1 Introduction

In this activity you will learn about binary numbers. This activity was based on material developed by Professor Saturnino Garcia of the University of San Diego. It is used here with permission.

Before you begin, take a minute to assign roles to each member in your group. Try to switch up the roles as much as possible: please don't pick a role for yourself that you have already done more than once. Below is a summary of the four roles; write the name of the person taking that role next to the summary.

If your group only has three members, combine the roles of Facilitator and Process Analyst.

For this and all future activities, the Facilitator should also take the role of the reader and read the questions aloud to the group.

Facilitator Reads question aloud; keeps track of time and makes sure everyone contributes appropriately.

Quality Control Records all questions and answers, and provides team reflection to team and instructor.

Spokesperson Talks to the instructor and other teams. