Lecture-2

Representation of data

- Signed integers
  - 1’s and 2’s complements
- Floating point numbers
- characters
- Images
- sounds

Last week

Engineer vs Scientist

Computer Engineer

Digital Machine

Digital Numerical Systems

Conversions

$(111001101010001)_2 = (?)_{10}$

$(111001101010001)_2 = (?)_{16}$

$(111001101010001)_2 = (?)_8$

$(111001101010001)_2 = (?)_4$

Conversions

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

$(1\text{- }1\text{- }1\text{- }0\text{- }0\text{-}1\text{-}1\text{-}0\text{-}1\text{-}0\text{-}1\text{-}0\text{-}0\text{-}0\text{-}1)_{\mathbf{b\ =\ 2}}$ = $(?)_{10}$

	b^{14}	b^{13}	b^{12}	b^{11}	b^{10}	b^{9}	b^{8}	b^{7}	b^{6}	b^{5}	b^{4}	b^{3}	b^{2}	b^{1}	b^{0}					
	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}					
(	1	1	1	0	0	1	1	0	1	0	1	0	0	0	1	$)_2$	=	(	29511	$)_{10}$
	16384	+8192	+4096	+0	+0	+512	+256	+0	+64	+0	+16	+0	+0	+0	+1					

$(111\text{-}0011\text{-}0101\text{-}0001)_2 = (7\text{-}3\text{-}5\text{-}1)_{16}$

0001 = 1

0101 = 5

0011 = 3

0111 = 7

$$(111\text{-}001\text{-}101\text{-}010\text{-}001)_2 = (7\text{-}1\text{-}5\text{-}2\text{-}1)_8$$

$$(1\text{-}11\text{-}00\text{-}11\text{-}01\text{-}01\text{-}00\text{-}01)_2 = (1\text{-}3\text{-}0\text{-}3\text{-}1\text{-}1\text{-}0\text{-}1)_4$$

The most significant bit(MSB) and the least significant bit(LSB)

1	1	1	0	0	1	1	0	1	0	1	0	0	0	1

MSB vs LSB

	b^{14}	b^{13}	b^{12}	b^{11}	b^{10}	b^{9}	b^{8}	b^{7}	b^{6}	b^{5}	b^{4}	b^{3}	b^{2}	b^{1}	b^{0}					
	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}					
(	1	1	1	0	0	1	1	0	1	0	1	0	0	0	1	$)_2$	=	(	29511	$)_{10}$
	16384	+8192	+4096	+0	+0	+512	+256	+0	+64	+0	+16	+0	+0	+0	+1					

Without MSB = **13137**

Without MSB = 29510

How to represent more complex data in binary

Characters (symbols):

- 1,0, s, x_,!$%^alsjkdom;lsmdf;l/*65

Numbers

- Integers
- Signed integers

Floating point numbers

- 1.4
- 5.25

Images ?

Musics ?

Signed integers

The MSB is the signed bit

- 0 for +
- 1 for -

0	1	1	0	0	1	1	0	1	0	1	0	0	0	1

With MSB = 0

+13137

1	1	1	0	0	1	1	0	1	0	1	0	0	0	1

With MSB = 1

-13137

Fractions

3.14159265359

How to represent in binary?

We know $3 = (011)_2$

$0.14159265359 = (?)_2$

$1/2 = 1 \times \mathbf{2^{-1}} = 0.5 = (0.1)_2$
$0.75 = 1 \times \mathbf{2^{-1}} + 1 \times \mathbf{2^{-2}} = (0.11)_2$

	b^1	b^0	.	b^{-1}	b^{-2}	b^{-3}	b^{-4}				
	2^1	2^0	.	$\mathbf{2^{-1}}$	$\mathbf{2^{-2}}$	$\mathbf{2^{-3}}$	$\mathbf{2^{-4}}$				
(	1	1	.	**0**	**0**	**1**	$1)_2$	=	(	3.1875	$)_{10}$
	1x2	1x1		0x1/2	0x1/4	1x1/8	1x1/16				

$1/2 = 1 \times \mathbf{2}^{-1} = 0.5 = (0.1)_2$

$1/3 = 0 \times \mathbf{2}^{-1} + 1 \times \mathbf{2}^{-2} + 0 \times \mathbf{2}^{-3} + 1 \times \mathbf{2}^{-4} + \ldots = 0.3125 + \ldots$

