictly even (2, 4, 6), guaranteeing index parity congruence.

Step 4: Dual-basis management. The output forms two valid representations distributed as follows:

– main Form: The code shifted left (maximizing the older bit). Used for standard operations.

– check Form: The code shifted right. Maintained in the buffer. If the Main Form loses alignment with the Check Form during execution, the processor generates a hardware exception.

EXECUTION EXAMPLE: NFNS ADDITION (7 + 8)

Consider the bit weights of our numeral grid (Fibonacci numbers):

$F_1 = 1$ (1st bit, Odd)

$F_2 = 1$ (2nd bit, Even)

$F_3 = 2$ (3rd bit, Odd)

$F_4 = 3$ (4th bit, Even)

$F_5 = 5$ (5th bit, Odd)

$F_6 = 8$ (6th bit, Even)

$F_7 = 13$ (7th bit, Odd)

To add the operands, 7 and 8 are initially represented in their valid NFNS forms.
Number 7 (5 + 1 + 1 = 7) is encoded as 0010011.
Number 8 (8 = 8) is encoded as 0100000. We perform a bitwise vector addition of the operands to get the intermediate vector S^* (Table 6).

Table 6 – Vector addition of operands 7 and 8

Position (i)	**7 (F_7)**	**6 (F_6)**	**5 (F_5)**	**4 (F_4)**	**3 (F_3)**	**2 (F_2)**	**1 (F_1)**
Weights (F_i)	13	8	5	3	2	1	1
Operand A (7)	0	0	1	0	0	1	1
Operand B (8)	0	1	0	0	0	0	0
Vector S^*	**0**	**1**	**1**	**0**	**0**	**1**	**1**

Because the resulting configuration (0110011) fully satisfies Voron's parity constraint rule (the active bits at positions 6 and 5 are adjacent, which is allowed as a base transition, and positions 2 and 1 match parity rules), the syntax filter validates this state. The arithmetic result stabilizes instantly as two valid dual representations of the invariant value 15.

DISCUSSION: PRACTICAL APPLICATION SPECIFICS OF NFNS

The design complexity of NFNS code structures is offset by hardware-level advantages that are mathematically impossible within classic binary architectures.

Mitigation of Single-Event Upsets (SEU) in Aerospace Computing

Standard computing chips operating in aerospace environments are exposed to heavy ions and cosmic radiation. This causes Single-Event Upsets (SEUs), or "bit-flips" (0↔1). Modern aerospace engineering relies on expensive mitigation techniques: radiation-hardened (Rad-Hard) semiconductors on sapphire substrates utilizing large process nodes (e.g., 65–130 nm); Triple Modular Redundancy (TMR) with majority voting logic across three parallel processors; and software-level ECC/Hamming codes.

Implementing NFNS shifts fault tolerance directly into the mathematical code of the processor. In a standard binary string, a bit-flip can silently modify 0110 (number 6) into 1110 (number 14), causing a critical failure. In contrast, flipping any bit within an NFNS sequence instantly pushes the combination into the forbidden zone (e.g., creating an illegal single zero pattern like 101). The processor's hardware logic catches this syntactic anomaly and isolates the error before execution.

Ultra-Fast Parallel Arithmetic

A major bottleneck of standard binary adders is the ripple-carry carry propagation issue, where an operation like 1+1 in the least significant bit can propagate across the entire word length. This forces the processor to stall while waiting for electrical signals to stabilize.

In the NFNS architecture, thanks to bitwise overflow splitting and local convolution algorithms, carry pulses propagate only to adjacent bits. This bounds calculations locally, eliminates ripple-carry overhead, and enables the design of high-speed parallel adders on specialized silicon dies.

Cryptographic and Steganographic Potential

The constant presence of exactly two valid encoding states for any natural number (the dual basis, such as 1111 and 10011 for the number 7) provides a novel mathematical tool for information security. By alternating between these valid code variants across data arrays using a secret algorithmic key, engineers can embed hidden watermarks, authentication tags, or cryptographic markers directly into the executable compiled code of numerical datasets without altering their scalar mathematical value.

3 Conclusion

1. Traditional positional numeral systems possess zero code redundancy, making hardware-level internal verification impossible and leaving systems vulnerable to silent data corruption. The Fibonacci numeral system bypasses this limitation via multi-valued encoding representation, though its redundancy levels vary dynamically across the integer spectrum.

2. This paper formulated the Voron Parity Constraint Rule, which dictates that even-indexed bits can only be actively represented in conjunction with odd-indexed bits. This formalized the New Fibonacci Numeral System (NFNS), ensuring exactly two valid dual representations for any discrete numerical value.

3. We designed a four-stage arithmetic pipeline for addition within the NFNS framework. This pipeline includes bitwise summation, dynamic index-separated overflow splitting, automated finite-state reduction, and final dual-code assignment (Main and Check forms).

4. It is shown that the structural properties of NFNS provide a robust mathematical foundation for designing next-generation computing systems for extreme environments. We identified three primary vectors of efficiency: native immunity to radiation-induced SEU bit-flips, elimination of the ripple-carry propagation bottleneck to enable ultra-fast parallel addition, and steganographic data hiding capabilities embedded within dual-basis code mutations.

References

1. Bergman, G. A number system with an irrational base. Mathematics Magazine, 1957, no. 31, pp. 98–119.

2. Stakhov, A.P. "Fibonacci" binary arithmetic and its application for computer systems control. In: Odnorodnye Vychislitel'nye Sistemy i Sredy [Homogeneous Computing Systems and Environments]. Proceedings of the IV All-Union Conference. Kiev, Naukova Dumka, 1975. [In Russian]

3. Stakhov, A.P. "Fibonacci" binary positional numeral systems. In: Kodirovanie i Peredacha Diskretnykh Soobshcheniy v Sistemakh Svyazi [Coding and Transmission of Discrete Messages in Communication Systems]. Moscow, Nauka, 1976. [In Russian]

4. Stakhov, A.P. Redundant binary positional numeral systems. In: Odnorodnye Tsifrovye Vychislitel'nye i Integriruyushchie Struktury [Homogeneous Digital Computing and Integrating Structures], issue 2. Taganrog Radio Engineering Institute Publishing House, 1974. [In Russian]

5. Stakhov, A.P. Using the natural redundancy of "Fibonacci" numeral systems for computing systems diagnostics. Avtomatika i Vychislitel'naya Tekhnika [Automation and Computer Science], 1975, no. 6. [In Russian]

6. Stakhov, A.P. Kody Zolotoy Proportsii [Codes of the Golden Proportion]. Moscow, Radio i Svyaz, 1984. 152 p. [In Russian]

7. Stakhov, A.P. Fibonacci microprocessors as a basic innovation of the future technological paradigm altering the information security level of systems. Akademiya Trinitarizma [Academy of Trinitarism], Moscow, El. No. 77-6567, publ. 16759, August 16, 2011. [In Russian]

APPENDICES

Table 1 – Fibonacci codes for numbers 20, 32, 52, 84

Number / count of codes for a number	Fibonacci-base numeral system									
	55	34	21	13	8	5	3	2	1	1
20/4	Codes without using the redundancy constraint rule									
				1	0	1	0	1	0	0
				1	0	1	0	0	1	1
				1	**0**	**0**	**1**	**1**	**1**	**1**
					1	**1**	**1**	**1**	**1**	**1**
	Codes applying the redundancy constraint rule									
				1	**0**	**0**	**1**	**1**	**1**	**1**
					1	**1**	**1**	**1**	**1**	**1**
32/7	Codes without using the redundancy constraint rule									
			1	0	1	0	1	0	0	0
			1	0	1	0	0	1	1	0
			1	0	1	0	0	1	0	1
			1	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	0	0	1	1	1	0	1
				1	**1**	**1**	**1**	**1**	**1**	**0**
				1	1	1	1	1	0	1
	Codes applying the redundancy constraint rule									
			1	**0**	**0**	**1**	**1**	**1**	**1**	**0**
				1	**1**	**1**	**1**	**1**	**1**	**0**
52/7	Codes without using the redundancy constraint rule									
		1	0	1	0	1	0	0	0	0
		1	0	1	0	0	1	1	0	0
		1	0	1	0	0	1	0	1	1
		1	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
		1	0	0	1	1	1	0	1	1
			1	**1**	**1**	**1**	**1**	**1**	**0**	**0**
			1	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule									
		1	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
			1	**1**	**1**	**1**	**1**	**1**	**0**	**0**
84/10	Codes without using the redundancy constraint rule									
	1	0	1	0	1	0	0	0	0	0
	1	0	1	0	0	1	1	0	0	0
	1	0	1	0	0	1	0	1	1	0
	1	0	1	0	0	1	0	1	0	1
	1	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	1	1	1	0	1	1	0
	1	0	0	1	1	1	0	1	0	1
		1	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	1	1	1	0	1	1	0
		1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule									
	1	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**

