ENSIN@ UFMS 2021
ISSN 2525-7056

A MATEMÁTICA DO QR CODE

THE QR CODE'S MATH

Alberto Renan Dias da Silva[1]

Silas Fantin[2]

RESUMO: O artigo é um recorte da dissertação de conclusão do Mestrado Profmat desenvolvido na Universidade Federal do Rio de Janeiro (UNIRIO), apresentada em Agosto de 2021 (SILVA, 2021). O objetivo deste trabalho é mostrar que códigos podem ser utilizados nas aulas de matemática básica com a finalidade de manter os alunos interessados na disciplina e desenvolvermos alguns conteúdos de forma concreta, com a finalidade de trazer a realidade para dentro da sala de aula. Mostraremos aqui as características dos QR codes e como eles podem ser relacionados com conteúdos matemáticos, algumas maneiras de criarmos e lermos um QR code. Apresentaremos também atividades de aplicação em sala de aula que envolvam o tema deste trabalho.

PALAVRAS-CHAVE: Tecnologia. QR Code. Atividades.

ABSTRACT: The article is an excerpt from the Profmat Master's thesis concluded at the Federal University of Rio de Janeiro (UNIRIO), presented in August 2021. The objective of this work is to show which codes can be used in basic mathematics classes in order to maintain students interested in the discipline and develop some content in a concrete way, with the purpose of bringing reality into the classroom. Here we will show the characteristics of QR codes and how they can be related to mathematical content, some ways we can create and read a QR code. We will also present application activities in the classroom that involve the theme of this work.

KEYWORDS: Technology. QR Code. Activities.

Introdução

O Código de Barras foi criado há mais de 70 anos onde a necessidade de automação na hora da venda de produtos em supermercados fez com que o CEO de uma grande rede procurasse um aluno do Instituto de Tecnologia Drexel (atual Universidade Drexel), na Filadélfia em 1948. A ideia era ter uma identificação do produto na hora em que o mesmo passasse no caixa, facilitando e agilizando as vendas. Segundo o site GS1BR, os alunos Bernard Silver e Joseph Woodland trabalharam juntos e desenvolveram vários tipos de códigos até chegarem no código de barras. Mas apenas vinte anos depois a evolução da tecnologia permitiu que fosse desenvolvido um leitor prático para a aplicação dos códigos de barras em supermercados. Já o QR code foi criado em 1994 pela empresa Denso Wave, uma subsidiária da Toyota, que produzia peças de automóveis, com o intuito de ter um código à mão que pudesse conter as características das peças produzidas.

[1] Universidade Federal do Estado do Rio de Janeiro. E-mail: alberto.renan.dias@gmail.com
https://orcid.org/0000-0002-4889-9532
[2] Universidade Federal do Estado do Rio de Janeiro. E-mail: silas.fantin@uniriotec.br
https://orcid.org/0000-0002-2183-6806

● Informações completas da obra no final do artigo

Uma grande diferença entre o código de barras e o QR Code é que o segundo código possui a licença livre tal qual qualquer empresa pode utilizá-lo livremente, ao contrário do código de barras que é gerenciado por uma empresa no mundo todo.

Este trabalho tem como objetivo apresentar uma proposta de abordagem sobre o QR Code para professores da Educação Básica, com o propósito de inseri-lo nas aulas de matemática trazendo seu contexto histórico e cultural como forma de um agente lúdico para melhor desenvolvimento dos conteúdos e mostraremos algumas aplicações no dia a dia. "Utilizar dispositivos móveis (celular) e o aplicativo QR Code como recurso pedagógico para potencializar o ensino e aprendizagem da matemática" (PINTO; FELCHER; FERREIRA, 2016, p. 5).

No decorrer do texto será necessária a compreensão sobre notação binária, como, por exemplo, o número 5 em binário é representado por $(101)_2$, e no desenvolvimento sobre o QR code mostraremos como reconhecer alguns padrões associados ao código e suas funções, exporemos também algumas maneiras de ler um QR code com o auxílio de um aparelho celular.

A ideia é criar uma linha de conhecimento acerca do QR Code para embasar o foco deste trabalho que será um conjunto de atividades que poderão ser desenvolvidas em sala de aula do Ensino Básico, que, segundo Silva e Bezerra (2016), pode diminuir a dificuldade dos alunos com relação a matérias ministradas pelos educadores.

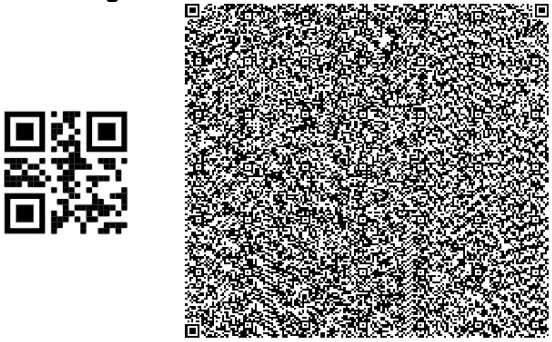
Como funciona o QR Code

Neste trabalho utilizaremos como modelo de estudo o QR Code Model 2 Version 1, embora haja outras variações do QR code, como o iQR Code e o Micro QR Code. Neste capítulo vamos mostrar suas subdivisões, quais os seus elementos e funções. Pode parecer à primeira vista ser de difícil compreensão a mecânica envolvida na sua construção, mas mostraremos passo a passo cada detalhe envolvido neste código mensageiro.

Quantidade de pixels

O QR Code é um quadrado subdividido em pequenos quadradinhos chamados pixels, que vão nos dar uma noção da capacidade de armazenagem de informação que a imagem terá. Um código é separado em versões e possui de 21×21 = 441 pixels na versão 1 até 177×177 = 31329 pixels na versão 40 (versão máxima).

Figura 2: Versão 1 e Versão 40.



Fonte: autores.

Acima podemos ver a diferença de tamanho entre a menor e a maior versão do código QR e abaixo temos uma tabela com todas as versões do QR e seus respectivos pixels (linha x coluna). Note que a quantidade de pixels por versão cresce de quatro em quatro.

Tabela 1: quantidade de pixels.

Versão	Pixels	Versão	Pixels	Versão	Pixels	Versão	Pixels
1	21x21	11	61x61	21	101x101	31	141x141
2	25x25	12	65x65	22	105x105	32	145x145
3	29x29	13	69x69	23	109x109	33	149x149
4	33x33	14	73x73	24	113x113	34	153x153
5	37x37	15	77x77	25	117x117	35	157x157
6	41x41	16	81x81	26	121x121	36	161x161
7	45x45	17	85x85	27	125x125	37	165x165
8	49x49	18	89x89	28	129x129	38	169x169
9	53x53	19	93x93	29	133x133	39	173x173
10	57x57	20	97x97	30	137x137	40	177x177

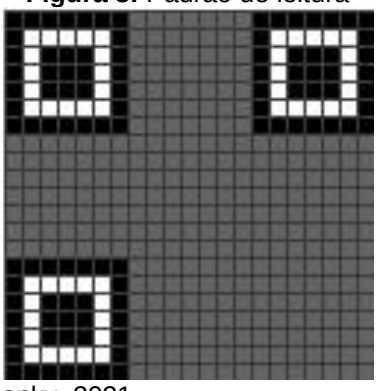
Fonte: autores.

Neste trabalho utilizaremos a versão 21x21 do QR Code como base para a maioria dos exemplos, já que é a menor delas e mais simples e com isso facilitará a explicação e o entendimento dos leitores e que ao falarmos de um determinado bit (10, 16), por exemplo, estamos falando sobre o bit da linha 10 e coluna 16.