Fraction	Decimal	Binary	Fractional approximation
1/1	1 or 0.999...	1 or $0.\overline{1}$	1/2 + 1/4 + 1/8...
1/2	0.5 or 0.4999...	0.1 or $0.0\overline{1}$	1/4 + 1/8 + 1/16 . . .
1/3	0.333...	$0.\overline{01}$	1/4 + 1/16 + 1/64 . . .
1/4	0.25 or 0.24999...	0.01 or $0.00\overline{1}$	1/8 + 1/16 + 1/32 . . .
1/5	0.2 or 0.1999...	$0.\overline{0011}$	1/8 + 1/16 + 1/128 . . .
1/6	0.1666...	$0.001\overline{01}$	1/8 + 1/32 + 1/128 . . .
1/7	0.142857142857...	$0.\overline{001}$	1/8 + 1/64 + 1/512 . . .
1/8	0.125 or 0.124999...	0.001 or $0.000\overline{1}$	1/16 + 1/32 + 1/64 . . .
1/9	0.111...	$0.\overline{000111}$	1/16 + 1/32 + 1/64 . . .
1/10	0.1 or 0.0999...	$0.0\overline{0011}$	1/16 + 1/32 + 1/256 . . .
1/11	0.090909...	$0.\overline{0001011101}$	1/16 + 1/64 + 1/128 . . .
1/12	0.08333...	$0.0001\overline{01}$	1/16 + 1/64 + 1/256 . . .
1/13	0.076923076923...	$0.\overline{000100111011}$	1/16 + 1/128 + 1/256 . . .
1/14	0.0714285714285...	$0.0\overline{001}$	1/16 + 1/128 + 1/1024 . . .
1/15	0.0666...	$0.\overline{0001}$	1/16 + 1/256 . . .
1/16	0.0625 or 0.0624999...	0.0001 or $0.0000\overline{1}$	1/32 + 1/64 + 1/128 . . .

What about “.” in 3.14

Floating point arithmetic

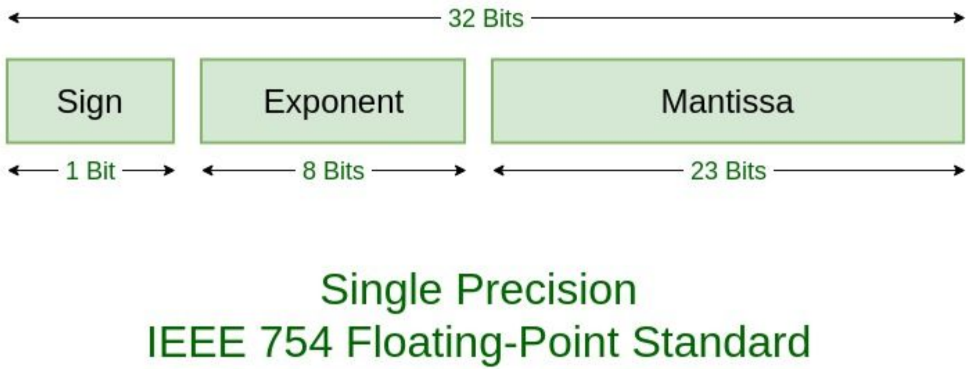
Represents subset of real numbers using **integers** with **fixed precision:**

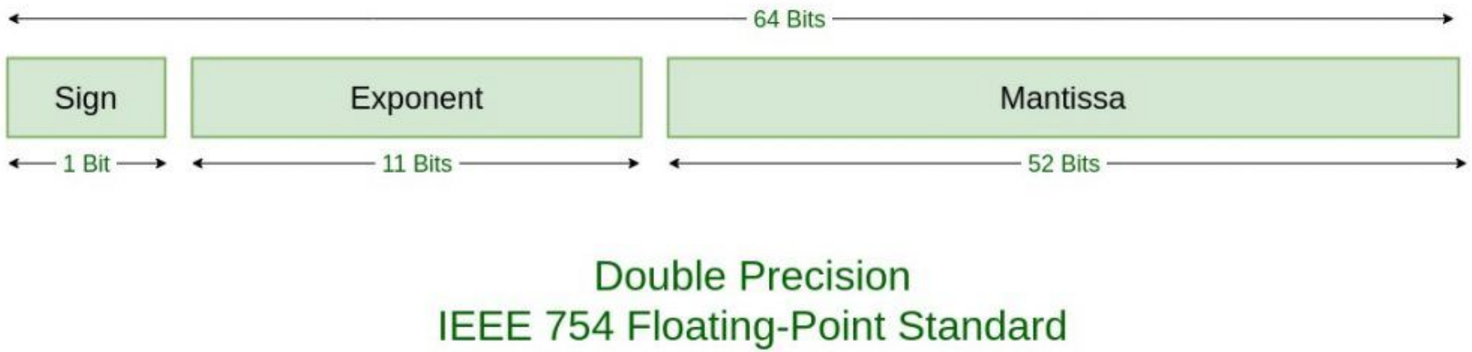
1. **Mantissa - Significand (significant digits)**
2. **Exponent (scaling integer)**

$$12.345 = \underbrace{12345}_{\text{significand}} \times \underbrace{10}_{\text{base}}^{\overbrace{-3}^{\text{exponent}}}$$

