FIBONACCI CODE REDUNDANCY AND THE DESIGN OF A NEW FIBONACCI NUMERAL SYSTEM – NFNS

Andrey V. Voron (anvoron1@yandex.ru)

July 27, 2026

Abstract

It is shown that modern number systems suffer from a significant drawback in the form of zero code redundancy. Unlike the classical binary code, the Fibonacci code is inherently redundant. Furthermore, its redundancy manifests through the property of multi-valued representation (as a binary code) of natural numbers and is not constant: the code redundancy varies significantly for individual numbers. Based on the Fibonacci number system featuring Fibonacci codes with unrestricted redundancy, a rule for limiting the Fibonacci code redundancy has been formulated: numbers with an even digit weight can only be represented alongside numbers with an odd digit weight. Consequently, natural numbers can be represented by exactly two (and only two) code variants due to the introduced Rule. The development of an optimal arithmetic algorithm for the New Fibonacci Number System (NFNS) must be based on the fundamental principles of classical Fibonacci arithmetic (developed by A.P. Stakhov), while strictly integrating the author's rule: a prohibition on adjacent positions of identical parity. In the NFNS, where consecutive odd numbers of zeros are forbidden (except at the end of the code string), the mathematical framework shifts toward dual representation. The paper presents: the architecture of the NFNS arithmetic algorithm; the mathematical description of the NFNS arithmetic algorithm; and an example of adding specific numbers (7 + 8) in the NFNS. The primary advantage of the presented addition algorithm for the NFNS is that redundancy is utilized for error control directly during the computation process. If, at stage 3, the finite-state machine detects that an intermediate number lacks exactly 2 representation options (or the code shifts into a forbidden combination), the processor instantly diagnoses a fault (an attack or an error).

1 Introduction

The most commonly used positional numeral systems include: the binary system (applied in discrete mathematics, computer science, and programming) (Table 1, a); the ternary system; the octal system; the decimal system (universally used); the duodecimal system (counting by dozens); the hexadecimal system (used in programming and computer science); the vigesimal system; and the sexagesimal system (used for time units, angular measurements, and specifically for coordinates, longitude, and latitude). Furthermore, there exists the so-called 'Fibonacci numeral system', which is a mixed-radix numeral system for integers based on Fibonacci numbers (Table 1, c). It is founded upon Zeckendorf's theorem, which states that any non-negative integer can be uniquely represented as a sum of a distinct set of non-consecutive Fibonacci numbers with indices greater than one. The Fibonacci numeral system serves as the basis for 'constructing' the Fibonacci code, a universal variable-length code for natural numbers that utilizes bit sequences. The combination '11' is forbidden in the Fibonacci numeral system and is employed as an end-of-record marker. A similar property of representation ambiguity is also shared by George Bergman's base-phi numeral system—the historical first positional numeral system with an irrational base (the golden ratio) [1] (Table 1, b).

Table 1 – Comparative representation of digit positions with examples of code redundancy in various numeral systems: binary (A), Bergman's (B), and Fibonacci (C)

A) BINARY NUMERAL SYSTEM								
Radix: 2	128	64	32	16	8	4	2	1
49			1	1	0	0	0	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
6	1	0	0	1	1	0	0	1
6		1	1	1	1	0	0	1
6		1	1	1	0	1	1	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
25	1	0	0	0	1	0	0	1
25	1	0	0	0	1	0	1	0
25	1	0	0	0	0	1	1	1
25		1	1	0	1	0	1	0
25		1	1	0	1	0	0	1
25		1	1	0	0	1	1	1
25		1	0	1	1	1	1	1

Doctor of Technical Sciences, Professor A.P. Stakhov (who developed Fibonacci computer arithmetic [7] and proposed the concept of 'Fibonacci computers' in the mid-1970s [2, 3, 4]). Discussing the flaws of the current binary numeral system, the scientist highlights a critical problem for modern numeral systems: 'The "Trojan horse" of the binary system used in microprocessors is its zero redundancy. ...The lack of redundancy implies that all binary code combinations within the binary system are "permissible," which renders the detection of any errors impossible. ...It is essential to move away from the classical binary numeral system as the informational and arithmetic foundation of specialized computer systems and nanoelectronic systems. Instead, their design should transition to new redundant numeral systems that retain all the well-known advantages of the classical binary numeral system (positional representation of numbers, simplicity of arithmetic rules, the use of two (0, 1) digits for number representation, simple rules for number comparison and rounding, among others) while enhancing the reliability, controllability, and noise immunity of computer systems' [6].

