

## Class 5 Recap: Binary

We learned about the binary number system, and how the digits are in **multiples of 2** (1, 2, 4, 8, 16, 32, etc).

Binary is a base-2 number system that's made up of only two numbers or digits: 0 (zero) and 1 (one). This numbering system is the basis for all binary code, which is used to write digital data such as the computer processor instructions used every day in your laptops, phones, computers etc. It is a simple and elegant design. Binary's 0 and 1 method is quick to detect an electrical signal's off (0) or on (1) state.

### Binary (operations)

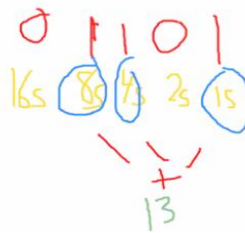
0s and 1s, 1 = True, 0 = False

1 10 100 1000 ..  
1 2 4 8 ...

1010 in binary = 10 in decimal  
101 in binary = 5 in decimal

- a) 111 = 7
- b) 0101 = 5
- c) 11100 = 28
- d) 01101 = 13

$$\begin{array}{r} 742 \\ 1005 \quad 103 \quad 15 \\ \hline = 7 \times 100 + 4 \times 10 + 2 \times 1 \\ = 742 \end{array}$$



We can see in above example how we can visualize the digits of a binary number, we read it from right to left with the multiples getting larger as you go to the left.

Turn the decimals into binary: (Hint: work backwards from turning binary to decimal)

a) 31  $11111$  ✓  
 b) 17  $10001$  ✓  
 c) 6  $110$   
 d) 63  $111111$

Binary operations)

0s and 1s, 1 = True, 0 = False

1 10 100 1000 ...  
 1 2 4 8 ...

1010 in binary = 10 in decimal  
 101 in binary = 5 in decimal

a) 111 = 7  
 b) 0101 = 5  
 c) 11100 = 28  
 d) 01101 = 13

computers cannot handle the number system we know, so they use a system of 0s and 1s to express the number system in their own way, that is called BINARY

whats the biggest binary multiple that can fit into this, recursively

10001

11111  
 16 8 4 2 1  
 11111  
 +  
 31

17  
 - 16  
 ---  
 1

8  
 - 4  
 ---  
 4  
 - 2  
 ---  
 2  
 - 1  
 ---  
 1

1  
 - 1  
 ---  
 0 ✓

In the above example, we can see how we can do the reverse and convert normal numbers (decimals) into binary. We use the subtraction method where we ask what is the biggest binary multiple that can be fitted inside the number. Then we take the remainder result and do the same process again and again until we are left with 0 at the end. We work down one multiple at a time (8 to 4 to 2 to 1 as an example). If we **CANNOT** fit the next binary multiple inside the number/remainder, then we mark that as a 0. If we **CAN**, then we mark that as a 1. (See the yellow box at the top right where I constructed the binary)

## Addition

We learned we can add binary numbers together

Just like in math, you still carry over digits, and if you go to new digits you just assume there are 0s

011011 → 27  
 + 1101 → 13  
 ---  
 100001 → 33

1111 → 15  
 + 11 → 3  
 ---  
 10010 → 18 ✓

In the recap document for this class, I will post more practice operations (including SUBTRACTION)

# Extra Practice

- 1) Convert the following decimal (normal) numbers to binary
  - a. 9
  - b. 21
  - c. 33
  - d. 64
- 2) Convert the following binary numbers back to decimal (normal) numbers
  - a. 10100
  - b. 00101
  - c. 10111
  - d. 100000
  - e. 101101
- 3) Add the following binary numbers, express your final answer as binary AND decimal (normal number)
  - a.  $1101 + 1011$
  - b.  $01011 + 11$
  - c.  $10000 + 111$
  - d.  $110 + 1010$

*I did not teach you subtraction in binary but the principles of carrying AND borrowing are the same as normal math, knowing that just apply the same principles here to solve the subtraction problems. You can do it!*

- 4) Subtract the following binary numbers, express your final answer as binary AND decimal (normal number)
  - a.  $111 - 10$
  - b.  $1011 - 100$
  - c.  $1111 - 111$
  - d.  $10101 - 101$
  - e.  $1000 - 11$