Table 2 – Fibonacci codes for numbers 4, 6, 10, 16

Number / count of codes for a number	Fibonacci-base numeral system									
	55	34	21	13	8	5	3	2	1	1
4/3	Codes without using the redundancy constraint rule									
							1	0	1	0
							1	**0**	**0**	**1**
								1	**1**	**1**
	Codes applying the redundancy constraint rule									
							1	**0**	**0**	**1**
								1	**1**	**1**
6/4	Codes without using the redundancy constraint rule									
						1	**0**	**0**	**1**	**0**
						1	0	0	0	1
							1	**1**	**1**	**0**
							1	1	0	1
	Codes applying the redundancy constraint rule									
						1	**0**	**0**	**1**	**0**
							1	**1**	**1**	**0**
10/4	Codes without using the redundancy constraint rule									
					1	**0**	**0**	**1**	**0**	**0**
					1	0	0	0	1	1
						1	**1**	**1**	**0**	**0**
						1	1	0	1	1
	Codes applying the redundancy constraint rule									
					1	**0**	**0**	**1**	**0**	**0**
						1	**1**	**1**	**0**	**0**
16/6	Codes without using the redundancy constraint rule									
				1	**0**	**0**	**1**	**0**	**0**	**0**
				1	0	0	0	1	1	0
				1	0	0	0	1	0	1
					1	**1**	**1**	**0**	**0**	**0**
					1	1	0	1	1	0
					1	1	0	1	0	1
	Codes applying the redundancy constraint rule									
				1	**0**	**0**	**1**	**0**	**0**	**0**
					1	**1**	**1**	**0**	**0**	**0**

Table 3 – Fibonacci codes for numbers 12, 19, 31, 50

Number / count of codes for a number	Fibonacci-base numeral system									
	55	34	21	13	8	5	3	2	1	1
12/4	Codes without using the redundancy constraint rule									
					1	0	1	0	1	0
					1	0	1	0	0	1
					1	**0**	**0**	**1**	**1**	**1**
						1	**1**	**1**	**1**	**1**
	Codes applying the redundancy constraint rule									
					1	**0**	**0**	**1**	**1**	**1**
						1	**1**	**1**	**1**	**1**
19/6	Codes without using the redundancy constraint rule									
				1	0	1	0	0	1	0
				1	0	1	0	0	0	1
				1	**0**	**0**	**1**	**1**	**1**	**0**
				1	0	0	1	1	0	1
					1	**1**	**1**	**1**	**1**	**0**
					1	1	1	1	0	1
	Codes applying the redundancy constraint rule									
				1	**0**	**0**	**1**	**1**	**1**	**0**
					1	**1**	**1**	**1**	**1**	**0**
31/6	Codes without using the redundancy constraint rule									
			1	0	1	0	0	1	0	0
			1	0	1	0	0	0	1	1
			1	**0**	**0**	**1**	**1**	**1**	**0**	**0**
			1	0	0	1	1	0	1	1
				1	**1**	**1**	**1**	**1**	**0**	**0**
				1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule									
			1	**0**	**0**	**1**	**1**	**1**	**0**	**0**
				1	**1**	**1**	**1**	**1**	**0**	**0**
50/9	Codes without using the redundancy constraint rule									
		1	0	1	0	0	1	0	0	0
		1	0	1	0	0	0	1	1	0
		1	0	1	0	0	0	1	0	1
		1	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
		1	0	0	1	1	0	1	1	0
		1	0	0	1	1	0	1	0	1
			1	**1**	**1**	**1**	**1**	**0**	**0**	**0**
			1	1	1	1	0	1	1	0
			1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule									
		1	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
			1	**1**	**1**	**1**	**1**	**0**	**0**	**0**

Table 4 – Fibonacci codes for numbers 33, 53, 86, 139

Number / count of codes for a number	Fibonacci-base numeral system										
	89	55	34	21	13	8	5	3	2	1	1
33/5	Codes without using the redundancy constraint rule										
				1	0	1	0	1	0	1	0
				1	0	1	0	1	0	0	1
				1	0	1	0	0	1	1	1
				1	**0**	**0**	**1**	**1**	**1**	**1**	**1**
					1	**1**	**1**	**1**	**1**	**1**	**1**
		Codes applying the redundancy constraint rule									
				1	**0**	**0**	**1**	**1**	**1**	**1**	**1**
					1	**1**	**1**	**1**	**1**	**1**	**1**
53/8	Codes without using the redundancy constraint rule										
			1	0	1	0	1	0	0	1	0
			1	0	1	0	1	0	0	0	1
			1	0	1	0	0	1	1	0	0
			1	0	1	0	0	1	0	1	1
			1	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
			1	0	0	1	1	1	0	1	1
				1	**1**	**1**	**1**	**1**	**1**	**0**	**0**
				1	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule										
			1	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
				1	**1**	**1**	**1**	**1**	**1**	**0**	**0**
86/8	Codes without using the redundancy constraint rule										
		1	0	1	0	1	0	0	1	0	0
		1	0	1	0	1	0	0	0	1	1
		1	0	1	0	0	1	1	1	0	0
		1	0	1	0	0	1	1	0	1	1
		1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
		1	0	0	1	1	1	1	0	1	1
			1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
			1	1	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule										
		1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
			1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
139/12		Codes without using the redundancy constraint rule									
	1	0	1	0	1	0	0	1	0	0	0
	1	0	1	0	1	0	0	0	1	1	0
	1	0	1	0	1	0	0	0	1	0	1
	1	0	1	0	0	1	1	1	0	0	0
	1	0	1	0	0	1	1	0	1	1	0
	1	0	1	0	0	1	1	0	1	0	1
	1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	1	1	1	1	0	1	1	0
	1	0	0	1	1	1	1	0	1	0	1
		1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	1	1	1	1	0	1	1	0
		1	1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule										
	1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**

Table 5 – Fibonacci codes for numbers 9, 14, 23, 37

Number / count of codes for a number	Fibonacci-base numeral system									
	55	34	21	13	8	5	3	2	1	1
9/5	Codes without using the redundancy constraint rule									
					1	0	0	0	1	0
					1	**0**	**0**	**0**	**0**	**1**
						1	1	0	1	0
						1	**1**	**0**	**0**	**1**
						1	0	1	1	1
	Codes applying the redundancy constraint rule									
					1	**0**	**0**	**0**	**0**	**1**
						1	**1**	**0**	**0**	**1**
14/6	Codes without using the redundancy constraint rule									
				1	**0**	**0**	**0**	**0**	**1**	**0**
				1	0	0	0	0	0	1
					1	**1**	**0**	**0**	**1**	**0**
					1	1	0	0	0	1
					1	0	1	1	1	0
					1	0	1	1	0	1
	Codes applying the redundancy constraint rule									
				1	**0**	**0**	**0**	**0**	**1**	**0**
					1	**1**	**0**	**0**	**1**	**0**
23/6	Codes without using the redundancy constraint rule									
			1	**0**	**0**	**0**	**0**	**1**	**0**	**0**
			1	0	0	0	0	0	1	1
				1	**1**	**0**	**0**	**1**	**0**	**0**
				1	1	0	0	0	1	1
				1	0	1	1	1	0	0
				1	0	1	1	0	1	1
	Codes applying the redundancy constraint rule									
			1	**0**	**0**	**0**	**0**	**1**	**0**	**0**
				1	**1**	**0**	**0**	**1**	**0**	**0**
37/9	Codes without using the redundancy constraint rule									
		1	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
		1	0	0	0	0	0	1	1	0
		1	0	0	0	0	0	1	0	1
			1	**1**	**0**	**0**	**1**	**0**	**0**	**0**
			1	1	0	0	0	1	1	0
			1	1	0	0	0	1	0	1
			1	0	1	1	1	0	0	0
			1	0	1	1	0	1	1	0
			1	0	1	1	0	1	0	1
	Codes applying the redundancy constraint rule									
		1	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
			1	**1**	**0**	**0**	**1**	**0**	**0**	**0**