Padrão de localização

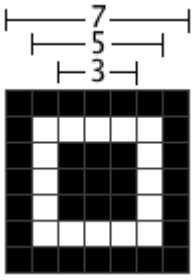
Uma forma de identificar um QR Code é olhando os padrões que ficam em três dos quatro cantos do quadrado, esses padrões distintivos possuem o mesmo tamanho e aparecem em qualquer tipo de QR Code para detectar a posição de rotação da imagem.

Figura 3: Padrão de leitura



Fonte: Thonky, 2021.

Figura 4: Pixels do padrão



Fonte: Thonky, 2021

Esta é a posição padrão do QR Code, um padrão no vértice esquerdo inferior, um padrão no vértice esquerdo superior e um padrão no vértice direito superior, com isso, o leitor de código, por exemplo a câmera do celular, consegue fazer a leitura mesmo se a

ISSN 2525-7056

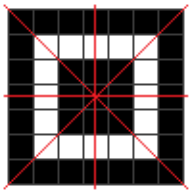
imagem estiver rotacionada. Outra função do padrão é determinar o tamanho do pixel. Cada padrão é formado por 7×7 pixels, ou seja, são 49 pixels divididos em 3 grupos.

- Grupo preto externo com 24 pixels;
- Grupo branco com 16 pixels;
- Grupo preto interno com 9 pixels.

Segundo ISO/IEC 18004, o padrão de localização foi feito para ser um padrão que provavelmente não aparecerá nas outras seções do QR Code. As larguras dos módulos do padrão do localizador têm uma proporção de 1: 1: 3: 1: 1. Os leitores de QR Code podem pesquisar essa proporção de módulos claros para escuros para detectar os padrões do localizador e orientar corretamente o QR Code para decodificação.

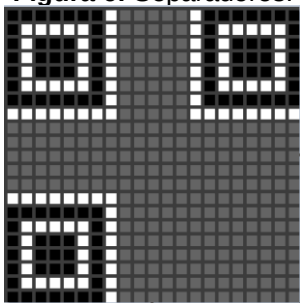
A imagem na Figura 5, abaixo mostra a proporção citada, com as linhas vermelhas mostrando a seguinte sequência de pixels em qualquer direção: 1 preto - 1 branco - 3 pretos - 1 branco - 1 preto.

Figura 5: Padrão 1:1:3:1:1.



Fonte: autores.

Figura 6: Separadores.



Fonte: Thonky, 2021.

Separadores

Os chamados separadores (Figura 6) são linhas brancas que são colocadas ao lado dos padrões distintivos para separá-los do restante do QR Code.

A função deste elemento do código é separar o padrão do restante do código, dando destaque aos três padrões citados na seção anterior e evitar erros de leitura. São compostas de três partes em formatos de L e cada parte é formada por 15 pixels brancos.

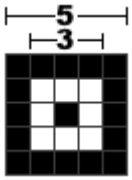
Padrão de alinhamento

QR Codes cuja imagem é muito grande precisam de um padrão de alinhamento para orientar o programa a fazer a leitura. Esse padrão é composto por um quadrado de tamanho $5 \times 5 = 25$ pixels também subdividido em três grupos.

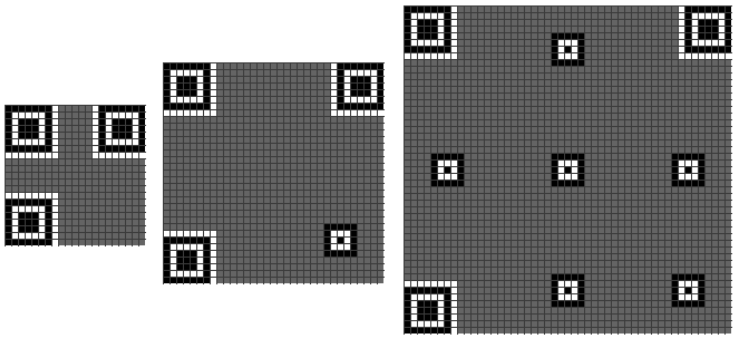
- Grupo preto externo com 16 pixels;
- Grupo branco com 8 pixels;
- Grupo preto interno com 1 pixel.

Figura 7: Estrutura de alinhamento.

Figura 8: Estrutura de alinhamento no QR Code.



Fonte: Thonky, 2021.



Fonte: autores.

Na versão 1 (21×21), que é o foco deste texto, esse elemento da estrutura do código não aparece, conforme abaixo.

Na Figura 8, temos três versões do QR Code que são versão 1 (21x21), versão 4 (33x33) e versão 8 (49x49), note que quanto maior for a versão do QR Code maior será a quantidade de estruturas de alinhamento para orientar o escâner.

Este padrão possui uma localização específica dentro de cada QR Code dependendo de sua versão, vide tabela a seguir.

Tabela 2: coordenadas dos padrões de alinhamento.

Versão	Linha e coluna				Versão	Linha e coluna						
QR Versão 2	6	18			QR Versão 21	6	28	50	72	94		
QR Versão 3	6	22			QR Versão 22	6	26	50	74	98		
QR Versão 4	6	26			QR Versão 23	6	30	54	78	102		
QR Versão 5	6	30			QR Versão 24	6	28	54	80	106		
QR Versão 6	6	34			QR Versão 25	6	32	58	84	110		
QR Versão 7	6	22	38		QR Versão 26	6	30	58	86	114		
QR Versão 8	6	24	42		QR Versão 27	6	34	62	90	118		
QR Versão 9	6	26	46		QR Versão 28	6	26	50	74	98	122	
QR Versão 10	6	28	50		QR Versão 29	6	30	54	78	102	126	
QR Versão 11	6	30	54		QR Versão 30	6	26	52	78	104	130	
QR Versão 12	6	32	58		QR Versão 31	6	30	56	82	108	134	
QR Versão 13	6	34	62		QR Versão 32	6	34	60	86	112	138	
QR Versão 14	6	26	46	66	QR Versão 33	6	30	58	86	114	142	
QR Versão 15	6	26	48	70	QR Versão 34	6	34	62	90	118	146	
QR Versão 16	6	26	50	74	QR Versão 35	6	30	54	78	102	126	150
QR Versão 17	6	30	54	78	QR Versão 36	6	24	50	76	102	128	154
QR Versão 18	6	30	56	82	QR Versão 37	6	28	54	80	106	132	158
QR Versão 19	6	30	58	86	QR Versão 38	6	32	58	84	110	136	162
QR Versão 20	6	34	62	90	QR Versão 39	6	26	54	82	110	138	166
					QR Versão 40	6	30	58	86	114	142	170

Fonte: autores.

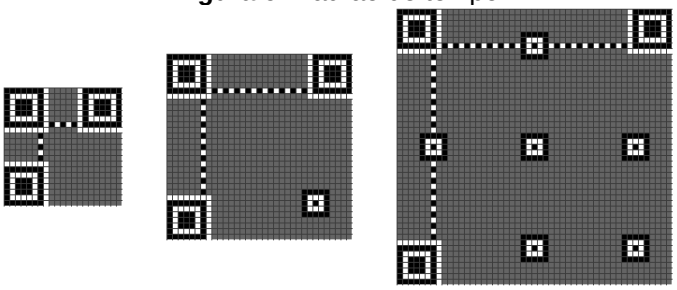
Os números no lado direito desta tabela devem ser usados como ambas as coordenadas de linha e coluna. Por exemplo, a versão 2 tem os números 6 e 18. Isso significa que os centros dos padrões de alinhamento devem ser colocados em (6, 6), (6, 18), (18, 6) e (18, 18). A versão 1 não possui este elemento, por isso a sua ausência na

tabela, e a versão 40 possui 46 padrões de alinhamento pois sempre são removidos os padrões que se sobrepuserem aos padrões de localização ou separadores.

Padrões de tempo

Esta parte do código é formada por duas linhas que são feitas de pixels de cores alternadas, sempre começando e terminando por um pixel preto.

Figura 9: Padrão de tempo.



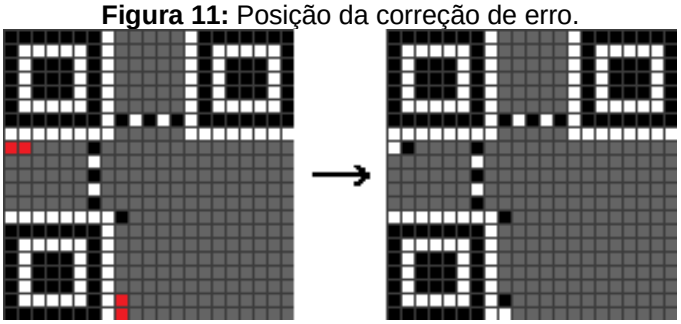
Fonte: Thonky, 2021.

Esta estrutura sempre é colocada na sexta coluna da esquerda para a direita e na sexta linha de cima para baixo. Isso permite ao programa leitor confirmar a versão do código assim como o tempo de cada pixel ao longo da imagem além de auxiliar na percepção do posicionamento das linhas e colunas.

Correção de erro

O próximo item da estrutura da imagem é onde fica armazenado o nível de correção de erro que será adotado na construção dos dados. Cada nível de correção de erro reserva uma parte do código para repetição dos dados da imagem, a fim de corrigir qualquer falha de leitura da mensagem original ou parte do código que esteja faltando ou danificada. Este elemento ocupa dois bits da mensagem e fica posicionado na parte vermelha da imagem, como mostrando na figura 11, a seguir.

Figura 11: Posição da correção de erro.



Fonte: autores.

Acima, vemos a posição onde esses pixels dever ser marcados (em vermelho) e, ao lado, vemos um QR Code com o nível L de correção de erro.

São quatro níveis de correção a serem escolhidos (L, M, Q, H).

Tabela 3: Correção de erro.

NÍVEL DE CORREÇÃO DE ERRO	BITS	PIXELS	INTEIRO EQUIVALENTE
L (LOW)	01		**1**
M (MEDIUM)	00		**0**
Q (QUARTILE)	11		**3**
H (HIGH)	10		**2**

Fonte: autores.

Note, na tabela 3 acima, que os valores associados aos níveis de correção de erro não seguem a ordem dos números inteiros. Os níveis de correção de erro estão ordenados do menor para o maior nível, porém os valores inteiros equivalentes seguem a ordem 1, 0, 3 e 2. A figura 12 abaixo nos mostra um exemplo de como mesmo a imagem do QR Code sendo danificada ainda será possível ler a mensagem contida nele.

Veja a tabela 4 abaixo com as porcentagens aproximadas de redundância da mensagem.

Figura 12: QR Code danificado.

Fonte: Wikipedia, 2021.

Tabela 4: Porcentagem da correção de erro.

$L(LOW) \approx 7\%$
$M(MEDIUM) \approx 15\%$
$Q(QUARTILE) \approx 25\%$
$H(HIGH) \approx 30\%$

Fonte: autores.

Tipos de dados armazenado

Os quatro primeiros bits da mensagem do código são para identificar o tipo de caracteres contidos na informação, que são: numéricos, alfanuméricos, byte, kanji, ECI, entre outros, além da possibilidade de combinação de tipos. Vamos mostrar os cinco principais abaixo.

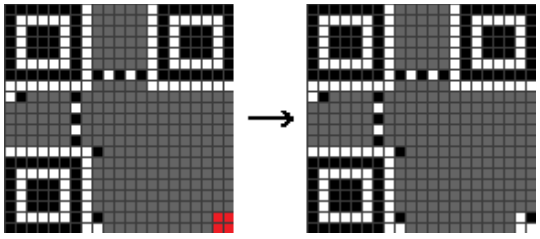
Tabela 5: Tipos de dados.

TIPO	BITS	PIXELS
NUMÉRICO	0001	
ALFANUMÉRICO	0010	
BYTE	0100	
KANJI	1000	
ECI	0111	

Fonte: autores.

Esses quatro bits ficam posicionados no canto inferior direito como podemos ver, a seguir, na figura 13.

Figura 13: Posição dos tipos de dados armazenados.



Fonte: autores.

Acima, à esquerda, vemos em vermelho os quatro bits destinados a este elemento e, à direita, vemos um QR Code preenchido com o tipo alfanumérico.

Segundo ISO/IEC 18004, os quatro modos de codificação incluem os seguintes caracteres:

O numérico é para dígitos decimais de 0 a 9.

O modo alfanumérico é para os dígitos decimais de 0 a 9, bem como letras maiúsculas e os símbolos $,%, *, +, -,., / e o espaço.

Tabela 6: Codificação alfanumérica.

0	0	F	15	U	30
1	1	G	16	V	31
2	2	H	17	W	32
3	3	I	18	X	33
4	4	J	19	Y	34
5	5	K	20	Z	35
6	6	L	21	Espaço	36
7	7	M	22	$	37

8	8	N	23	%	38
9	9	O	24	*	39
A	10	P	25	+	40
B	11	Q	26	-	41
C	12	R	27	.	42
D	13	S	28	/	43
E	14	T	29	:	44

Fonte: autores.

Na tabela acima, temos nas colunas em branco os caracteres supracitados e nas colunas em cinza os valores que cada caractere assumirá na codificação. Por exemplo, se um byte da mensagem carregar como informação a letra A então na sua codificação aparecerá o número 10. Todas as informações sobre bytes e codificações serão expostas nas seções a seguir.

O modo byte, por padrão, é para caracteres do conjunto de caracteres ISO-8859-1 (codificação de caracteres do alfabeto latino). No entanto, alguns scanners de QR Code podem detectar automaticamente se UTF-8 (tipo de codificação binária para caracteres) é usado no modo byte.

E, finalmente, o modo Kanji (Os kanji são caracteres da língua japonesa adquiridos a partir de caracteres chineses) é para caracteres de byte duplo do conjunto de caracteres Shift JIS, que é uma codificação de caracteres para o idioma japonês, originalmente desenvolvido por uma empresa japonesa chamada ASCII Corporation[3] em conjunto com a Microsoft.

Segundo ISO/IEC 18004, se a mensagem consistir apenas em dígitos decimais (0 a 9), será utilizado o modo numérico, se o modo numérico não for suficiente para cobrir toda a informação então será aplicado o modo alfanumérico conforme a tabela 10, se houver um caractere que não está na tabela anterior, mas pode ser codificado em ISO 8859-1, aplicar-se-á o modo byte ou, caso ainda seja necessário, a mensagem será codificada em modo Kanji.

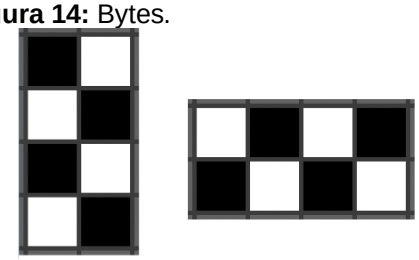
O modo Byte será decodificado pela tabela ASCII que possui 256 caracteres, e é o modo de codificação mais utilizado. Ele precisa de 8 bits para escrever um caractere.

Byte

Os bytes são responsáveis por carregar a mensagem contida na imagem do QR Code, são blocos de 8 bits, como mostra a figura 14, mas dependendo da versão e tipo de dados armazenados a quantidade de bits em um byte pode mudar. Neste trabalho iremos utilizar o 8-bit-byte, ou seja, os bytes de tamanho 8.