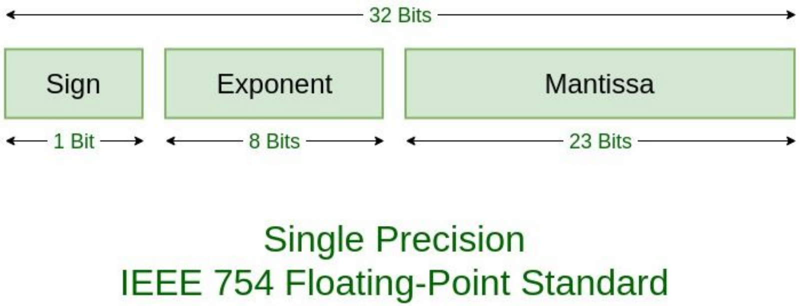
https://www.geeksforgeeks.org/ieee-standard-754-floating-point-numbers/

The IEEE Standard for Floating-Point Arithmetic (IEEE 754)





The IEEE Standard for Floating-Point Arithmetic (IEEE 754)



Example: 85.125 = $(\mathbf{1010101.001})_2$

Number = $(-1)^{sign} \times 1.F \times 2^{exponent-127}$

The IEEE Standard for Floating-Point Arithmetic (IEEE 754)

Example: 85.125 = **$(1010101.001)_2$**

=**(1.010101001×2^6)**

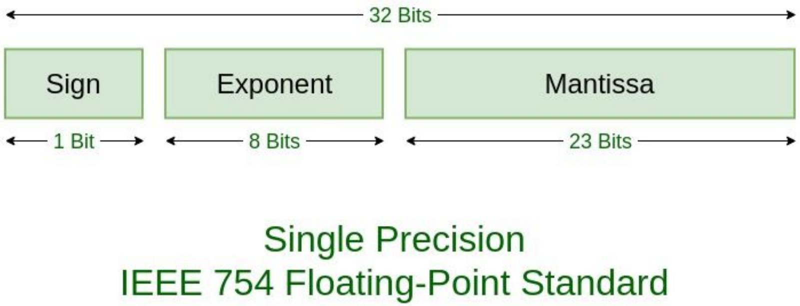
Sign = 0

- Add bias to **exponent**
  - 127+6 = 133 (even if exponent -, it becomes +)
  - 133 = **$(10000101)_2$**
- **Mantissa 010101001 add 0s to complete 23 bits**

The IEEE 754 Single precision is:

= **0 10000101 01010100100000000000000**

This can be written in hexadecimal form **42AA4000**



Number = $(-1)^{sign} \times 1.F \times 2^{exponent-127}$

Binary Integer operations

Addition (Add

Subtraction (Sub)

Division(Div)

0 + 0 → 0
0 + 1 → 1
1 + 0 → 1
1 + 1 → 0, carry 1 (since 1 + 1 = 2 = 0 + (1 × 2^1))

```
      1 0 1 1   (A)
    × 1 0 1 0   (B)
    ---------
      0 0 0 0   ← Corresponds to the rightmost 'zero' in B
+     1 0 1 1    ← Corresponds to the next 'one' in B
+   0 0 0 0
+ 1 0 1 1
---------------
= 1 1 0 1 1 1 0
```

How to do operations with signed integers?

0	1	1	0	0	1	1	0	1	0	1	0	0	0	1

With MSB = 0

+13137

With MSB = 1

-13137

When sign bit used…

Assume we have 3 bits

0.., + numbers

1.. - numbers

0	1	2	3	4	5	6	7
-3	-2	-1	-0	+0	+1	+2	+3
111	110	101	100	000	001	010	011

Why math works in decimal?

In decimal

$(-5)_{10} + (+5)_{10} = (0)_{10}$

Make the math works in binary

In decimal

$(-5)_{10} + (+5)_{10} = (0)_{10}$

In signed binary using (1,0) for (+-)

$(1???)_2 + (0101)_2 = (0000)_2$

Ignore carry(extra) bit,

Say $x = (1???)_2$

$x = (1\ 0000)_2 - (101)_2 = (1011)_2$

$(1011)_2$ 2's complement representation of **(-5)**

examples

0000 1111 (15)

+ 1111 1011 (−5)

===========

0000 1010 (10)

—----------------------------------

0000 0101 (5)

+ 1111 0001 (−15)

===========

1111 0110 (−10)

Two's complement	Decimal
0111	7.
0110	6.
0101	5.
0100	4.
0011	3.
0010	2.
0001	1.
0000	0.
1111	−1.
1110	−2.
1101	−3.
1100	−4.
1011	−5.
1010	−6.
1001	−7.
1000	−8.

https://en.wikipedia.org/wiki/Two%27s_complement

1s complement vs 2s complement

```
 1111 1111                            255.
− 0101 1111                           − 95.
===========                           =====
  1010 0000  (ones' complement)        160.
+         1                           +  1
===========                           =====
  1010 0001  (two's complement)        161.
```

How to represent more complex data in binary

Integers

- Signed integers
- 2s complement Two's complement - Wikipedia

Floating point numbers IEEE Standard 754 Floating Point Numbers - GeeksforGeeks

- 1.4
- 5.25
- IEEE 754

➔ Characters (symbols):
  - 1,0, s, x_,!$%^alsjkdom;lsmdf;l/*65

➔ Images ?