Table 6 – Fibonacci codes for numbers 15, 24, 39, 63

Number / count of codes for a number	Fibonacci-base numeral system									
	55	34	21	13	8	5	3	2	1	1
15/5	Codes without using the redundancy constraint rule									
				1	0	0	0	1	0	0
				1	**0**	**0**	**0**	**0**	**1**	**1**
					1	1	0	1	0	0
					1	**1**	**0**	**0**	**1**	**1**
					1	0	1	1	1	1
	Codes applying the redundancy constraint rule									
				1	**0**	**0**	**0**	**0**	**1**	**1**
					1	**1**	**0**	**0**	**1**	**1**
24/8	Codes without using the redundancy constraint rule									
			1	0	0	0	1	0	0	0
			1	**0**	**0**	**0**	**0**	**1**	**1**	**0**
			1	0	0	0	0	1	0	1
				1	1	0	1	0	0	0
				1	**1**	**0**	**0**	**1**	**1**	**0**
				1	1	0	0	1	0	1
				1	0	1	1	1	1	0
				1	0	1	1	1	0	1
	Codes applying the redundancy constraint rule									
			1	**0**	**0**	**0**	**0**	**1**	**1**	**0**
				1	**1**	**0**	**0**	**1**	**1**	**0**
39/8	Codes without using the redundancy constraint rule									
		1	0	0	0	1	0	0	0	0
		1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**
		1	0	0	0	0	1	0	1	1
			1	1	0	1	0	0	0	0
			1	**1**	**0**	**0**	**1**	**1**	**0**	**0**
			1	1	0	0	1	0	1	1
			1	0	1	1	1	1	0	0
			1	0	1	1	1	0	1	1
	Codes applying the redundancy constraint rule									
		1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**
			1	**1**	**0**	**0**	**1**	**1**	**0**	**0**
63/11	Codes without using the redundancy constraint rule									
	1	0	0	0	1	0	0	0	0	0
	1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
	1	0	0	0	0	1	0	1	1	0
	1	0	0	0	0	1	0	1	0	1
		1	1	0	1	0	0	0	0	0
		1	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
		1	1	0	0	1	0	1	1	0
		1	1	0	0	1	0	1	0	1
		1	0	1	1	1	1	0	0	0
		1	0	1	1	1	0	1	1	0
		1	0	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule									
	1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**

Table 7 – Fibonacci codes for numbers 25, 40, 65, 105

Number / count of codes for a number	Fibonacci-base numeral system										
	89	55	34	21	13	8	5	3	2	1	1
25/7	Codes without using the redundancy constraint rule										
				1	0	0	0	1	0	1	0
				1	0	0	0	1	0	0	1
				1	**0**	**0**	**0**	**0**	**1**	**1**	**1**
					1	1	0	1	0	1	0
					1	1	0	1	0	0	1
					1	**1**	**0**	**0**	**1**	**1**	**1**
					1	0	1	1	1	1	1
		Codes applying the redundancy constraint rule									
				1	**0**	**0**	**0**	**0**	**1**	**1**	**1**
					1	**1**	**0**	**0**	**1**	**1**	**1**
40/10	Codes without using the redundancy constraint rule										
			1	0	0	0	1	0	0	1	0
			1	0	0	0	1	0	0	0	1
			1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**
			1	0	0	0	0	1	1	0	1
				1	1	0	1	0	0	1	0
				1	1	0	1	0	0	0	1
				1	**1**	**0**	**0**	**1**	**1**	**1**	**0**
				1	1	0	0	1	1	0	1
				1	0	1	1	1	1	1	0
				1	0	1	1	1	1	0	1
	Codes applying the redundancy constraint rule										
			1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**
				1	**1**	**0**	**0**	**1**	**1**	**1**	**0**
65/10	Codes without using the redundancy constraint rule										
		1	0	0	0	1	0	0	1	0	0
		1	0	0	0	1	0	0	0	1	1
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
		1	0	0	0	0	1	1	0	1	1
			1	1	0	1	0	0	1	0	0
			1	1	0	1	0	0	0	1	1
			1	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
			1	1	0	0	1	1	0	1	1
			1	0	1	1	1	1	1	0	0
			1	0	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule										
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
			1	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
105/15		Codes without using the redundancy constraint rule									
	1	0	0	0	1	0	0	1	0	0	0
	1	0	0	0	1	0	0	0	1	1	0
	1	0	0	0	1	0	0	0	1	0	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	0	0	1	1	0	1	1	0
	1	0	0	0	0	1	1	0	1	0	1
		1	1	0	1	0	0	1	0	0	0
		1	1	0	1	0	0	0	1	1	0
		1	1	0	1	0	0	0	1	0	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	0	0	1	1	0	1	1	0
		1	1	0	0	1	1	0	1	0	1
		1	0	1	1	1	1	1	0	0	0
		1	0	1	1	1	1	0	1	1	0
		1	0	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule										
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**

Table 8 – Fibonacci codes for numbers 41, 66, 107

Number / count of codes for a number	Fibonacci-base numeral system										
	89	55	34	21	13	8	5	3	2	1	1
41/7	Codes without using the redundancy constraint rule										
			1	0	0	0	1	0	1	0	0
			1	0	0	0	1	0	0	1	1
			1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
				1	1	0	1	0	1	0	0
				1	1	0	1	0	0	1	1
				1	**1**	**0**	**0**	**1**	**1**	**1**	**1**
				1	0	1	1	1	1	1	1
	Codes applying the redundancy constraint rule										
			1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
				1	**1**	**0**	**0**	**1**	**1**	**1**	**1**
66/12	Codes without using the redundancy constraint rule										
		1	0	0	0	1	0	1	0	0	0
		1	0	0	0	1	0	0	1	1	0
		1	0	0	0	1	0	0	1	0	1
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
		1	0	0	0	0	1	1	1	0	1
			1	1	0	1	0	1	0	0	0
			1	1	0	1	0	0	1	1	0
			1	1	0	1	0	0	1	0	1
			1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	1	0	0	1	1	1	0	1
			1	0	1	1	1	1	1	1	0
			1	0	1	1	1	1	1	0	1
	Codes applying the redundancy constraint rule										
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
107/12	Codes without using the redundancy constraint rule										
	1	0	0	0	1	0	1	0	0	0	0
	1	0	0	0	1	0	0	1	1	0	0
	1	0	0	0	1	0	0	1	0	1	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
	1	0	0	0	0	1	1	1	0	1	1
		1	1	0	1	0	1	0	0	0	0
		1	1	0	1	0	0	1	1	0	0
		1	1	0	1	0	0	1	0	1	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
		1	1	0	0	1	1	1	0	1	1
		1	0	1	1	1	1	1	1	0	0
		1	0	1	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule										
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**

Table 9 – Fibonacci codes for numbers 17, 27, 44, 71

Number / count of codes for a number	Fibonacci-base numeral system									
	55	34	21	13	8	5	3	2	1	1
17/6	Codes without using the redundancy constraint rule									
				1	0	0	1	0	1	0
				1	**0**	**0**	**1**	**0**	**0**	**1**
				1	0	0	0	1	1	1
					1	1	1	0	1	0
					1	**1**	**1**	**0**	**0**	**1**
					1	1	0	1	1	1
	Codes applying the redundancy constraint rule									
				1	**0**	**0**	**1**	**0**	**0**	**1**
					1	**1**	**1**	**0**	**0**	**1**
27/8	Codes without using the redundancy constraint rule									
			1	**0**	**0**	**1**	**0**	**0**	**1**	**0**
			1	0	0	1	0	0	0	1
			1	0	0	0	1	1	1	0
			1	0	0	0	1	1	0	1
				1	**1**	**1**	**0**	**0**	**1**	**0**
				1	1	1	0	0	0	1
				1	1	0	1	1	1	0
				1	1	0	1	1	0	1
	Codes applying the redundancy constraint rule									
			1	**0**	**0**	**1**	**0**	**0**	**1**	**0**
				1	**1**	**1**	**0**	**0**	**1**	**0**
44/8	Codes without using the redundancy constraint rule									
		1	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**
		1	0	0	1	0	0	0	1	1
		1	0	0	0	1	1	1	0	0
		1	0	0	0	1	1	0	1	1
			1	**1**	**1**	**0**	**0**	**1**	**0**	**0**
			1	1	1	0	0	0	1	1
			1	1	0	1	1	1	0	0
			1	1	0	1	1	0	1	1
	Codes applying the redundancy constraint rule									
		1	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**
			1	**1**	**1**	**0**	**0**	**1**	**0**	**0**
71/12	Codes without using the redundancy constraint rule									
	1	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
	1	0	0	1	0	0	0	1	1	0
	1	0	0	1	0	0	0	1	0	1
	1	0	0	0	1	1	1	0	0	0
	1	0	0	0	1	1	0	1	1	0
	1	0	0	0	1	1	0	1	0	1
		1	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
		1	1	1	0	0	0	1	1	0
		1	1	1	0	0	0	1	0	1
		1	1	0	1	1	1	0	0	0
		1	1	0	1	1	0	1	1	0
		1	1	0	1	1	0	1	0	1
	Codes applying the redundancy constraint rule									
	1	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**