[3] A origem da tabela ASCII - Disponível em: <https://en.wikipedia.org/wiki/ASCII> Acesso em: 05 dez, 2021.

Figura 14: Bytes.



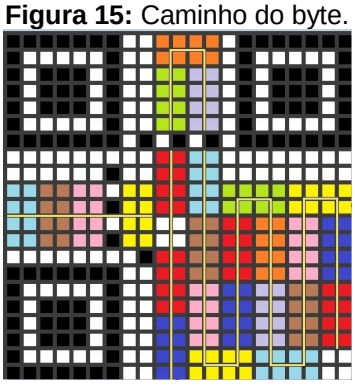
Fonte: autores.

Este elemento do QR Code pode aparecer na posição vertical (4 linhas e 2 colunas) ou horizontal (2 linhas e 4 colunas).

Por padrão, o primeiro byte do QR Code é responsável por carregar a informação sobre a quantidade total de bytes contidos na imagem, ou seja, se um QR Code carrega uma mensagem com 10 caracteres, a palavra “matemática” por exemplo, então o primeiro byte terá o número 11 como informação.

Os bytes da mensagem seguirão o padrão da imagem a seguir, sendo o primeiro byte (em vermelho) o que carrega a informação da quantidade total de bytes, e será alocado logo acima dos quatro primeiros bits no canto inferior direito.

Figura 15: Caminho do byte.



Fonte: autores.

A linha em dourado começa no primeiro byte (vermelho) e segue o caminho da informação dentro do QR Code, ou seja, o segundo byte está na cor azul-escuro, o terceiro na cor amarela e assim por diante, até chegar no último byte de cor azul-claro.
Veja a seguir a ordem dos bytes dentro da imagem nomeados pelas suas respectivas cores.

Tabela 7: Posições e cores dos bytes

1º	Vermelho	6 º	Azul Claro	11 º	Azul Escuro	16 º	Cinza	21 º	Azul
2 º	Azul Escuro	7 º	Cinza	12 º	Amarelo	17 º	Laranja	22 º	Amarelo
3 º	Amarelo	8 º	Laranja	13 º	Rosa	18 º	Verde	23 º	Rosa
4 º	Rosa	9 º	Verde	14 º	Marrom	19 º	Vermelho	24 º	Marrom
5 º	Marrom	10 º	Vermelho	15 º	Azul Claro	20 º	Vermelho	25 º	Azul Claro

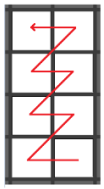
Fonte: autores.

Note que, na figura 15, entre os 19º e 20º bytes existe um bloco com quatro bits brancos. Ele indica o fim da mensagem original, e após esse bloco a mensagem será repetida até que se complete todo o espaço restante do QR Code destinado aos bytes.

Essa repetição é necessária para que o programa leitor possa fazer a restauração de alguma parte danificada do QR Code, como visto, na seção 1.7.

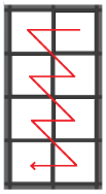
Dentro de cada byte, os bits serão alocados respeitando a ordem mostrada nas figuras 16 a 20, a seguir. Caso os bytes estejam sendo alocados no sentido de baixo para cima os bits serão alocados também de baixo para cima.

Figura 16: Caminho do bit 1.



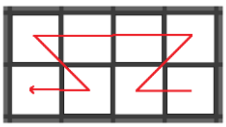
Fonte: autores.

Figura 17: Caminho do bit 4.



Fonte: autores.

Figura 18: Caminho do bit 3

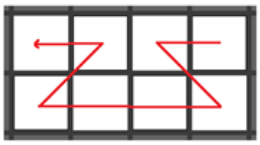


Fonte: autores.

Caso os bytes estejam sendo alocados de cima para baixo, os bits também serão alocados neste sentido.

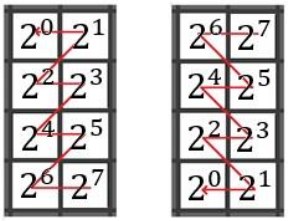
O byte também aparecerá na posição horizontal, com duas linhas e quatro colunas de bits, e será preenchido seguindo as orientações mostradas nas duas próximas figuras.

Figura 19: Caminho do bit 6.



Fonte: autores.

Figura 20: Potências de 2 nos bits



Fonte: autores.

Cada bit possuirá um valor diferente dentro dos bytes, sendo o primeiro bit o mais significativo e o último bit o menos significativo, conforme apresentado na Figura 20.

Exemplo:

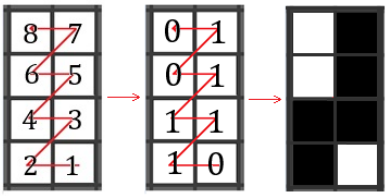
Um byte terá como mensagem a letra “z” que na codificação do Modo Byte terá o valor de 01111010. Se sua escrita é de baixo para cima, como ficará o preenchimento?

Primeiro, para facilitar, organizamos os bits de 1 a 8.

1	2	3	4	5	6	7	8
0	1	1	1	1	0	1	0

Vamos agora preencher cada quadradinho da imagem a seguir, de acordo com a numeração da primeira linha da tabela anterior. De acordo com a segunda linha, cada bit que estiver relacionado com o dígito 1 será pintado de preto.

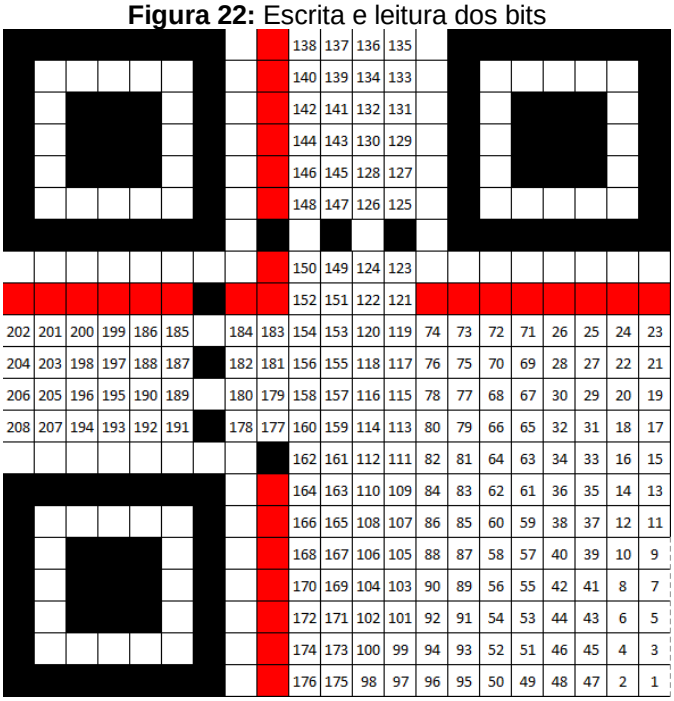
Figura 21: Completando um byte.



Fonte: autores.

A seguir, poderemos ter uma visão geral do comportamento de escrita e leitura dos bits na próxima figura.

Figura 22: Escrita e leitura dos bits



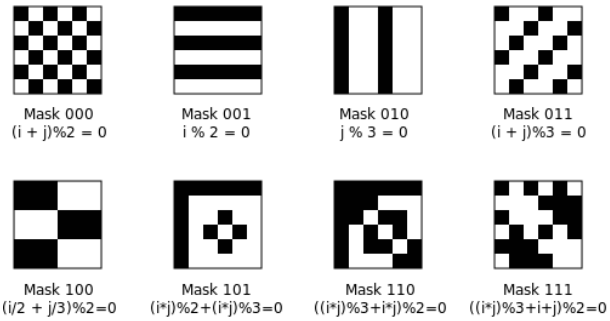
Fonte: autores.