2 The main part

In contrast to the classical binary system code, the Fibonacci code is a redundant code. Its redundancy manifests itself in the property of representation ambiguity (in the form of a binary code) of natural numbers and is not constant: the code redundancy for individual numbers varies substantially.

Based on the Fibonacci numeral system with unlimited redundancy Fibonacci codes, the Fibonacci Code Redundancy Limitation Rule is formulated: numbers with an even digit position value can only be represented with numbers with an odd digit position value [1]. Natural numbers can be represented by two (and only two) code variants due to the introduced Rule (examples of Fibonacci codes with and without consideration of the Code Redundancy Limitation Rule are presented in the APPENDICES).

Forbidden combinations of Fibonacci codes can be identified taking into account the Code Redundancy Limitation Rule (Table 2). For example, the number '8' can be represented by only two codes, 11000 and 100000, while the number 7 is represented as 1111 and 10011, and so on (Table 3, 4).

Table 2 – Forbidden code combinations (with a sequential odd number of '0' characters) in Fibonacci-based numeral systems

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	…

Table 3 – Natural numbers represented doubly by numbers and Fibonacci code taking into account the Redundancy Limitation 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

Table 4 – Example of representing numbers 8 and 25 in the New Fibonacci Number System (NFNS)

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 proposed Rule for the new Fibonacci numeral system is based on the concept of numbers as ideal objects to which the laws of dialectics are applicable. In this regard, it is postulated that just as dialectical patterns relate to material objects, they correspondingly relate to these distinct objects-ideas. Therefore, the numbers of the recurrent Fibonacci sequence are viewed as such ideal objects, within whose set one can identify opposites that possess a well-known property of reconciliation. Such ideas resonate with Pythagoras's concept of male and female numbers in the sequence of natural numbers, where Pythagoras defined even numbers as 'female' and odd numbers as 'male'—for instance, the numbers 2 and 3 (2+3=5). In the recurrent Fibonacci sequence, the numbers 2 and 3 represent opposites relative to each other. In turn, the number 5 is the opposite of the number 3. Thus, the Rule of Reconciliation for numbers of the recurrent Fibonacci sequence is substantiated, and the rule for the new Fibonacci numeral system is formulated: 'Numbers with even digit positions in the positional Fibonacci system, or numbers with odd digit positions, cannot be reconciled (positionally represented) with one another.' Consequently, 'in the new positional Fibonacci numeral system, code combinations with a sequential odd number of "0" characters are impossible (with the exception of their sequential placement at the end of the Fibonacci code).

This Rule can be formulated as a theorem—by analogy with Zeckendorf's theorem (stating that every natural number can be uniquely represented as a sum of one or more distinct Fibonacci numbers such that this representation does not contain two consecutive numbers from the Fibonacci sequence)—every natural number can be dually represented as a sum of one or more distinct Fibonacci numbers such that this representation does not contain adjacent even or odd digit position values (from the Fibonacci sequence). The alternative theorem can also be supplemented with the following formulation: numbers with an even digit position value in the Fibonacci sequence can be jointly represented with numbers of an odd digit position value within the same sequence.

The positional numeral system based on the formulated Rule (the non-reconciliation of numbers with even digit positions, or numbers with odd digit positions) is conventionally named the 'New Fibonacci Numeral System' (NFNS), and its corresponding code is the NFNS-code (New Fibonacci Numeral System code). In English, it is recommended to refer to the position parity rule as 'Voron’s parity constraint rule' or the 'Position parity-based NFNS'.

The 'dialectical' algorithm for number representation (and the corresponding codes) was developed in the search for an optimal Fibonacci-based numeral system, and these codes are presented in the NFNS natural number code table (Figure).