Table 10 – Fibonacci codes for numbers 28, 45, 73, 118

Number / count of codes for a number	Fibonacci-base numeral system										
	89	55	34	21	13	8	5	3	2	1	1
28/6	Codes without using the redundancy constraint rule										
				1	0	0	1	0	1	0	0
				1	**0**	**0**	**1**	**0**	**0**	**1**	**1**
				1	0	0	0	1	1	1	1
					1	1	1	0	1	0	0
					1	**1**	**1**	**0**	**0**	**1**	**1**
					1	1	0	1	1	1	1
	Codes applying the redundancy constraint rule										
				1	**0**	**0**	**1**	**0**	**0**	**1**	**1**
					1	**1**	**1**	**0**	**0**	**1**	**1**
45/10	Codes without using the redundancy constraint rule										
			1	0	0	1	0	1	0	0	0
			1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**
			1	0	0	1	0	0	1	0	1
			1	0	0	0	1	1	1	1	0
			1	0	0	0	1	1	1	0	1
				1	1	1	0	1	0	0	0
				1	**1**	**1**	**0**	**0**	**1**	**1**	**0**
				1	1	1	0	0	1	0	1
				1	1	0	1	1	1	1	0
				1	1	0	1	1	1	0	1
	Codes applying the redundancy constraint rule										
			1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**
				1	**1**	**1**	**0**	**0**	**1**	**1**	**0**
73/10	Codes without using the redundancy constraint rule										
		1	0	0	1	0	1	0	0	0	0
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**
		1	0	0	1	0	0	1	0	1	1
		1	0	0	0	1	1	1	1	0	0
		1	0	0	0	1	1	1	0	1	1
			1	1	1	0	1	0	0	0	0
			1	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**
			1	1	1	0	0	1	0	1	1
			1	1	0	1	1	1	1	0	0
			1	1	0	1	1	1	0	1	1
	Codes applying the redundancy constraint rule										
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**
			1	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**
118/14	Codes without using the redundancy constraint rule										
	1	0	0	1	0	1	0	0	0	0	0
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
	1	0	0	1	0	0	1	0	1	1	0
	1	0	0	1	0	0	1	0	1	0	1
	1	0	0	0	1	1	1	1	0	0	0
	1	0	0	0	1	1	1	0	1	1	0
	1	0	0	0	1	1	1	0	1	0	1
		1	1	1	0	1	0	0	0	0	0
		1	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
		1	1	1	0	0	1	0	1	1	0
		1	1	1	0	0	1	0	1	0	1
		1	1	0	1	1	1	1	0	0	0
		1	1	0	1	1	1	0	1	1	0
		1	1	0	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule										
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**

Table 11 – Fibonacci codes for numbers 46, 74, 120, 194

Number / count of codes for a number	Fibonacci-base numeral system										
	89	55	34	21	13	8	5	3	2	1	1
46/8	Codes without using the redundancy constraint rule										
			1	0	0	1	0	1	0	1	0
			1	0	0	1	0	1	0	0	1
			1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**
			1	0	0	0	1	1	1	1	1
				1	1	1	0	1	0	1	0
				1	1	1	0	1	0	0	1
				1	**1**	**1**	**0**	**0**	**1**	**1**	**1**
				1	1	0	1	1	1	1	1
		Codes applying the redundancy constraint rule									
			1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**
				1	**1**	**1**	**0**	**0**	**1**	**1**	**1**
74/12	Codes without using the redundancy constraint rule										
		1	0	0	1	0	1	0	0	1	0
		1	0	0	1	0	1	0	0	0	1
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**
		1	0	0	1	0	0	1	1	0	1
		1	0	0	0	1	1	1	1	1	0
		1	0	0	0	1	1	1	1	0	1
			1	1	1	0	1	0	0	1	0
			1	1	1	0	1	0	0	0	1
			1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**
			1	1	1	0	0	1	1	0	1
			1	1	0	1	1	1	1	1	0
			1	1	0	1	1	1	1	0	1
	Codes applying the redundancy constraint rule										
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**
			1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**
120/12	Codes without using the redundancy constraint rule										
	1	0	0	1	0	1	0	0	1	0	0
	1	0	0	1	0	1	0	0	1	1	1
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
	1	0	0	1	0	0	1	1	1	1	1
	1	0	0	0	1	1	1	1	1	0	0
	1	0	0	0	1	1	1	1	1	1	1
		1	1	1	0	1	0	0	1	0	0
		1	1	1	0	1	0	0	1	1	1
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
		1	1	1	0	0	1	1	1	1	1
		1	1	0	1	1	1	1	1	0	0
		1	1	0	1	1	1	1	1	1	1
	Codes applying the redundancy constraint rule										
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**

Table 12 – Fibonacci codes for numbers 75, 121, 196

Number / count of codes for a number	Fibonacci-base numeral system											
	144	89	55	34	21	13	8	5	3	2	1	1
75/8	Codes without using the redundancy constraint rule											
			1	0	0	1	0	1	0	1	0	0
			1	0	0	1	0	1	0	0	1	1
			1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
			1	0	0	0	1	1	1	1	1	1
				1	1	1	0	1	0	1	0	0
				1	1	1	0	1	0	0	1	1
				1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
				1	1	0	1	1	1	1	1	1
	Codes applying the redundancy constraint rule											
			1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
				1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
121/14		Codes without using the redundancy constraint rule										
		1	0	0	1	0	1	0	1	0	0	0
		1	0	0	1	0	1	0	0	1	1	0
		1	0	0	1	0	1	0	0	1	0	1
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
		1	0	0	1	0	0	1	1	1	0	1
		1	0	0	0	1	1	1	1	1	1	0
		1	0	0	0	1	1	1	1	1	0	1
			1	1	1	0	1	0	1	0	0	0
			1	1	1	0	1	0	0	1	1	0
			1	1	1	0	1	0	0	1	0	1
			1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	1	1	0	0	1	1	1	0	1
			1	1	0	1	1	1	1	1	1	0
			1	1	0	1	1	1	1	1	0	1
	Codes applying the redundancy constraint rule											
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
196/14	Codes without using the redundancy constraint rule											
	1	0	0	1	0	1	0	1	0	0	0	0
	1	0	0	1	0	1	0	0	1	1	0	0
	1	0	0	1	0	1	0	0	1	0	1	1
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
	1	0	0	1	0	0	1	1	1	0	1	1
	1	0	0	0	1	1	1	1	1	1	0	0
	1	0	0	0	1	1	1	1	1	0	1	1
		1	1	1	0	1	0	1	0	0	0	0
		1	1	1	0	1	0	0	1	1	0	0
		1	1	1	0	1	0	0	1	0	1	1
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
		1	1	1	0	0	1	1	1	0	1	1
		1	1	0	1	1	1	1	1	1	0	0
		1	1	0	1	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule											
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**