Iniciando pelo bit número 1 na parte inferior direita, que pertence ao bloco dos tipos de dados armazenados, a escrita da informação vai seguindo a ordem dos números naturais até atingir o último bit no número 208. Note que este tipo de escrita segue um padrão de “ziguezague” sempre alternando colunas.

Máscara

As máscaras têm como objetivo modificar a apresentação final do QR Code a fim de evitar grandes blocos de pixels de uma mesma cor, o que pode ocasionar dificuldades ao aplicativo leitor para escanear a imagem. Ela é uma camada de pixels que se sobrepõem ao código original, apenas a parte da mensagem. Além disso, ela evita a aparição do padrão de alinhamento “1011101” no corpo da mensagem. Abaixo temos os oito tipos de máscaras.

Figura 23: Tipos de máscaras



Fonte: Wikimedia, 2021.

Cada máscara possui uma numeração que vai de 0 a 7 convertida para binário, como mostrado na imagem anterior.

Cada pixel da máscara tem uma função específica

- Pixel preto inverte a cor do pixel original.
- Pixel branco mantém a cor do pixel original.

Desta forma, se um pixel branco (ou preto) original do QR Code encontrar um pixel preto da máscara, o pixel mostrado na imagem final será preto (ou branco). Veja abaixo um exemplo.

Figura 24: Aplicação da máscara.

Fonte: Youtube 1, 2021.

Veja a seguir uma imagem contendo oito QR Codes onde foi codificada a mensagem “profmat 2021” e cada um foi sobreposto por um tipo de máscara diferente.

Figura 25: profmat 2021.

Fonte: autores.

A partir daí surge uma pergunta, como o programa escolhe a máscara a ser utilizada?

O programa gerador do QR Code selecionará a imagem com a menor penalidade, sendo que as penalidades são definidas de quatro formas para melhor determinar a distribuição entre bits pretos e brancos.

Tabela 8: Penalidades.

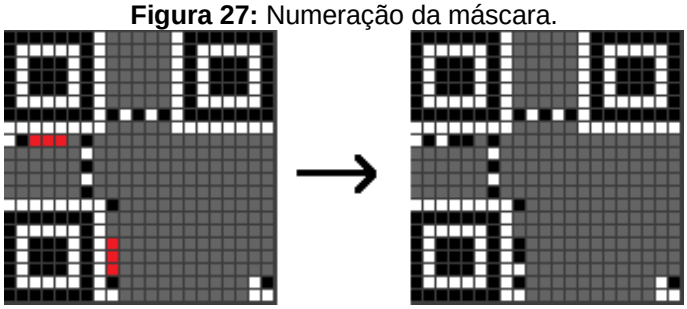
Penalidade	Característica	Condição	Pontos
1	Grupo de cinco ou mais bits da mesma cor em uma linha (ou coluna).	Número de bits $5 + i$	$3 + i$
2	Área de, pelo menos, 2×2 bits da mesma cor	Tamanho do bloco $m \times n$	$3 \times (m-1) \times (n-1)$
3	Existência da razão 1:1:3:1:1 que tenha quatro bits brancos em cada lado.	Existência do padrão	40
4	Proporção entre bits brancos e pretos	$50 \pm (5 \times k)\%$ a $50 \pm (5 \times (k+1))\%$	$40 \times k$

Fonte: autores.

Na tabela acima k é o desvio da proporção de bits pretos em torno dos 50% esperados em passos de 5% e i é a quantidade de bits adjacentes de mesma cor que excede o valor 5.

Na figura 25 a máscara escolhida pelo programa foi a de valor 100 por possuir a menor penalidade entre todas. Cada máscara tem uma numeração de 3 dígitos e essa numeração ocupará os bits em vermelho na figura a seguir.

Figura 27: Numeração da máscara.



Fonte: autores.

Acima temos um QR Code com a máscara 011, ou seja, bits branco-preto-preto. A máscara é a última etapa a ser adicionada ao QR Code.

Após todos esses passos, o programa criador do QR Code irá, através de divisões polinomiais, codificar a informação de formato que terá 15 dígitos. A string de formato sempre tem 15 bits de comprimento.

> Para criar a string, primeiro você cria uma string de cinco bits que codifica o nível de correção de erros e o padrão de máscara em uso neste código QR. Em seguida, você usa esses cinco bits para gerar dez bits de correção de erro. Os quinze bits resultantes são XORed. (Thonky, 2021)

Abaixo temos a tabela completa com as 32 codificações possíveis através dos 4 tipos de nível de correção de erro (low, medium, quartile e high) e dos 8 tipos de máscaras possíveis numeradas de 0 até 7.

Tabela 9: Lista de todas as strings de informações de formato.

Nível de correção de erro	Padrão da Máscara	Tipo de bits de informação	Nível de correção de erro	Padrão da Máscara	Tipo de bits de informação
L	0	111011111000100	Q	0	011010101011111
L	1	111001011110011	Q	1	011000001101000
L	2	111110110101010	Q	2	011111100110001
L	3	111100010011101	Q	3	011101000000110
L	4	110011000101111	Q	4	010010010110100
L	5	110001100011000	Q	5	010000110000011
L	6	110110001000001	Q	6	010111011011010
L	7	110100101110110	Q	7	010101111101101
M	0	101010000010010	H	0	001011010001001
M	1	101000100100101	H	1	001001110111110
M	2	101111001111100	H	2	001110011100111
M	3	101101101001011	H	3	001100111010000
M	4	100010111111001	H	4	000011101100010
M	5	100000011001110	H	5	000001001010101
M	6	100111110010111	H	6	000110100001100
M	7	100101010100000	H	7	000100000111011

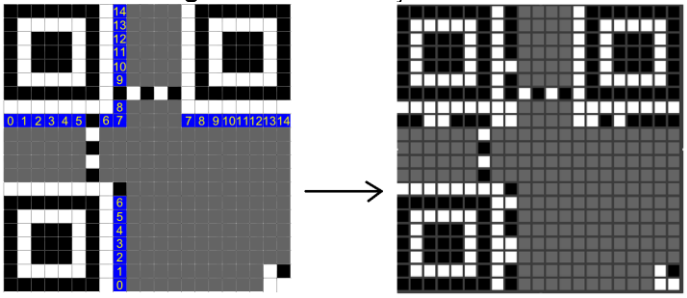
Fonte: autores.

Exemplo: Um QR Code com nível de correção de erro L e padrão de máscara 4 terá o número 110011000101111 como informação de formato de acordo com a tabela anterior. Numerando cada dígito, da esquerda para a direita, de 0 a 14 temos:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	0	0	1	1	0	0	0	1	0	1	1	1	1

Na imagem a seguir poderemos ver o resultado final após o processo de escolha da correção de erro e da máscara e, enfim, a codificação do formato e o preenchimento de cada bit em azul de acordo com sua numeração. Note que os bits de 0 a 4 anteriormente preenchidos com o nível de correção e tipo de máscara serão sobrepostos pela nova numeração.

Figura 28: Codificação do formato.



Fonte: Thonky, 2021.

Aplicativos e sites de QR Code

Os novos smartphones já vem de fábrica com leitor de QR Code embutido na câmera do aparelho, porém, muitos ainda não possuem esta funcionalidade e, com isso, os usuários podem baixar alguns aplicativos gratuitos para a leitura. Veja abaixo alguns exemplos de apps.

Figura 29: Aplicativos de leitura de QR Code e Código de Barras.

Fonte: autores.

Esses aplicativos possuem as funções de criar e ler os códigos, além disso, alguns deles podem inserir logotipos nos QR Codes e muito mais. Alguns sites também disponibilizam essas funções gratuitamente.