➔ Musics ?

How to represent more complex data in binary

Integers

- Signed integers
- 2s complement Two's complement - Wikipedia

Floating point numbers IEEE Standard 754 Floating Point Numbers - GeeksforGeeks

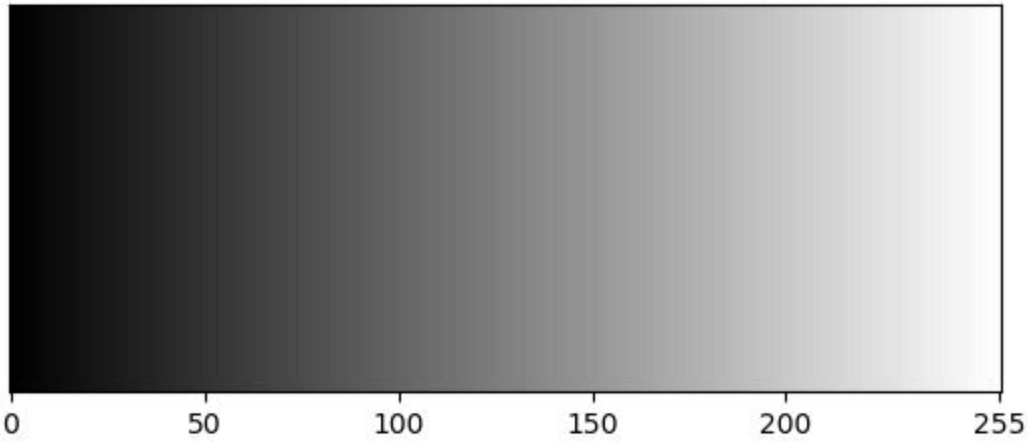
- 1.4
- 5.25
- IEEE 754

➔ Characters (symbols):
  - 1,0, s, x_,!$%^alsjkdom;lsmdf;l/*65
  - ASCII codes ASCII table
  -
  -

➔ Images ?
  - RGB values
    - Colors RGB and RGBA

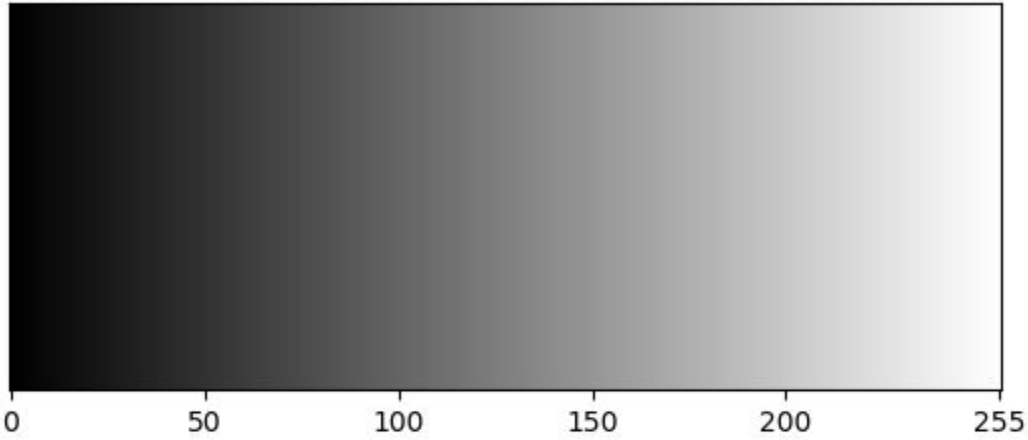
➔ Musics ?