Table 13 – Fibonacci codes for numbers 67, 108, 175

Number / count of codes for a number	Fibonacci-base numeral system											
	144	89	55	34	21	13	8	5	3	2	1	1
67/9	Codes without using the redundancy constraint rule											
			1	0	0	0	1	0	1	0	1	0
			1	0	0	0	1	0	1	0	0	1
			1	0	0	0	1	0	0	1	1	1
			1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
				1	1	0	1	0	1	0	1	0
				1	1	0	1	0	1	0	0	1
				1	1	0	1	0	0	1	1	1
				1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
				1	0	1	1	1	1	1	1	1
	Codes applying the redundancy constraint rule											
			1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
				1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
108/14		Codes without using the redundancy constraint rule										
		1	0	0	0	1	0	1	0	0	1	0
		1	0	0	0	1	0	1	0	0	0	1
		1	0	0	0	1	0	0	1	1	1	0
		1	0	0	0	1	0	0	1	1	0	1
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
		1	0	0	0	0	1	1	1	1	0	1
			1	1	0	1	0	1	0	0	1	0
			1	1	0	1	0	1	0	0	0	1
			1	1	0	1	0	0	1	1	1	0
			1	1	0	1	0	0	1	1	0	1
			1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
			1	1	0	0	1	1	1	1	0	1
			1	0	1	1	1	1	1	1	1	0
			1	0	1	1	1	1	1	1	0	1
	Codes applying the redundancy constraint rule											
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
			1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
175/14	Codes without using the redundancy constraint rule											
	1	0	0	0	1	0	1	0	0	1	0	0
	1	0	0	0	1	0	1	0	0	0	1	1
	1	0	0	0	1	0	0	1	1	1	0	0
	1	0	0	0	1	0	0	1	1	0	1	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
	1	0	0	0	0	1	1	1	1	0	1	1
		1	1	0	1	0	1	0	0	1	0	0
		1	1	0	1	0	1	0	0	0	1	1
		1	1	0	1	0	0	1	1	1	0	0
		1	1	0	1	0	0	1	1	0	1	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
		1	1	0	0	1	1	1	1	0	1	1
		1	0	1	1	1	1	1	1	1	0	0
		1	0	1	1	1	1	1	1	0	1	1
	Codes applying the redundancy constraint rule											
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**

Table 14 – Fibonacci coding of a number 194

Number / count of codes for a number	Fibonacci-base numeral system											
	144	89	55	34	21	13	8	5	3	2	1	1
194/18			Codes without using the redundancy constraint rule									
	1	0	0	1	0	1	0	0	1	0	0	0
	1	0	0	1	0	1	0	0	0	1	1	0
	1	0	0	1	0	1	0	0	0	1	0	1
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	1	0	0	1	1	0	1	1	0
	1	0	0	1	0	0	1	1	0	1	0	1
	1	0	0	0	1	1	1	1	1	0	0	0
	1	0	0	0	1	1	1	1	0	1	1	0
	1	0	0	0	1	1	1	1	0	1	0	1
		1	1	1	0	1	0	0	1	0	0	0
		1	1	1	0	1	0	0	0	1	1	0
		1	1	1	0	1	0	0	0	1	0	1
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	1	0	0	1	1	0	1	1	0
		1	1	1	0	0	1	1	0	1	0	1
		1	1	0	1	1	1	1	1	0	0	0
		1	1	0	1	1	1	1	0	1	1	0
		1	1	0	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule											
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**

Table 15 – Fibonacci codes for numbers 665, 699

Number / count of codes for a number	Fibonacci-base numeral system														
	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
665/18	Codes without using the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0
	1	0	0	0	0	0	1	0	1	1	0	0	0	0	0
	1	0	0	0	0	0	1	0	1	0	1	1	0	0	0
	1	0	0	0	0	0	1	0	1	0	1	0	1	1	0
	1	0	0	0	0	0	1	0	1	0	1	0	1	0	1
		1	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		1	1	0	0	0	1	1	0	0	0	0	0	0	0
		1	1	0	0	0	1	0	1	1	0	0	0	0	0
		1	1	0	0	0	1	0	1	0	1	1	0	0	0
		1	1	0	0	0	1	0	1	0	1	0	1	1	0
		1	1	0	0	0	1	0	1	0	1	0	1	0	1
		1	0	1	1	0	1	1	0	0	0	0	0	0	0
		1	0	1	1	0	1	0	1	1	0	0	0	0	0
		1	0	1	1	0	1	0	1	0	1	1	0	0	0
		1	0	1	1	0	1	0	1	0	1	0	1	1	0
		1	0	1	1	0	1	0	1	0	1	0	1	0	1
		1	0	1	1	0	1	1	0	0	0	0	0	0	0
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
699/17	Codes without using the redundancy constraint rule														
	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0
	1	0	0	0	0	1	0	1	0	1	1	0	0	0	0
	1	0	0	0	0	1	0	1	0	1	0	1	1	0	0
	1	0	0	0	0	1	0	1	0	1	0	1	0	1	1
		1	1	0	1	0	0	0	0	0	0	0	0	0	0
		1	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		1	1	0	0	1	0	1	1	0	0	0	0	0	0
		1	1	0	0	1	0	1	0	1	1	0	0	0	0
		1	1	0	0	1	0	1	0	1	0	1	1	0	0
		1	1	0	0	1	0	1	0	1	0	1	0	1	1
		1	0	1	1	1	1	0	0	0	0	0	0	0	0
		1	0	1	1	1	0	1	1	0	0	0	0	0	0
		1	0	1	1	1	0	1	0	1	1	0	0	0	0
		1	0	1	1	1	0	1	0	1	0	1	1	0	0
		1	0	1	1	1	0	1	0	1	0	1	0	1	1
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table 16 – Fibonacci coding of a number 720

Number / count of codes for a number	Fibonacci-base numeral system														
	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
720/25	Codes without using the redundancy constraint rule														
	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0
	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0
	1	0	0	0	1	0	0	0	1	0	1	1	0	0	0
	1	0	0	0	1	0	0	0	1	0	1	0	1	1	0
	1	0	0	0	1	0	0	0	1	0	1	0	1	0	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	0	0	0	0	1	1	0	1	1	0	0	0	0	0
	1	0	0	0	0	1	1	0	1	0	1	1	0	0	0
	1	0	0	0	0	1	1	0	1	0	1	0	1	1	0
	1	0	0	0	0	1	1	0	1	0	1	0	1	0	1
		1	1	0	1	0	0	1	0	0	0	0	0	0	0
		1	1	0	1	0	0	0	1	1	0	0	0	0	0
		1	1	0	1	0	0	0	1	0	1	1	0	0	0
		1	1	0	1	0	0	0	1	0	1	0	1	1	0
		1	1	0	1	0	0	0	1	0	1	0	1	0	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		1	1	0	0	1	1	0	1	1	0	0	0	0	0
		1	1	0	0	1	1	0	1	0	1	1	0	0	0
		1	1	0	0	1	1	0	1	0	1	0	1	1	0
		1	1	0	0	1	1	0	1	0	1	0	1	0	1
		1	0	1	1	1	1	1	0	0	0	0	0	0	0
		1	0	1	1	1	1	0	1	1	0	0	0	0	0
		1	0	1	1	1	1	0	1	0	1	1	0	0	0
		1	0	1	1	1	1	0	1	0	1	0	1	1	0
		1	0	1	1	1	1	0	1	0	1	0	1	0	1
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table 17 – Fibonacci coding of a number 733

Number / count of codes for a number	Fibonacci-base numeral system														
	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
733/22	Codes without using the redundancy constraint rule														
	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0
	1	0	0	0	1	0	0	1	1	0	0	0	0	0	0
	1	0	0	0	1	0	0	1	0	1	1	0	0	0	0
	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0
	1	0	0	0	1	0	0	1	0	1	0	1	0	1	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	1	0	0	0	0	1	1	1	0	1	1	0	0	0	0
	1	0	0	0	0	1	1	1	0	1	0	1	1	0	0
	1	0	0	0	0	1	1	1	0	1	0	1	0	1	1
		1	1	0	1	0	1	0	0	0	0	0	0	0	0
		1	1	0	1	0	0	1	1	0	0	0	0	0	0
		1	1	0	1	0	0	1	0	1	1	0	0	0	0
		1	1	0	1	0	0	1	0	1	0	1	1	0	0
		1	1	0	1	0	0	1	0	1	0	1	0	1	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
		1	1	0	0	1	1	1	0	1	1	0	0	0	0
		1	1	0	0	1	1	1	0	1	0	1	1	0	0
		1	1	0	0	1	1	1	0	1	0	1	0	1	1
		1	0	1	1	1	1	1	1	0	0	0	0	0	0
		1	0	1	1	1	1	1	0	1	1	0	0	0	0
		1	0	1	1	1	1	1	0	1	0	1	1	0	0
		1	0	1	1	1	1	1	0	1	0	1	0	1	1
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**