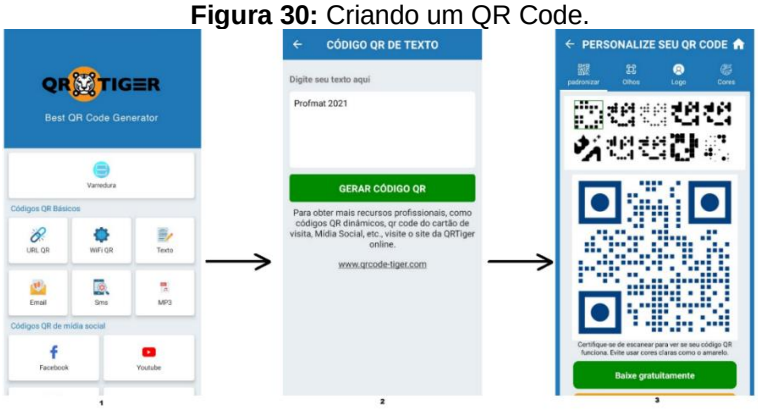
- https://www.qrcodefacil.com/
- https://br.qr-code-generator.com/
- https://www.flowcode.com/
- https://br.qr-code-generator.com/

Outros sites disponibilizam até funções para fins de estudo onde é possível clicar na opção “Show Advanced Options” e escolher o tipo de máscara.

- https://www.thonky.com/qrcode/

Para criar um QR Code usando um aplicativo é fácil. Vamos utilizar o aplicativo QRTiger como exemplo. Na primeira imagem temos a página principal do aplicativo, clicando em “Texto” abre-se a segunda tela onde digitamos a mensagem desejada e clicamos em “GERAR CÓDIGO QR”. Na terceira imagem há opções de personalização do código, como cores, formatos do módulo, inserção de logotipo.

Figura 30: Criando um QR Code.



Fonte: autores.

Finalmente, clicamos em “Baixe gratuitamente” e a imagem será salva na galeria de fotos do celular. Veja abaixo um exemplo de QR Code personalizado feito como o passo a passo anterior.

Figura 31: QR Code customizado.

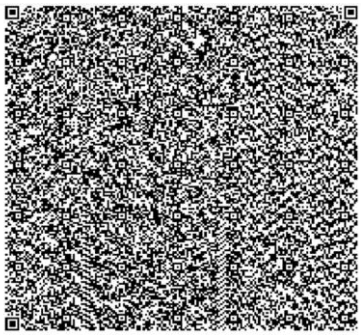
Fonte: autores.

Este possui personalizações como padrão de alinhamento, pixels em formatos diferentes e coloridos, e logotipo no centro.

Atividades em Sala de Aula

Neste capítulo vamos mostrar algumas atividades sugeridas para aplicação em sala de aula. Alguns dos exercícios são dos principais vestibulares nacionais e outros são de criação própria do autor sobre algumas características dos códigos que possam ser exploradas em uma turma.

Atividade 1. Um QR Code Model 2 é uma figura quadrada, pois cada linha e cada coluna possui a mesma quantidade de pixels (quadradinhos). Este modelo de QR Code é dividido em quarenta versões que vão de 1 a 40. Na Versão 1, temos 21 por 21 pixels totalizando 441 quadradinhos, na Versão 2, temos 25 por 25 e, para cada versão maior, a quantidade de pixels, por linha e coluna, aumenta sempre 4 unidades.



Ao lado, vemos a Versão 1 (21x21 pixels), à esquerda, e a Versão 40 (177x177 pixels), à direita. Com base no crescimento da quantidade de pixels em um QR Code, podemos afirmar que é:

a) Linear
b) Exponencial
c) Quadrático
d) Logarítmico
e) N.R.A

Resolução:

Como foi dito no texto, partindo da Versão 1 (21x21), cada versão do QR Code aumenta de quatro em quatro a quantidade de pixels nas linhas e colunas.

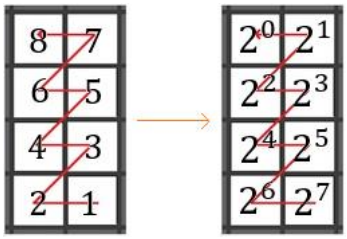
Versão1 - 21×21 – 441 pixels
Versão2 - 25×25 – 625 pixels
Versão3 - 29×29 – 814 pixels

Até

Versão40 - 177×177 – 31329 pixels

Logo, a quantidade de pixels em uma imagem de QR Code é sempre um quadrado perfeito e, com isso, seu crescimento ao longo das versões é quadrático.

Atividade 2. Em um QR Code, um byte pode ser um bloco de 8 bits (quadradinhos) que contém um caractere como mensagem cada. Cada bit pode receber a cor branca ou preta que correspondem, respectivamente, a zero ou um. Dentro de cada byte a pintura das cores correspondem a seguinte ordem.



Ao lado, vemos que o primeiro bit pintado corresponde a potência 2^7, o segundo a 2^6, e assim por diante até chegarmos a potência 2^0 que corresponde ao oitavo bit.
Vamos utilizar a tabela de codificação abaixo para consulta dos valores associados a cada caractere que utilizaremos.

Decimal	Caractere	Binário:	Decimal	Caractere	Binário:
48	0	00110000	105	i	01101001
49	1	00110001	106	j	01101010
50	2	00110010	107	k	01101011
51	3	00110011	108	l	01101100
52	4	00110100	109	m	01101101
53	5	00110101	110	n	01101110
54	6	00110110	111	o	01101111
55	7	00110111	112	p	01110000
56	8	00111000	113	q	01110001
57	9	00111001	114	r	01110010
97	a	01100001	115	s	01110011
98	b	01100010	116	t	01110100
99	c	01100011	117	u	01110101
100	d	01100100	118	v	01110110
101	e	01100101	119	w	01110111
102	f	01100110	120	x	01111000
103	g	01100111	121	y	01111001
104	h	01101000	122	z	01111010

Note que, a letra “a” está associada ao valor 97 que em binário corresponde a $(1100001)_2$,como este valor possui sete algarismos, adicionamos um caractere de valor zero à esquerda dele formando assim $(01100001)_2$. Pois, cada byte precisa ter “tamanho” de 8 bits.

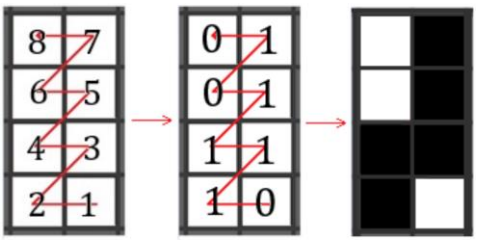
Agora, veja um exemplo de preenchimento de um byte.

Um byte terá como mensagem a letra “z” que na codificação do Modo Byte terá o valor de 01111010, se sua escrita é como foi mostrado anteriormente, como ficará o preenchimento?

Primeiro, para facilitar, organizamos os bits de 1 a 8, como a tabela a seguir.

1	2	3	4	5	6	7	8
0	1	1	1	1	0	1	0

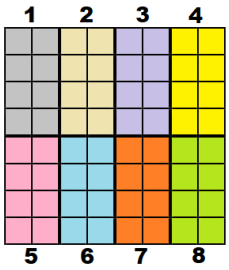
Vamos agora preencher cada quadradinho da imagem a seguir de acordo com a numeração da primeira linha da tabela anterior. Analisando a segunda linha, cada bit que estiver relacionado com o dígito 1 será pintado de preto.