Grey scale image



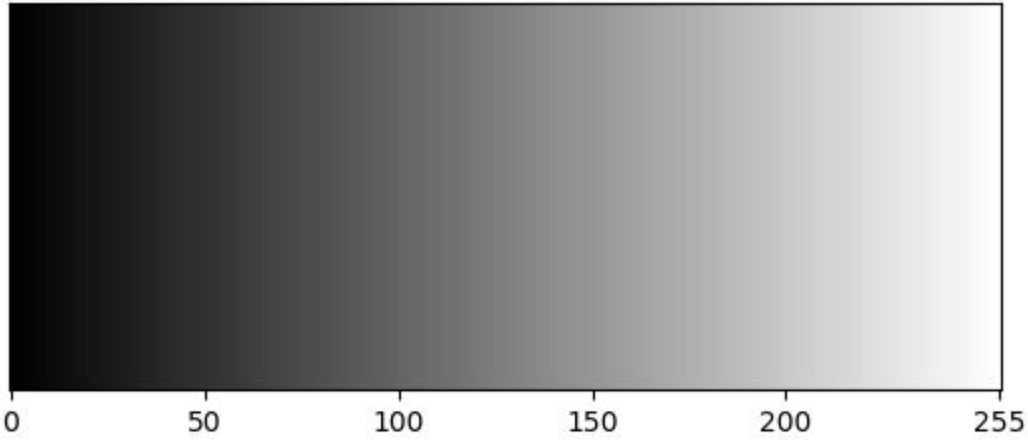
https://theailearner.com/2018/10/22/create-own-image-using-numpy-and-opencv/