Table 18 – Fibonacci coding of a number 741

Number / count of codes for a number	Fibonacci-base numeral system														
	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
	Codes without using the redundancy constraint rule														
741/28	1	0	0	0	1	0	1	0	0	1	0	0	0	0	0
	1	0	0	0	1	0	1	0	0	0	1	1	0	0	0
	1	0	0	0	1	0	1	0	0	0	1	0	1	1	0
	1	0	0	0	1	0	1	0	0	0	1	0	1	0	1
	1	0	0	0	1	0	0	1	1	1	0	0	0	0	0
	1	0	0	0	1	0	0	1	1	0	1	1	0	0	0
	1	0	0	0	1	0	0	1	1	0	1	0	1	1	0
	1	0	0	0	1	0	0	1	1	0	1	0	1	0	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
	1	0	0	0	0	1	1	1	1	0	1	1	0	0	0
	1	0	0	0	0	1	1	1	1	0	1	0	1	1	0
	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1
		1	1	0	1	0	1	0	0	1	0	0	0	0	0
		1	1	0	1	0	1	0	0	0	1	1	0	0	0
		1	1	0	1	0	1	0	0	0	1	0	1	1	0
		1	1	0	1	0	1	0	0	0	1	0	1	0	1
		1	1	0	1	0	0	1	1	1	0	0	0	0	0
		1	1	0	1	0	0	1	1	0	1	1	0	0	0
		1	1	0	1	0	0	1	1	0	1	0	1	1	0
		1	1	0	1	0	0	1	1	0	1	0	1	0	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
		1	1	0	0	1	1	1	1	0	1	1	0	0	0
		1	1	0	0	1	1	1	1	0	1	0	1	1	0
		1	1	0	0	1	1	1	1	0	1	0	1	0	1
		1	0	1	1	1	1	1	1	1	0	0	0	0	0
		1	0	1	1	1	1	1	1	0	1	1	0	0	0
		1	0	1	1	1	1	1	1	0	1	0	1	1	0
		1	0	1	1	1	1	1	1	0	1	0	1	0	1
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**

Table 19 – Fibonacci coding of a number 746

Number / count of codes for a number	Fibonacci-base numeral system														
	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
746/23	Codes without using the redundancy constraint rule														
	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0
	1	0	0	0	1	0	1	0	0	1	1	0	0	0	0
	1	0	0	0	1	0	1	0	0	1	0	1	1	0	0
	1	0	0	0	1	0	1	0	0	1	0	1	0	1	1
	1	0	0	0	1	0	0	1	1	1	1	0	0	0	0
	1	0	0	0	1	0	0	1	1	1	0	1	1	0	0
	1	0	0	0	1	0	0	1	1	1	0	1	0	1	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
	1	0	0	0	0	1	1	1	1	1	0	1	1	0	0
	1	0	0	0	0	1	1	1	1	1	0	1	0	1	1
		1	1	0	1	0	1	0	1	0	0	0	0	0	0
		1	1	0	1	0	1	0	0	1	1	0	0	0	0
		1	1	0	1	0	1	0	0	1	0	1	1	0	0
		1	1	0	1	0	1	0	0	1	0	1	0	1	1
		1	1	0	1	0	0	1	1	1	1	0	0	0	0
		1	1	0	1	0	0	1	1	1	0	1	1	0	0
		1	1	0	1	0	0	1	1	1	0	1	0	1	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
		1	1	0	0	1	1	1	1	1	0	1	1	0	0
		1	1	0	0	1	1	1	1	1	0	1	0	1	1
		1	0	1	1	1	1	1	1	1	1	0	0	0	0
		1	0	1	1	1	1	1	1	1	0	1	1	0	0
		1	0	1	1	1	1	1	1	1	0	1	0	1	1
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**

Table 20 – Fibonacci coding of a number 749

Number / count of codes for a number	Fibonacci-base numeral system														
	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
749/27	Codes without using the redundancy constraint rule														
	1	0	0	0	1	0	1	0	1	0	0	1	0	0	0
	1	0	0	0	1	0	1	0	1	0	0	0	1	1	0
	1	0	0	0	1	0	1	0	1	0	0	0	1	0	1
	1	0	0	0	1	0	1	0	0	1	1	1	0	0	0
	1	0	0	0	1	0	1	0	0	1	1	0	1	1	0
	1	0	0	0	1	0	1	0	0	1	1	0	1	0	1
	1	0	0	0	1	0	0	1	1	1	1	1	0	0	0
	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0
	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	0	0	1	1	1	1	1	1	0	1	1	0
	1	0	0	0	0	1	1	1	1	1	1	0	1	0	1
		1	1	0	1	0	1	0	1	0	0	1	0	0	0
		1	1	0	1	0	1	0	1	0	0	0	1	1	0
		1	1	0	1	0	1	0	1	0	0	0	1	0	1
		1	1	0	1	0	1	0	0	1	1	1	0	0	0
		1	1	0	1	0	1	0	0	1	1	0	1	1	0
		1	1	0	1	0	1	0	0	1	1	0	1	0	1
		1	1	0	1	0	0	1	1	1	1	1	0	0	0
		1	1	0	1	0	0	1	1	1	1	0	1	1	0
		1	1	0	1	0	0	1	1	1	1	0	1	0	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	0	0	1	1	1	1	1	1	0	1	1	0
		1	1	0	0	1	1	1	1	1	1	0	1	0	1
		1	0	1	1	1	1	1	1	1	1	1	0	0	0
		1	0	1	1	1	1	1	1	1	1	0	1	1	0
		1	0	1	1	1	1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule														
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**

Table 21 – Fibonacci codes for numbers 122, 197

Number / count of codes for a number	Fibonacci-base numeral system											
	144	89	55	34	21	13	8	5	3	2	1	1
122/10	Codes without using the redundancy constraint rule											
		1	0	0	1	0	1	0	1	0	1	0
		1	0	0	1	0	1	0	1	0	0	1
		1	0	0	1	0	1	0	0	1	1	0
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
		1	0	0	0	1	1	1	1	1	1	0
			1	1	1	0	1	0	1	0	1	0
			1	1	1	0	1	0	1	0	0	1
			1	1	1	0	1	0	0	1	1	0
			1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	1	0	1	1	1	1	1	1	0
	Codes applying the redundancy constraint rule											
		1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
			1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
197/16	Codes without using the redundancy constraint rule											
	1	0	0	1	0	1	0	1	0	0	1	0
	1	0	0	1	0	1	0	1	0	0	0	1
	1	0	0	1	0	1	0	0	1	1	1	0
	1	0	0	1	0	1	0	0	1	1	0	1
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
	1	0	0	1	0	0	1	1	1	1	0	1
	1	0	0	0	1	1	1	1	1	1	1	0
	1	0	0	0	1	1	1	1	1	1	0	1
		1	1	1	0	1	0	1	0	0	1	0
		1	1	1	0	1	0	1	0	0	0	1
		1	1	1	0	1	0	0	1	1	1	0
		1	1	1	0	1	0	0	1	1	0	1
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
		1	1	1	0	0	1	1	1	1	0	1
		1	1	0	1	1	1	1	1	1	1	0
		1	1	0	1	1	1	1	1	1	0	1
	Codes applying the redundancy constraint rule											
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**

Table 22 – Fibonacci codes for numbers 109, 176

Number / count of codes for a number	Fibonacci-base numeral system										
	144	89	55	34	21	13	8	5	3	2	1	1
109/9	Codes without using the redundancy constraint rule											
		1	0	0	0	1	0	1	0	1	0	0
		1	0	0	0	1	0	1	0	0	1	1
		1	0	0	0	1	0	0	1	1	0	0
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
			1	1	0	1	0	1	0	1	0	0
			1	1	0	1	0	1	0	0	1	1
			1	1	0	1	0	0	1	1	0	0
			1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
			1	0	1	1	1	1	1	1	0	0
	Codes applying the redundancy constraint rule											
		1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
			1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**
176/16	Codes without using the redundancy constraint rule											
	1	0	0	0	1	0	1	0	1	0	0	0
	1	0	0	0	1	0	1	0	0	1	1	0
	1	0	0	0	1	0	1	0	0	1	0	1
	1	0	0	0	1	0	0	1	1	1	1	0
	1	0	0	0	1	0	0	1	1	1	0	1
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**
	1	0	0	0	0	1	1	1	1	1	0	1
		1	1	0	1	0	1	0	1	0	0	0
		1	1	0	1	0	1	0	0	1	1	0
		1	1	0	1	0	1	0	0	1	0	1
		1	1	0	1	0	0	1	1	1	1	0
		1	1	0	1	0	0	1	1	1	0	1
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**
		1	1	0	0	1	1	1	1	1	0	1
		1	0	1	1	1	1	1	1	1	1	0
		1	0	1	1	1	1	1	1	1	0	1
	Codes applying the redundancy constraint rule											
	1	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**
		1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**