Agora, realizaremos a pintura de uma figura de 8 bytes com a mensagem “educador”. Cada letra da palavra a ser codificada será associada ao byte cujo valor corresponde a sua posição dentro da palavra.

e (byte1), d (byte 2), u (byte 3), c (byte 4)

a (byte 5), d (byte 6), o (byte 7), o (byte 8)



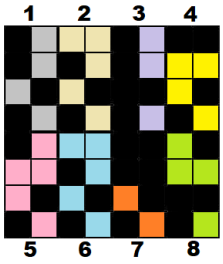
Os bits estão coloridos para facilitar a identificação de cada byte. Na resolução, vamos pintar apenas os bits de cor preta.

Resolução:

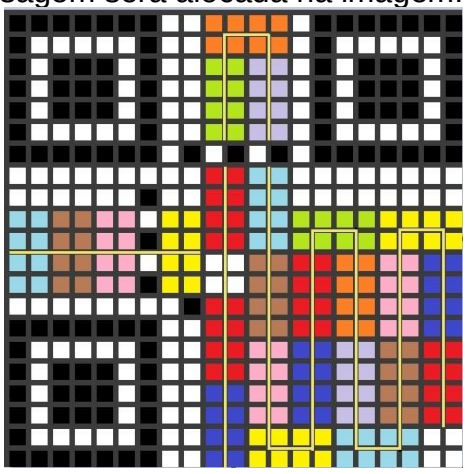
Primeiro verificamos o valor que cada letra está associada nos números decimais. Após isso fazemos a conversão de cada valor para binário. Veja a tabela a seguir.

e	101	01100101	a	97	01100001
d	100	01100100	d	100	01100100
u	117	01110101	o	111	01101111
c	99	01100011	r	114	01110010

Por último, cada letra será pintada seguindo a ordem posicional.



Atividade 4. O QR Code da versão 1 possui 21×21 = 441 pixels (bits), que são quadradinhos pretos ou brancos, sendo que o quadrado subdividido em quatro bits na parte inferior direita indica o conteúdo desta atividade que será letras e números no Modo Byte, nos bits 1, 2, 3 e 4, e serão marcados com os dígitos 0100, e a parte colorida indica onde a mensagem será alocada na imagem.

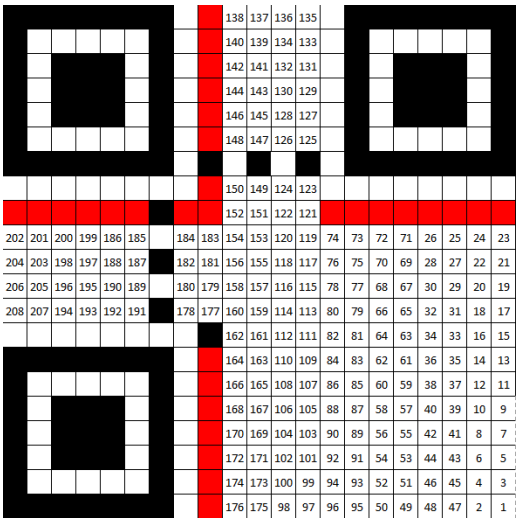


Decimal	Caractere	Binário:	Decimal	Caractere	Binário:
48	0	00110000	105	i	01101001
49	1	00110001	106	j	01101010
50	2	00110010	107	k	01101011
51	3	00110011	108	l	01101100
52	4	00110100	109	m	01101101
53	5	00110101	110	n	01101110
54	6	00110110	111	o	01101111
55	7	00110111	112	p	01110000
56	8	00111000	113	q	01110001
57	9	00111001	114	r	01110010
97	a	01100001	115	s	01110011
98	b	01100010	116	t	01110100
99	c	01100011	117	u	01110101
100	d	01100100	118	v	01110110
101	e	01100101	119	w	01110111
102	f	01100110	120	x	01111000
103	g	01100111	121	y	01111001
104	h	01101000	122	z	01111010

Cada parte da mensagem é subdividida em 8 quadradinhos (byte) que serão preenchidos de acordo com uma ordem específica (segunda imagem). Os bits brancos e pretos indicam os dígitos binários 0 e 1, respectivamente. Os bits de 5 a 12 (primeiro byte) indicam quantos caracteres estarão presentes na mensagem, por exemplo, se contém a palavra “escola ” então o primeiro byte terá o número 7 como mensagem, que corresponde ao primeiro byte mais seis bytes da “escola”.

Modo Byte	1º Byte (7)	e	s	c	o	l	a
0100	00110111	01100101	01110011	01100011	01101111	01101100	01100001

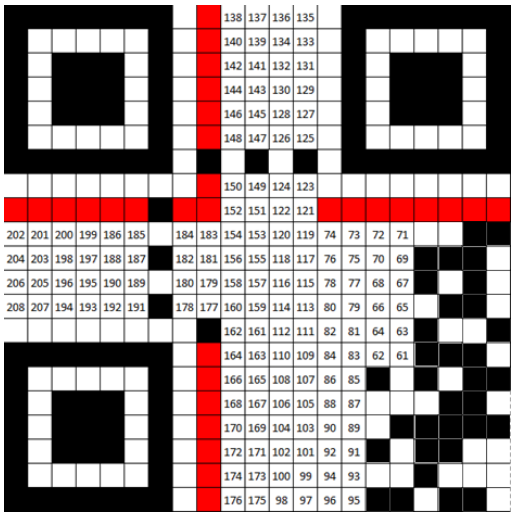
(Se o valor for maior do que 9 então basta transformar o decimal para binário e preenchê-lo à esquerda para ficar com 8 algarismos de tamanho). A partir do bit 13 em diante até o bit 208 será escrita a mensagem desejada. Por exemplo, codificando um QR Code com a mensagem “escola” no Modo Byte e com base na tabela ao lado, teremos a seguinte ordem dos bits.



Desta forma, utilizaremos os 60 primeiros bits do QR Code e pintaremos de preto os bits relacionados aos dígitos 1. A tabela abaixo está preenchida com os 60 algarismos da tabela anterior relacionando-os à numeração do bit correspondente.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	1	0	0	0	0	1	1	0	1	1	1	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
1	0	0	1	1	1	0	1	1	0	1	1	1	1	0	1	1	0	1	1	0	0	0	1	1	0	0	0	0	1

O resultado será o QR Code pintado abaixo.



Com base nas informações apresentadas, preencha o QR Code a seguir com a mensagem “mestradoprofmatunirio” no Modo Byte (tabela de codificação da página anterior). Utilize as tabelas a seguir como guia para o preenchimento.

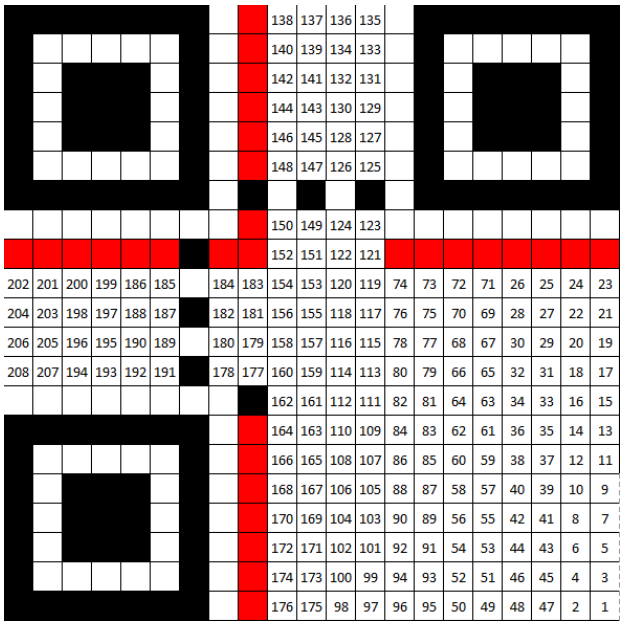
Preencha a tabela abaixo com os valores binários do Modo Byte, do primeiro byte e dos caracteres.