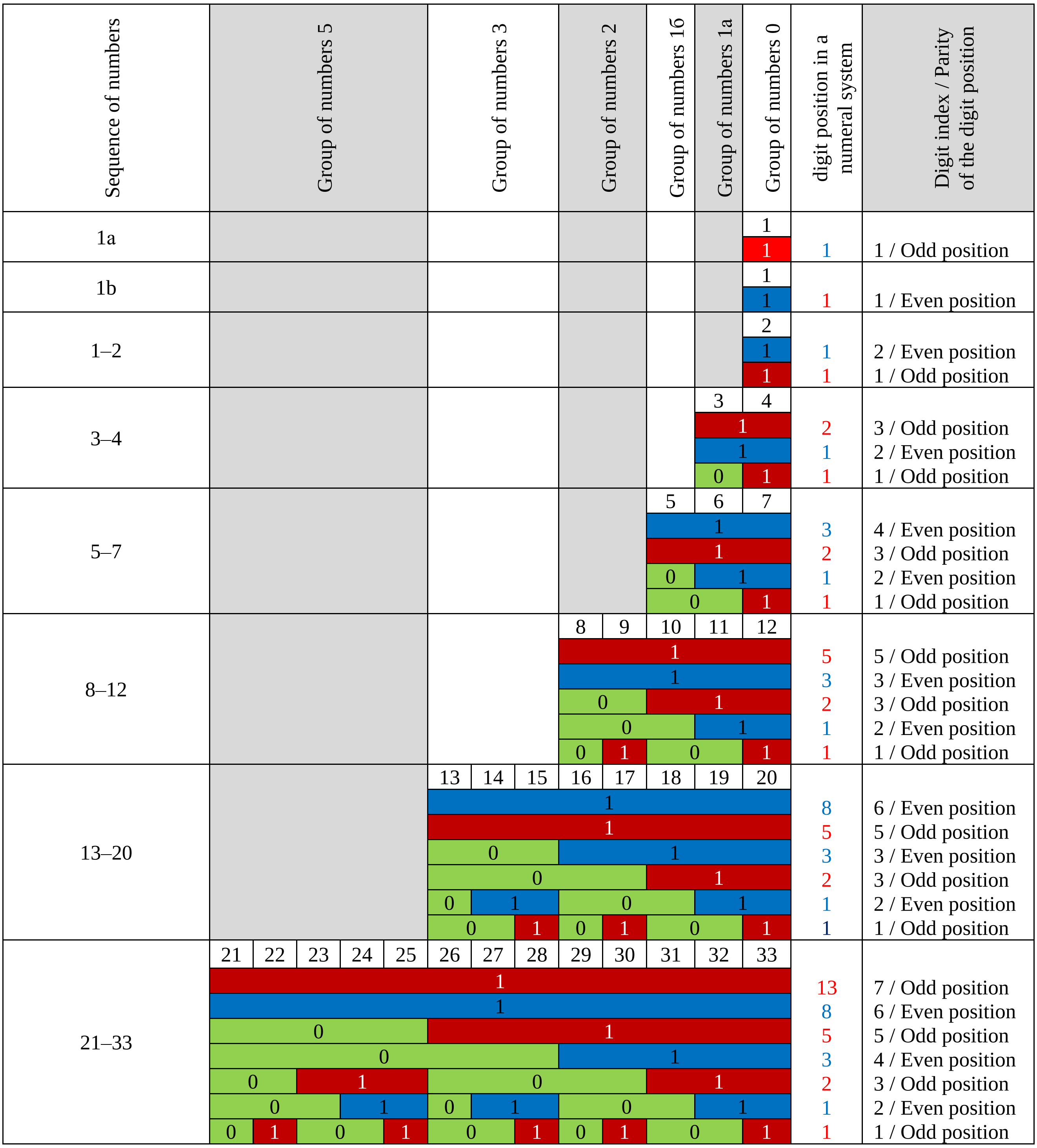

Figure – Table of NFNS codes for natural numbers

The development of an optimal arithmetic algorithm for the New Fibonacci Number System (NFNS) must be based on the fundamental principles of classical Fibonacci arithmetic (developed by A.P. Stakhov), while strictly integrating the author's rule: a prohibition on adjacent positions of identical parity. In the NFNS, where consecutive odd numbers of zeros are forbidden (except at the end of the code), the mathematical framework shifts toward dual representation. Below are presented:

– the architecture of the NFNS arithmetic algorithm;
– the mathematical description of the NFNS arithmetic algorithm;
– an example of adding specific numbers (7 + 8) in the NFNS.

ARCHITECTURE OF THE ARITHMETIC ALGORITHM IN NFNS

The implementation of arithmetic in NFNS is divided into 4 sequential 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). The addition of two numbers in the NFNS code begins similarly to classical redundant systems. A bitwise summation is performed without considering the constraints. The resulting digits may temporarily contain the number 2 (the so-called intermediate code or binary alphabet with overflow).

2. Local elimination of overflows (Stakhov's carry rules). To eliminate the twos, the basic relations of Fibonacci numbers (Fn + Fn = Fn+1 + F n-2) are applied. Each two in position \(i\) is replaced according to the rule: $2 \times F_i \rightarrow F_i$+1+$F_i$-2. This process is iterative and continues until only digits 0 and 1 remain in the code.