Table 23 – Fibonacci codes for numbers 225, 364

Number / count of codes for a number	Fibonacci-base numeral system												
	233	144	89	55	34	21	13	8	5	3	2	1	1
225/12	Codes without using the redundancy constraint rule												
		1	0	1	0	1	0	0	1	0	0	0	0
		1	0	1	0	1	0	0	0	1	1	0	0
		1	0	1	0	1	0	0	0	1	0	1	1
		1	0	1	0	0	1	1	1	0	0	0	0
		1	0	1	0	0	1	1	0	1	1	0	0
		1	0	1	0	0	1	1	0	1	0	1	1
		1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
		1	0	0	1	1	1	1	0	1	1	0	0
		1	0	0	1	1	1	1	0	1	0	1	1
			1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
			1	1	1	1	1	1	0	1	1	0	0
			1	1	1	1	1	1	0	1	0	1	1
	Codes applying the redundancy constraint rule												
		1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
			1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
364/16	Codes without using the redundancy constraint rule												
	1	0	1	0	1	0	0	1	0	0	0	0	0
	1	0	1	0	1	0	0	0	1	1	0	0	0
	1	0	1	0	1	0	0	0	1	0	1	1	0
	1	0	1	0	1	0	0	0	1	0	1	0	1
	1	0	1	0	0	1	1	1	0	0	0	0	0
	1	0	1	0	0	1	1	0	1	1	0	0	0
	1	0	1	0	0	1	1	0	1	0	1	1	0
	1	0	1	0	0	1	1	0	1	0	1	0	1
	1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
	1	0	0	1	1	1	1	0	1	1	0	0	0
	1	0	0	1	1	1	1	0	1	0	1	1	0
	1	0	0	1	1	1	1	0	1	0	1	0	1
		1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
		1	1	1	1	1	1	0	1	1	0	0	0
		1	1	1	1	1	1	0	1	0	1	1	0
		1	1	1	1	1	1	0	1	0	1	0	1
	Codes applying the redundancy constraint rule												
	1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**

Table 24 – Fibonacci codes for numbers 228, 369

Number / count of codes for a number	Fibonacci-base numeral system												
	233	144	89	55	34	21	13	8	5	3	2	1	1
228/13	Codes without using the redundancy constraint rule												
		1	0	1	0	1	0	1	0	0	0	0	0
		1	0	1	0	1	0	0	1	1	0	0	0
		1	0	1	0	1	0	0	1	0	1	1	0
		1	0	1	0	1	0	0	1	0	1	0	1
		1	0	1	0	0	1	1	1	1	0	0	0
		1	0	1	0	0	1	1	1	0	1	1	0
		1	0	1	0	0	1	1	1	0	1	0	1
		1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	0	0	1	1	1	1	1	0	1	1	0
		1	0	0	1	1	1	1	1	0	1	0	1
			1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
			1	1	1	1	1	1	1	0	1	1	0
			1	1	1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule												
		1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
			1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
369/13	Codes without using the redundancy constraint rule												
	1	0	1	0	1	0	1	0	0	0	0	0	0
	1	0	1	0	1	0	0	1	1	0	0	0	0
	1	0	1	0	1	0	0	1	0	1	1	0	0
	1	0	1	0	1	0	0	1	0	1	0	1	1
	1	0	1	0	0	1	1	1	1	0	0	0	0
	1	0	1	0	0	1	1	1	0	1	1	0	0
	1	0	1	0	0	1	1	1	0	1	0	1	1
	1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
	1	0	0	1	1	1	1	1	0	1	1	0	0
	1	0	0	1	1	1	1	1	0	1	0	1	1
		1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
		1	1	1	1	1	1	1	0	1	1	0	0
		1	1	1	1	1	1	1	0	1	0	1	1
	Codes applying the redundancy constraint rule												
	1	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
		1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**

Table 25 – Fibonacci coding of a number 1359

Number / count of codes for a number	Fibonacci-base numeral system															
	987	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
	Codes without using the redundancy constraint rule															
	1	0	0	1	0	1	0	1	0	1	0	0	1	0	0	0
	1	0	0	1	0	1	0	1	0	1	0	0	0	1	1	0
	1	0	0	1	0	1	0	1	0	1	0	0	0	1	0	1
	1	0	0	1	0	1	0	1	0	0	1	1	1	0	0	0
	1	0	0	1	0	1	0	1	0	0	1	1	0	1	1	0
	1	0	0	1	0	1	0	1	0	0	1	1	0	1	0	1
	1	0	0	1	0	1	0	0	1	1	1	1	1	0	0	0
	1	0	0	1	0	1	0	0	1	1	1	1	0	1	1	0
	1	0	0	1	0	1	0	0	1	1	1	1	0	1	0	1
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	1	0	0	1	1	1	1	1	1	0	1	1	0
	1	0	0	1	0	0	1	1	1	1	1	1	0	1	0	1
	1	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
	1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	0
	1	0	0	0	1	1	1	1	1	1	1	1	0	1	0	1
1359/30		1	1	1	0	1	0	1	0	1	0	0	1	0	0	0
		1	1	1	0	1	0	1	0	1	0	0	0	1	1	0
		1	1	1	0	1	0	1	0	1	0	0	0	1	0	1
		1	1	1	0	1	0	1	0	0	1	1	1	0	0	0
		1	1	1	0	1	0	1	0	0	1	1	0	1	1	0
		1	1	1	0	1	0	1	0	0	1	1	0	1	0	1
		1	1	1	0	1	0	0	1	1	1	1	1	0	0	0
		1	1	1	0	1	0	0	1	1	1	1	0	1	1	0
		1	1	1	0	1	0	0	1	1	1	1	0	1	0	1
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	1	0	0	1	1	1	1	1	1	0	1	1	0
		1	1	1	0	0	1	1	1	1	1	1	0	1	0	1
		1	1	0	1	1	1	1	1	1	1	1	1	0	0	0
		1	1	0	1	1	1	1	1	1	1	1	0	1	1	0
		1	1	0	1	1	1	1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule															
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**

Table 26 – Fibonacci coding of a number 1351

Number / count of codes for a number	Fibonacci-base numeral system															
	987	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
1351/32	Codes without using the redundancy constraint rule															
	1	0	0	1	0	1	0	1	0	0	1	0	0	0	0	0
	1	0	0	1	0	1	0	1	0	0	0	1	1	0	0	0
	1	0	0	1	0	1	0	1	0	0	0	1	0	1	1	0
	1	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1
	1	0	0	1	0	1	0	0	1	1	1	0	0	0	0	0
	1	0	0	1	0	1	0	0	1	1	0	1	1	0	0	0
	1	0	0	1	0	1	0	0	1	1	0	1	0	1	1	0
	1	0	0	1	0	1	0	0	1	1	0	1	0	1	0	1
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
	1	0	0	1	0	0	1	1	1	1	0	1	1	0	0	0
	1	0	0	1	0	0	1	1	1	1	0	1	0	1	1	0
	1	0	0	1	0	0	1	1	1	1	0	1	0	1	0	1
	1	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
	1	0	0	0	1	1	1	1	1	1	0	1	1	0	0	0
	1	0	0	0	1	1	1	1	1	1	0	1	0	1	1	0
	1	0	0	0	1	1	1	1	1	1	0	1	0	1	0	1
		1	1	1	0	1	0	1	0	0	1	0	0	0	0	0
		1	1	1	0	1	0	1	0	0	0	1	1	0	0	0
		1	1	1	0	1	0	1	0	0	0	1	0	1	1	0
		1	1	1	0	1	0	1	0	0	0	1	0	1	0	1
		1	1	1	0	1	0	0	1	1	1	0	0	0	0	0
		1	1	1	0	1	0	0	1	1	0	1	1	0	0	0
		1	1	1	0	1	0	0	1	1	0	1	0	1	1	0
		1	1	1	0	1	0	0	1	1	0	1	0	1	0	1
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
		1	1	1	0	0	1	1	1	1	0	1	1	0	0	0
		1	1	1	0	0	1	1	1	1	0	1	0	1	1	0
		1	1	1	0	0	1	1	1	1	0	1	0	1	0	1
		1	1	0	1	1	1	1	1	1	1	0	0	0	0	0
		1	1	0	1	1	1	1	1	1	0	1	1	0	0	0
		1	1	0	1	1	1	1	1	1	0	1	0	1	1	0
		1	1	0	1	1	1	1	1	1	0	1	0	1	0	1
	Codes applying the redundancy constraint rule															
	1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
		1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**