Grey scale image



https://theailearner.com/2018/10/22/create-own-image-using-numpy-and-opencv/

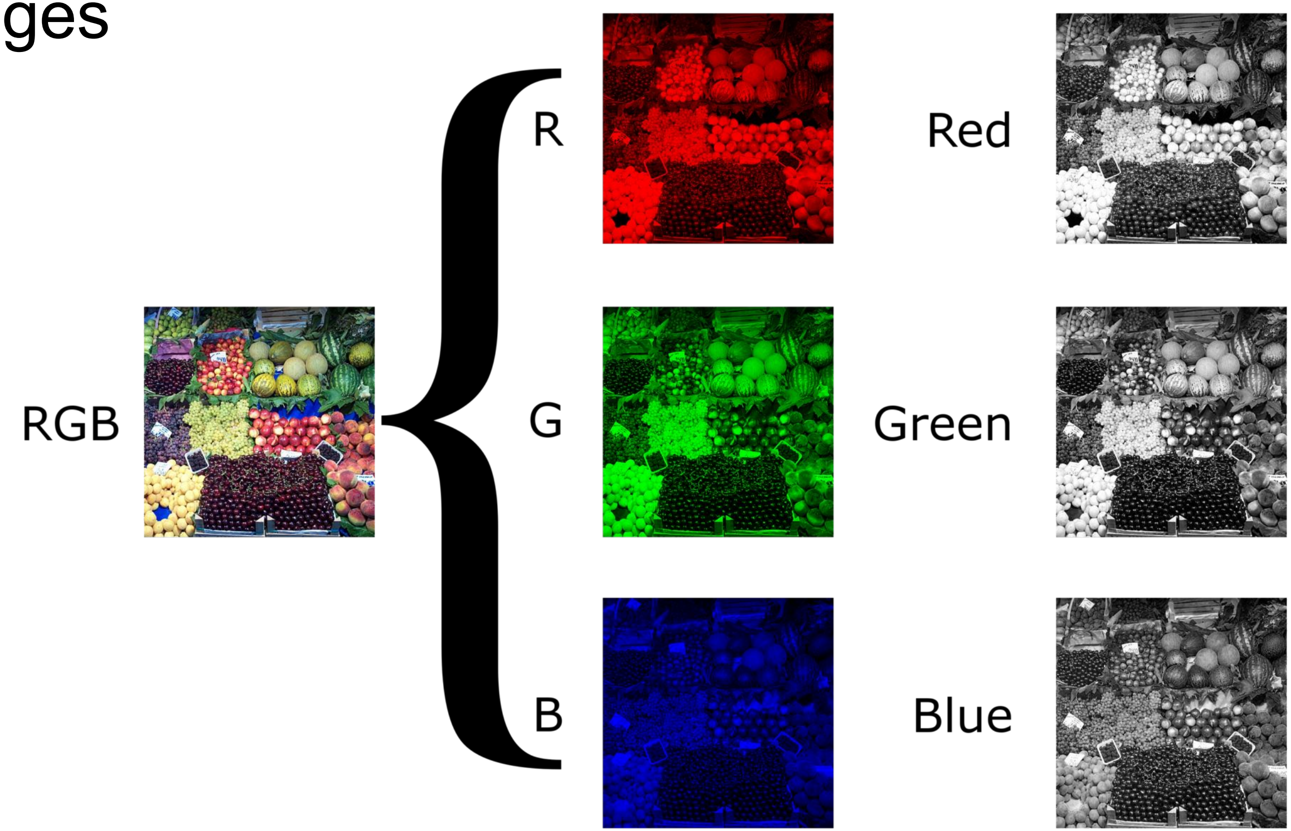
Grey scale image



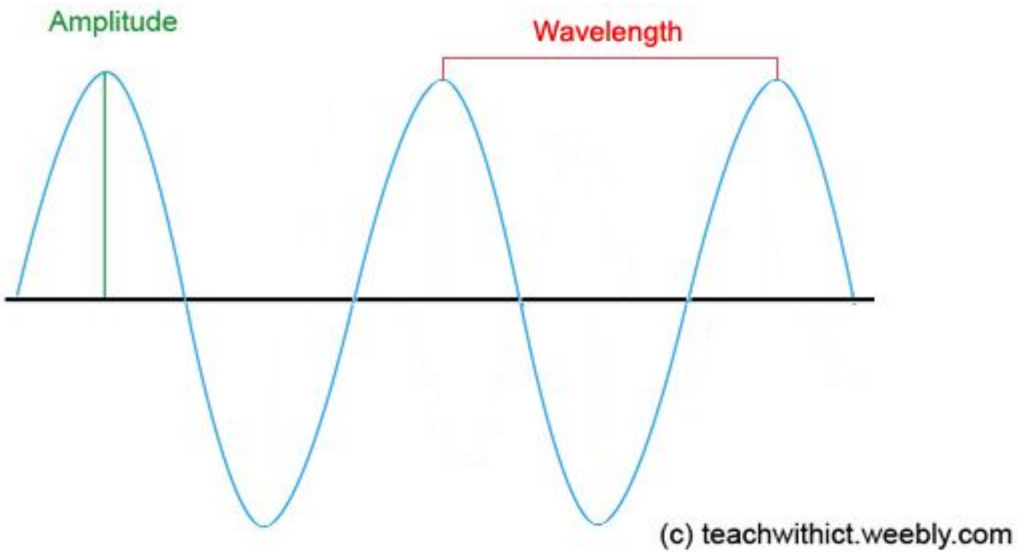
https://theailearner.com/2018/10/22/create-own-image-using-numpy-and-opencv/