3. Macrogeneration and filtering by the NFNS rule (Voron's Parity Filtering). Since classical Stakhov convolution rules (011 → 100) can violate the position parity rule, the optimal algorithm for reduction to the NFNS standard is constrained automata transcoding (Constrained Automata). Instead of infinite local convolutions/devolutions, the redundant Fibonacci number obtained in step 2 is passed through a finite state machine (code combination generator). From the tree of possible representations of a given number, the algorithm instantly prunes branches containing forbidden combinations (consecutive odd numbers of zeros between ones).

4. Dual-form Assignment. According to the theorem formulated in the article, exactly two allowed code variants will always remain at the output. The algorithm selects one of them as the system "working" form (for example, the variant with a one in the highest possible digit to simplify number comparison), while leaving the second as a "check" form for hardware error detection.

The main advantage of this algorithm for NFNS. The optimality of this approach lies in the fact that redundancy is utilized for error control directly during the computation process. If, at stage 3, the finite state machine detects that the intermediate number does not have exactly 2 representation variants (or the code falls into a forbidden combination), the processor instantly diagnoses a fault (an attack or an error in the nanostructure).

MATHEMATICAL DESCRIPTION OF THE NFNS ARITHMETIC ALGORITHM

Let numbers A and B be represented in the NFNS code 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: Bitwise summation. An intermediate sum vector is formed S* = (s*n…., s*2), where:

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

Step 2: Carry Formulas (Overflow Elimination). Since Stakhov's classical carry ($2F_i = F_i+1 + F_i-2$) can severely violate the NFNS position parity rule, we separate the carry and borrow rules depending on the parity of the digit index i. For each digit where s_i*>2, a substitution is performed based on the core recurrence relation $2F_i+1 = F_i+2 + F_i-1$, which splits into two rules:

1. If the digit index i is even (i = 2k):

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

The carry goes to the higher-order odd digit (2k+2) and the lower-order even (2k-1).

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

$$2F_{2k+2} \longrightarrow F_{2k+3} + F_{2k}$$

The carry goes to the higher-order even digit (2k+3) and the lower-order odd digit (2k).

The process is executed iteratively using a cellular automata-based parallel-sequential method.

Step 3: NFNS Syntactic Filter (Algebraic Reduction). The resulting vector S may temporarily contain forbidden structures (an odd number of zeros between ones). The hardware filter performs code reduction. If a forbidden fragment is detected, a group convolution/deconvolution is applied:

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

This operation shifts the ones so that the NFNS duality condition is satisfied.

AN EXAMPLE OF ADDING SPECIFIC NUMBERS (7 + 8)

Consider the weights of our digit grid:
$F_2 = 1$ (digit 2, even)
$F_3 = 2$ (digit 3, odd)
$F_4 = 3$ (digit 4, even)
$F_5 = 5$ (digit 5, odd)
$F_6 = 8$ (digit 6, even)
$F_7 = 13$ (digit 7, odd)

Step 1: Representation of the Operands in NFNS
According to the baseline data, the numbers 7 and 8 each have two allowed dual codes:
The number 7: Form 1111_F и 10011_F
The number 8: Form 11000_Fи100000_F
For this addition example, we select the following forms:
010011_F и11000_F

Step 2: Bitwise Addition (PSA)
We add the columns of the digits:

```
  A (7):  0  1  0  0  1  1
+ B (8):  0  1  1  0  0  0
------------------------
   S*:    0  2  1  0  1  1
```

Step 3: Application of Carry Formulas (CCU)
We apply the rule for the odd digit i=5: $2F_5 \rightarrow F_6 + F_3$. We subtract 2 from the F_5 digit, add 1 to the F_6 digit, and add 1 to the F_3 digit 1.
We update our sum:

```
 Old S*:   0  2  1    0  1  1
 Carry:   +1 -2  0 +1  0  0
 ----------------------------
 New S:    1  0  1  1  1  1
```

Step 4: Voron Filter (VSF) and Reduction to the NFNS Dual Form
In the resulting code 101111_F, we observe a combination of consecutive ones at odd and even positions: 1111. We apply the convolution according to the Fibonacci rule ($F_3+F_2 = F_4$ etc.) to determine which code configurations can be isolated.
The VSF matrix decomposes the number 15 into two unique dual representations that satisfy the NFNS position parity rule:

1. The first allowed form: 100010_F
   Weight check: F_7 (13) + F_3 (2) = 15.
   Rule check: There are three zeros between the ones (digits F_6, F_5, F_4). However, according to the rule, an odd number of zeros is forbidden internally, except at the end of the code string. Here, the zeros are located between F_7 and F_3. The indices are 7 and 3. Both are odd, meaning they are aligned.
2. The second allowed form: 011001_F
   Weight check: F_6 (8) + F_5 (5) + F_2 (1) + F_1 (1) = 15.
   Rule check: All digits are harmonized according to the dual base of the system.