Modo Byte	1º Byte	m	e	s	t	r	a	d	o	p	r
o	f	m	a	t	u	n	i	r	i	o	

Complete a tabela abaixo com os valores de cada bit segundo a tabela acima.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180

Pinte de preto os quadradinhos do QR Code correspondentes ao algarismo 1.



Resolução: Vamos preencher a primeira tabela com os valores dos caracteres no modo binário.

1º) Preencher o valor do Modo Byte (0100)

2º) Preencher o primeiro byte com o valor correspondente ao número 23 em binário (10111), sendo que, como o número precisar ter tamanho de 8 bits, então adicionamos 3 zeros à esquerda, e finalmente temos (00010111).

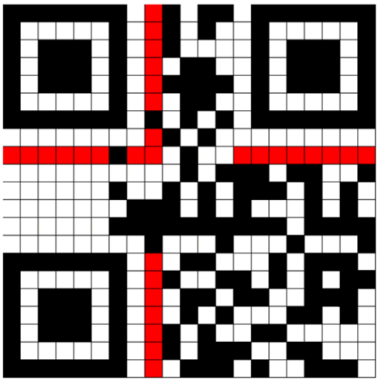
3º) Preencher o restante da tabela com o valor correspondente as letras.

Modo Byte	1º Byte	m	e	s	t	r	a	d	o	p	r
0100	00010111	01101101	01100101	01110011	01110100	01110010	01100001	01100100	01101111	01110000	01110010
o	f	m	a	t	u	n	i	r	i	o	
01101111	01100110	01101101	01100001	01110100	01110101	01101110	01101001	01110010	01101001	01101111	

O passo seguinte é preencher a segunda tabela com os 180 bits da atividade.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	1	0	0	0	0	0	1	0	1	1	1	0	1	1	0	1	1	0	1	0	1	1	0	0	1	0	1	0	1
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
1	1	0	0	1	1	0	1	1	1	0	1	0	0	0	1	1	1	0	0	1	0	0	1	1	0	0	0	0	1
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90
0	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	0	1	1	1	0	0	0	0	0	1	1	1	0	0
91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
1	0	0	1	1	0	1	1	1	1	0	1	1	0	0	1	1	0	0	1	1	0	1	1	0	1	0	1	1	0
121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
0	0	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	0	1	0	1	1	0	1	1	1	0	0	1
151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
1	0	1	0	0	1	0	1	1	1	0	0	1	0	0	1	1	0	1	0	0	1	0	1	1	0	1	1	1	1

Agora, vamos pintar de preto cada quadradinho correspondente ao algarismo 1 na tabela acima.



Este será o resultado final da pintura, os bits brancos foram pintados apenas para ilustração.

Considerações Finais

Esperamos que esse trabalho possa contribuir para os docentes que se interessem por novas tecnologias nas salas de aula do Ensino Básico e que possa ajudar os alunos a compreenderem melhor os códigos que nos cercam diariamente e que possa ser uma proposta diferenciada para o ensino dos conteúdos de Matemática correlacionados ao QR code.

Estamos inseridos em uma sociedade onde o uso de tecnologias vem aumentando rapidamente e a educação precisa acompanhar essas evoluções, trazendo para dentro da sala de aula algumas inovações, com a finalidade de formar jovens capazes de compreender o mundo que o cerca e ter um diferencial no mercado de trabalho cada vez mais disputado. Além disso, uma aula com códigos poderá despertar no aluno a curiosidade de aprender mais sobre o tema e, assim, o discente terá a possibilidade de trilhar sua carreira no mundo matemático e/ou informático. Também esperamos que este trabalho sirva de material bibliográfico para os professores se aprofundarem no assunto e criarem outras atividades e que inspire novos estudos sobre a aplicação dos códigos na sala de aula.

Referências Bibliográficas

GS1BR - Disponível em: https://gs1br.org/codigos-e-padroes/captura/gs1-qr-code. Acesso em: 24 mai. 2021.

ISO/IEC 18004:2015 - Information technology - Automatic identification and data capture techniques - QR Code 2015 bar code symbology specification, 2015.

GS1BR - Disponível em: https://www.gs1br.org/codigos-e-padroes/captura/Paginas/GS1-QR-Code.aspx. Acesso em: 24 mai. 2021.

PINTO, A. C. M; FELCHER, C. D. O; FERREIRA, A. L. A. **Considerações sobre o uso do aplicativo QR CODE no ensino da matemática**: reflexões sobre o papel do professor, São Paulo, 2016.

Disponível em http://www.sbem.com.br/enem2016/anais/pdf/8323_4386_ID.pdf. Acesso 20 jan. 2021.

SILVA, T. B. da; BEZERRA, S. M. C. B. **O Uso do Qr Code no Ensino De Matemática na Formação Inicial**. X Simpósio Linguagens e Identidades da/na Amazônia sul-ocidental VIII Colóquio Internacional "As Amazônias, as Áfricas na Pan-Amazônia", 2016.

SILVA, A. R. D; FANTIN, S. **A matemática do código de barras e Qr Code**. Programa De Pós-Graduação Matemática Em Rede Nacional (Profmat) UNIRIO, 2021.

Referências das imagens

Thonky - Disponível em: https://www.thonky.com/qr-code-tutorial/. Acesso em: 23 jan. 2021.
Wikipedia - Disponível em: https://pt.wikipedia.org/wiki/C%C3%B3digo_QR. Acesso em: 23 jan. 2021.
Wikimedia - Disponível em: https://commons.wikimedia.org/wiki/File:QR_Code_Mask_Patterns.svg. Acesso em: 31 jan. 2021.
Youtube - Disponível em: https://www.youtube.com/watch?v=142TGhaTMtI. Acesso em: 25 jan. 2021.

NOTAS

IDENTIFICAÇÃO DO TEXTO

O presente texto é um recorte de A Matemática do Código de Barras e QR Code, dissertação de Mestrado Profissional em Matemática em Rede Nacional (Profmat) apresentada na Universidade Federal do Estado do Rio de Janeiro (UNIRIO), em 12/08/2021, elaborada sob orientação do Professor Dr. Silas Fantin.

IDENTIFICAÇÃO DE AUTORIA

Alberto Renan Dias da Silva. Mestre em Matemática pelo Mestrado Profissional em Matemática em Rede Nacional (Profmat). Universidade Federal do Estado do Rio de Janeiro (UNIRIO), Rio de Janeiro, RJ, Brasil.
E-mail: alberto.renan.dias@gmail.com
https://orcid.org/0000-0002-4889-9532

Silas Fantin. Doutor em Matemática pela Universidade de São Paulo (USP). Professor associado da Universidade Federal do Estado do Rio de Janeiro (UNIRIO), Rio de Janeiro, RJ, Brasil.
E-mail: silas.fantin@uniriotec.br
https://orcid.org/0000-0002-2183-6806

AGRADECIMENTOS

Os autores agradecem à Universidade Federal do Estado do Rio de Janeiro (UNIRIO).
O primeiro autor agradece em especial ao professor Dr. Silas Fantin, orientador e coautor do artigo.

FINANCIAMENTO

Não se aplica.

CONSENTIMENTO DE USO DE IMAGEM

Não se aplica.

APROVAÇÃO DE COMITÊ DE ÉTICA EM PESQUISA

Não se aplica.

EDITORES
Patricia Helena Mirandola Garcia, Eugenia Brunilda Opazo Uribe, Gerson dos Santos Farias.

HISTÓRICO
Recebido em: 28/08/2021 – Aprovado em: 04/12/2020 – Publicado em: 15/12/2021.

COMO CITAR
SILVA, A. R. D; FANTIN, S. A Matemática do QR Code. **Revista ENSIN@ UFMS**, Três Lagoas, v. 2, número especial, p. 374-399. 2021.

.