https://en.wikipedia.org/wiki/Grayscale#

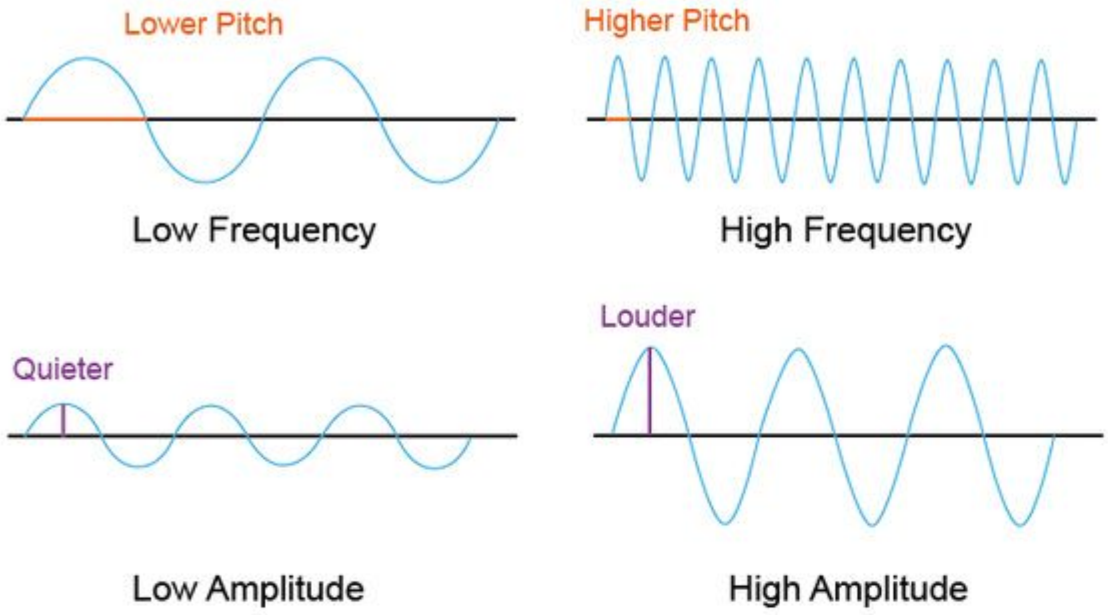
RGB color images



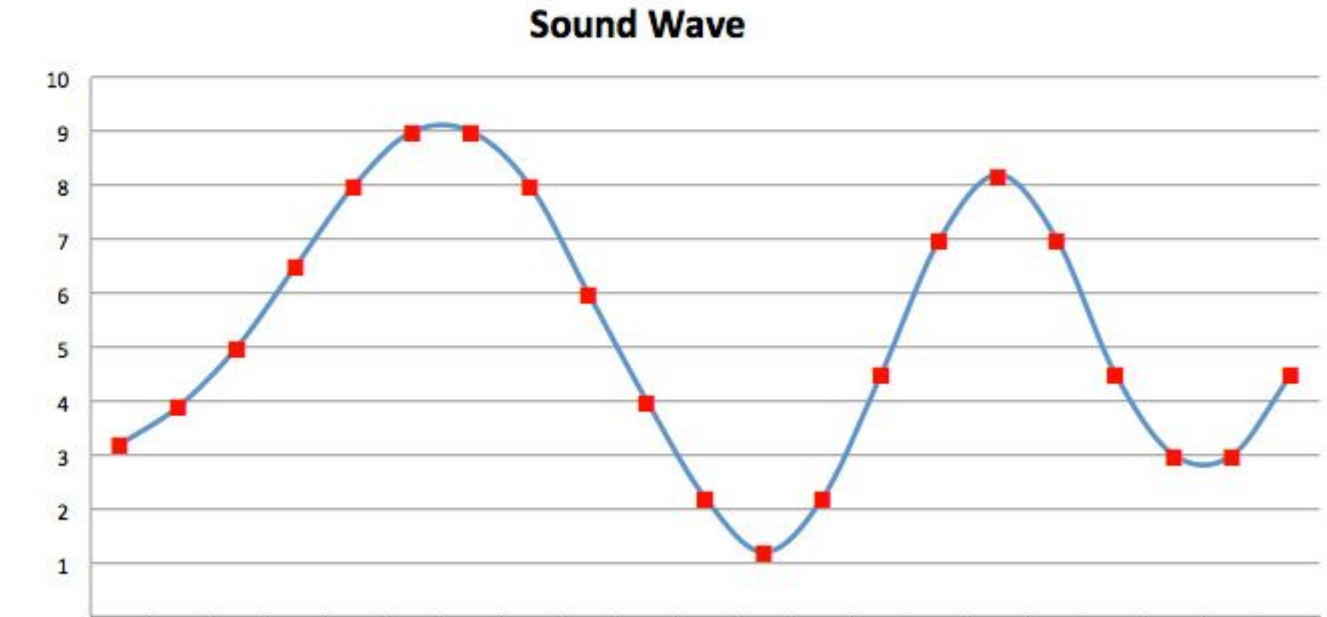
Sound representation



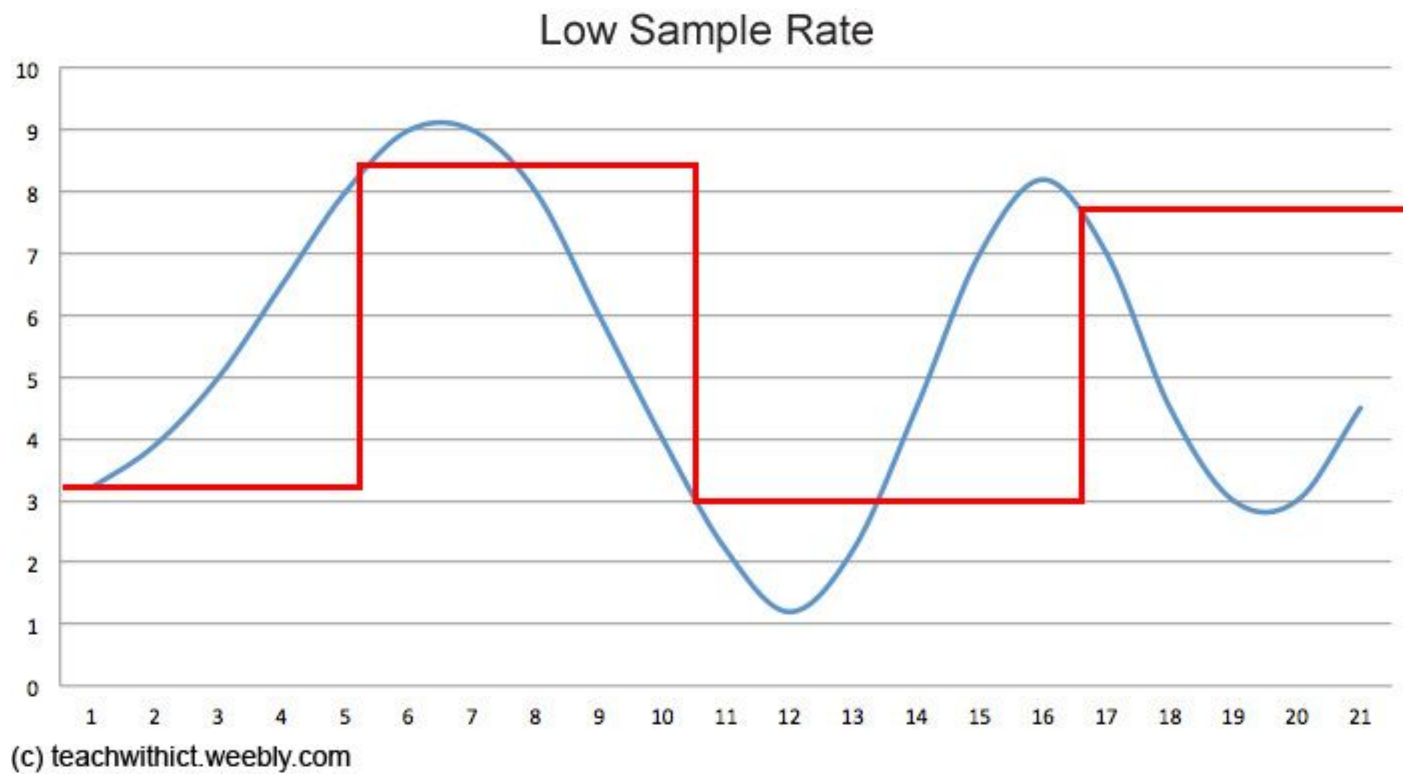
https://www.teachwithict.com/binary
-representation-of-sound.html