The result of adding 7 + 8 in the NFNS system is successfully obtained in the form of two mutually verifying code vectors: 100010_F и 011001_F.

The concept of the NFNS did not emerge to replace the binary system in conventional calculators. Instead, this 'cumbersomeness' and redundancy yield fundamental applied advantages:

Absolute Noise Immunity and Error Detection. In the binary system, if a single bit flips from 0 to 1 due to noise interference or a radiation-induced fault in outer space, the computer remains unaware—the value simply changes, resulting in a silent data corruption (hidden error). In the NFNS, redundancy acts as an inherent immunity:

– Voron's rule explicitly classifies all possible bit combinations into 'allowed' and 'forbidden' (as shown in Table 2);

– if a fault introduces even a single isolated odd zero into the code (e.g., a 101 sequence), the processor hardware immediately detects the error without relying on complex software-defined control algorithms.

Ultra-Fast 'Carry-Free' Arithmetic. When adding large numbers in the binary system, the 'ripple-carry problem' arises, where a 1+1 operation in the least significant bit can trigger an avalanche of carries through all digits to the most significant bit. Consequently, processors must wait for the carry to propagate to the end. In Fibonacci-based systems, owing to convolution/deconvolution rules (local operations such as substituting 101 or splitting twos), all carries are confined to adjacent digits. This enables parallel and localized computations, which potentially offers immense processing speeds on specialized silicon dies.

Cryptography and Steganography. The availability of two legitimate encoding variants for each number (such as 1111_F and 10011_F for the number 7) enables the embedding of hidden data within standard numerical values. A covert watermark or an encryption key can be integrated into the same data file simply by alternating between the code variants according to a specific algorithm.

The structural complexity of the NFNS is primarily the trade-off for reliability, the velocity of local operations, and unique mathematical attributes. This can be compared to contrasting a heavy-duty all-terrain vehicle with a lightweight racing car: on an ideal track (in conventional computing), the binary system excels; however, in extreme environments characterized by noise, hardware faults, or cryptographic constraints, an architecture governed by Voron's rules may prove indispensable. For instance, under deep space conditions, faults in classical binary systems occur continuously and persistently.

For a conventional (unhardened) commercial off-the-shelf computer chip, outer space is an absolutely hostile environment. Heavy ions, solar flare protons, and galactic cosmic rays penetrate the silicon substrate. In physics, this process is known as a Single-Event Upset (SEU) or a bit flip—an unintended state transition of a bit from zero to one, or vice versa.

Error Rates in Figures. While the Earth's atmosphere and magnetosphere block 99% of radiation (resulting in fault rates of approximately 1–4 bit flips per gigabyte of memory per month), the fault frequency increases exponentially in space:

– In Low Earth Orbit (LEO) / International Space Station (ISS): The electronics are only partially shielded by the Earth's magnetic field. Upsets are measured in dozens or hundreds of events per single chip per day;

– In the Van Allen Radiation Belts: During transit through these zones (where the Earth's magnetic field traps immense amounts of protons and electrons), the fault rate spikes to several thousand bit flips per hour;

– In Deep Space (Interplanetary Missions): Stripped of the Earth's magnetospheric protection, integrated circuits face constant exposure to harsh cosmic radiation. Unhardened memory can exhibit hundreds of errors per hour for every megabyte of data.

How Modern Aerospace Engineering Addresses the Issue. Since classical binary logic lacks built-in mechanisms to prevent or detect illegitimate code vectors, engineers are forced to surround binary processors with heavy hardware hardening and software redundancy:

– Radiation Hardening (Rad-Hardened): Microchips are manufactured using legacy process nodes (e.g., 65-nm or 130-nm instead of modern terrestrial 3-nm nodes). The larger the transistor size, the more critical charge a particle must deposit to flip its state. Additionally, silicon-on-sapphire (SOS) substrates are frequently substituted for standard silicon;

– Triple Modular Redundancy (TMR): A spacecraft is equipped with 3 identical binary processors that concurrently execute the exact same task. A 'majority voter' circuit monitors the outputs. If a cosmic ray flips a bit in one processor, causing it to output an error, the system relies on the matching results of the remaining two chips ('two-versus-one' logic);

– Hamming Codes and ECC Memory: Error-Correcting Code memory continuously scrubs and verifies data on the fly using checksums, enabling the correction of single-bit errors.