Table 27 – Fibonacci codes for numbers 22, 35, 57, 92

Number / count of codes for a number	Fibonacci-base numeral system										
	89	55	34	21	13	8	5	3	2	1	1
22/7	Codes without using the redundancy constraint rule										
				1	0	0	0	0	0	1	0
				1	**0**	**0**	**0**	**0**	**0**	**0**	**1**
					1	1	0	0	0	1	0
					1	**1**	**0**	**0**	**0**	**0**	**1**
					1	0	1	1	0	1	0
					1	0	1	1	0	0	1
					1	0	1	0	1	1	1
	Codes applying the redundancy constraint rule										
				1	**0**	**0**	**0**	**0**	**0**	**0**	**1**
					1	**1**	**0**	**0**	**0**	**0**	**1**
35/8	Codes without using the redundancy constraint rule										
			1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
			1	0	0	0	0	0	0	0	1
				1	**1**	**0**	**0**	**0**	**0**	**1**	**0**
				1	1	0	0	0	0	0	1
				1	0	1	1	0	0	1	0
				1	0	1	1	0	0	0	1
				1	0	1	0	1	1	1	0
				1	0	1	0	1	1	0	1
	Codes applying the redundancy constraint rule										
			1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
				1	**1**	**0**	**0**	**0**	**0**	**1**	**0**
57/8	Codes without using the redundancy constraint rule										
		1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
		1	0	0	0	0	0	0	0	1	1
			1	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
			1	1	0	0	0	0	0	1	1
			1	0	1	1	0	0	1	0	0
			1	0	1	1	0	0	0	1	1
			1	0	1	0	1	1	1	0	0
			1	0	1	0	1	1	0	1	1
	Codes applying the redundancy constraint rule										
		1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
			1	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
92/12	Codes without using the redundancy constraint rule										
	1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
	1	0	0	0	0	0	0	0	1	1	0
	1	0	0	0	0	0	0	0	1	0	1
		1	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
		1	1	0	0	0	0	0	1	1	0
		1	1	0	0	0	0	0	1	0	1
		1	0	1	1	0	0	1	0	0	0
		1	0	1	1	0	0	0	1	1	0
		1	0	1	1	0	0	0	1	0	1
		1	0	1	0	1	1	1	0	0	0
		1	0	1	0	1	1	0	1	1	0
		1	0	1	0	1	1	0	1	0	1
	Codes applying the redundancy constraint rule										
	1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**

Table 28 – Fibonacci coding of a number 1736

Number / count of codes for a number	Fibonacci-base numeral system																
	1597	987	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
1736/39	Codes without using the redundancy constraint rule																
	1	0	0	0	0	0	1	0	1	0	1	0	0	1	0	0	0
	1	0	0	0	0	0	1	0	1	0	1	0	0	0	1	1	0
	1	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	1
	1	0	0	0	0	0	1	0	1	0	0	1	1	1	0	0	0
	1	0	0	0	0	0	1	0	1	0	0	1	1	0	1	1	0
	1	0	0	0	0	0	1	0	1	0	0	1	1	0	1	0	1
	1	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0
	1	0	0	0	0	0	1	0	0	1	1	1	1	0	1	1	0
	1	0	0	0	0	0	1	0	0	1	1	1	1	0	1	0	1
	1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	0	0	0	0	1	1	1	1	1	1	0	1	1	0
	1	0	0	0	0	0	0	1	1	1	1	1	1	0	1	0	1
		1	1	0	0	0	1	0	1	0	1	0	0	1	0	0	0
		1	1	0	0	0	1	0	1	0	1	0	0	0	1	1	0
		1	1	0	0	0	1	0	1	0	1	0	0	0	1	0	1
		1	1	0	0	0	1	0	1	0	0	1	1	1	0	0	0
		1	1	0	0	0	1	0	1	0	0	1	1	0	1	1	0
		1	1	0	0	0	1	0	1	0	0	1	1	0	1	0	1
		1	1	0	0	0	1	0	0	1	1	1	1	1	0	0	0
		1	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0
		1	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1
		1	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	0	0	0	0	1	1	1	1	1	1	0	1	1	0
		1	1	0	0	0	0	1	1	1	1	1	1	0	1	0	1
		1	0	1	1	0	1	0	1	0	1	0	0	1	0	0	0
		1	0	1	1	0	1	0	1	0	1	0	0	0	1	1	0
		1	0	1	1	0	1	0	1	0	1	0	0	0	1	0	1
		1	0	1	1	0	1	0	1	0	0	1	1	1	0	0	0
		1	0	1	1	0	1	0	1	0	0	1	1	0	1	1	0
		1	0	1	1	0	1	0	1	0	0	1	1	0	1	0	1
		1	0	1	1	0	1	0	0	1	1	1	1	1	0	0	0
		1	0	1	1	0	1	0	0	1	1	1	1	0	1	1	0
		1	0	1	1	0	1	0	0	1	1	1	1	0	1	0	1
		1	0	1	1	0	0	1	1	1	1	1	1	1	0	0	0
		1	0	1	1	0	0	1	1	1	1	1	1	0	1	1	0
		1	0	1	1	0	0	1	1	1	1	1	1	0	1	0	1
		1	0	1	0	1	1	1	1	1	1	1	1	1	0	0	0
		1	0	1	0	1	1	1	1	1	1	1	1	0	1	1	0
		1	0	1	0	1	1	1	1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule																
	1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**

Table 29 – Fibonacci coding of a number 2346

Number / count of codes for a number	Fibonacci-base numeral system																
	1597	987	610	377	233	144	89	55	34	21	13	8	5	3	2	1	1
	Codes without using the redundancy constraint rule																
	1	0	1	0	0	0	1	0	1	0	1	0	0	1	0	0	0
	1	0	1	0	0	0	1	0	1	0	1	0	0	0	1	1	0
	1	0	1	0	0	0	1	0	1	0	1	0	0	0	1	0	1
	1	0	1	0	0	0	1	0	1	0	0	1	1	1	0	0	0
	1	0	1	0	0	0	1	0	1	0	0	1	1	0	1	1	0
	1	0	1	0	0	0	1	0	1	0	0	1	1	0	1	0	1
	1	0	1	0	0	0	1	0	0	1	1	1	1	1	0	0	0
	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0
	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1
	1	0	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0
	1	0	1	0	0	0	0	1	1	1	1	1	1	0	1	1	0
	1	0	1	0	0	0	0	1	1	1	1	1	1	0	1	0	1
	1	0	0	1	1	0	1	0	1	0	1	0	0	1	0	0	0
	1	0	0	1	1	0	1	0	1	0	1	0	0	0	1	1	0
	1	0	0	1	1	0	1	0	1	0	1	0	0	0	1	0	1
	1	0	0	1	1	0	1	0	1	0	0	1	1	1	0	0	0
	1	0	0	1	1	0	1	0	1	0	0	1	1	0	1	1	0
	1	0	0	1	1	0	1	0	1	0	0	1	1	0	1	0	1
	1	0	0	1	1	0	1	0	0	1	1	1	1	1	0	0	0
	1	0	0	1	1	0	1	0	0	1	1	1	1	0	1	1	0
	1	0	0	1	1	0	1	0	0	1	1	1	1	0	1	0	1
2346/42	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
	1	0	0	1	1	0	0	1	1	1	1	1	1	0	1	1	0
	1	0	0	1	1	0	0	1	1	1	1	1	1	0	1	0	1
	1	0	0	1	0	1	1	1	1	1	1	1	1	1	0	0	0
	1	0	0	1	0	1	1	1	1	1	1	1	1	0	1	1	0
	1	0	0	1	0	1	1	1	1	1	1	1	1	0	1	0	1
		1	1	1	1	0	1	0	1	0	1	0	0	1	0	0	0
		1	1	1	1	0	1	0	1	0	1	0	0	0	1	1	0
		1	1	1	1	0	1	0	1	0	1	0	0	0	1	0	1
		1	1	1	1	0	1	0	1	0	0	1	1	1	0	0	0
		1	1	1	1	0	1	0	1	0	0	1	1	0	1	1	0
		1	1	1	1	0	1	0	1	0	0	1	1	0	1	0	1
		1	1	1	1	0	1	0	0	1	1	1	1	1	0	0	0
		1	1	1	1	0	1	0	0	1	1	1	1	0	1	1	0
		1	1	1	1	0	1	0	0	1	1	1	1	0	1	0	1
		1	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	1	1	1	0	0	1	1	1	1	1	1	0	1	1	0
		1	1	1	1	0	0	1	1	1	1	1	1	0	1	0	1
		1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	0
		1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	0
		1	1	1	0	1	1	1	1	1	1	1	1	0	1	0	1
	Codes applying the redundancy constraint rule																
	1	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
		1	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**