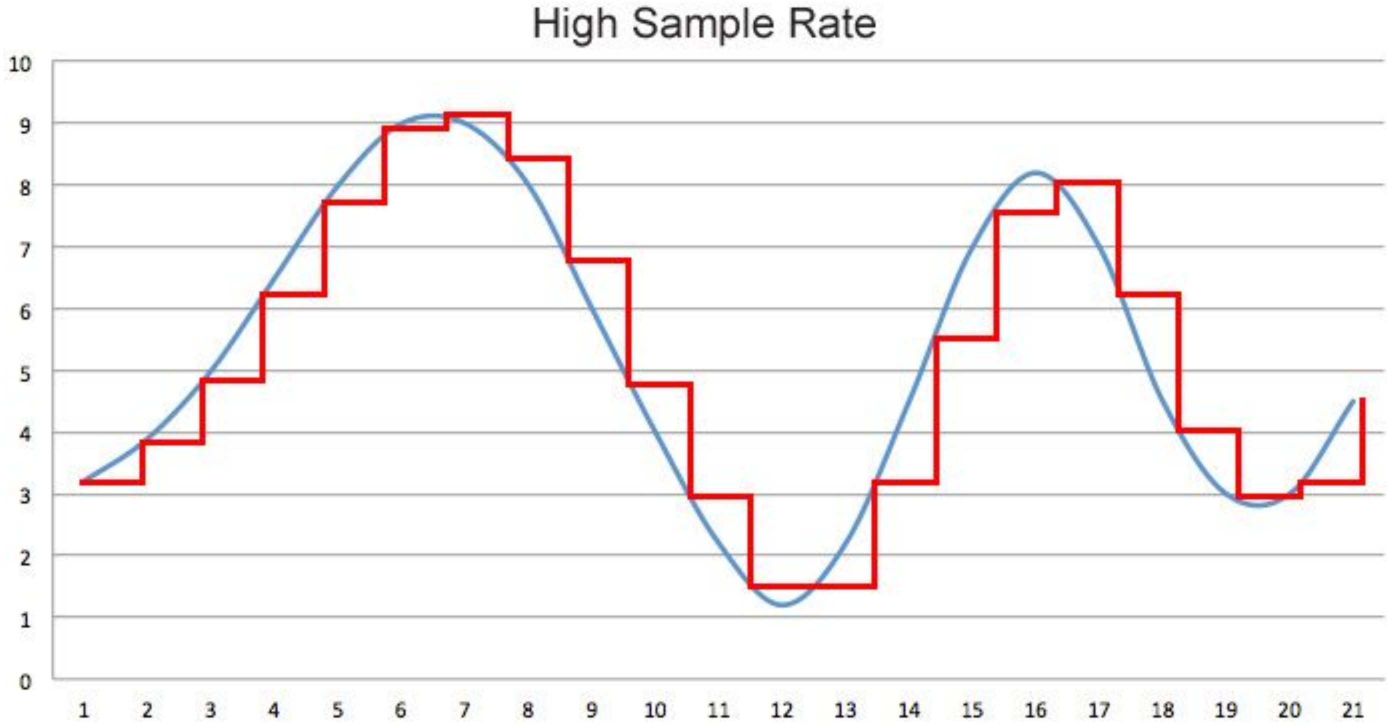

Lower Pitch
Higher Pitch
Low Frequency
High Frequency
Quieter
Louder
Low Amplitude
High Amplitude

Time sampling of a wave



Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Sample	3.2	3.9	5	6.5	8	9	9	8	6	4	2.2	1.2	2.2	4.5	7	8.2	7	4.5	3	3	4.5


Low Sample Rate
10
9
8
7
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
(c) teachwithict.weebly.com


High Sample Rate
10
9
8
7
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21

Next week

How computer works?

- How to represent 0, 1
    - Transistor
- Logical Operations
- Program
- Algorithm
- …