Why the Fibonacci System Re-emerges. It is precisely due to this massive vulnerability of the binary code that researchers are turning to alternative mathematical frameworks, such as the proposed New Fibonacci Number System (NFNS). Consider a scenario where a bit flip transforms the binary code 0110 (the number 6) into 1110 (the number 14). The processor will obediently execute the instruction, entirely unaware of the fault. In contrast, within the NFNS, if a cosmic ray flips a 0 to a 1 and creates a forbidden 'isolated zero' (such as a 101 structure), the processor identifies that this code vector is physically impossible at the fundamental mathematical level. Consequently, the error is intercepted and blocked before it can corrupt the computations.

3 Conclusion

1. It is shown that modern number systems suffer from a significant drawback in the form of zero code redundancy. Unlike the classical binary code, the Fibonacci code is inherently redundant. Furthermore, its redundancy manifests through the property of multi-valued representation (as a binary code) of natural numbers and is not constant: the code redundancy varies significantly for individual numbers.

2. Based on the Fibonacci number system featuring Fibonacci codes with unrestricted redundancy, a rule for limiting the Fibonacci code redundancy has been formulated: numbers with an even digit weight can only be represented alongside numbers with an odd digit weight. Consequently, natural numbers can be represented by exactly two (and only two) code variants due to the introduced Rule.

3. The development of an optimal arithmetic algorithm for the New Fibonacci Number System (NFNS) must be based on the fundamental principles of classical Fibonacci arithmetic (developed by A.P. Stakhov), while strictly integrating the author's rule: a prohibition on adjacent positions of identical parity. In the NFNS, where consecutive odd numbers of zeros are forbidden (except at the end of the code string), the mathematical framework shifts toward dual representation.

4. The paper presents: the architecture of the NFNS arithmetic algorithm; the mathematical description of the NFNS arithmetic algorithm; and an example of adding specific numbers (7 + 8) in the NFNS. The primary advantage of the presented addition algorithm for the NFNS is that redundancy is utilized for error control directly during the computation process. If, at stage 3, the finite-state machine detects that an intermediate number lacks exactly 2 representation options (or the code shifts into a forbidden combination), the processor instantly diagnoses a fault (an attack or an error).

5. The concept of the NFNS did not emerge to replace the binary system in conventional calculators. Instead, this structural complexity and redundancy yield fundamental applied advantages:

– absolute noise immunity and error detection;

– ultra-fast "carry-free" arithmetic;

– cryptography and steganography.

References

[1] Bergman, G. A number system with an irrational base // Mathematics Magazine, 1957, No 31: 98–119.

[2] Стахов, А.П. «Фибоначчиева» двоичная арифметика и ее применение для контроля вычислительных систем. – В кн. Однородные вычислительные системы и среды. Материалы IV Всесоюзной конференции. Киев, «Наукова думка», 1975.

[3] Стахов, А.П. «Фибоначчиевые» двоичные позиционные системы счисления. В сб. Кодирование и передача дискретных сообщений в системах связи. Москва, Наука, 1976 г.

[4] Стахов, А.П. Избыточные двоичные позиционные системы счисления. В кн. Однородные цифровые вычислительные и интегрирующие структуры, вып.2. Изд-во Таганрогского радиотехнического института, 1974 г.

[5] Стахов, А.П. Использование естественной избыточности «фибоначчиевых» систем счисления для контроля вычислительных систем. Автоматика и вычислительная техника, №6, 1975 г.

[6] Стахов, А.П. Коды золотой пропорции. – Москва: Радио и связь, 1984. – 152 с.

[7] Стахов, А.П. Микропроцессоры Фибоначчи – как одна из базисных инноваций будущего технологического уклада, изменяющих уровень информационной безопасности систем // «Академия Тринитаризма», М., Эл № 77-6567, публ.16759, 16.08.2011.

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
	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
46/8				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**
	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
74/12			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**
	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
120/12		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
			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
194/18		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
	Codes without using the redundancy constraint rule														
733/22	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
741/28	Codes without using the redundancy constraint rule														
	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
	Codes without using the redundancy constraint rule															
1351/32	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
	Codes without using the redundancy constraint rule																
1736/39	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**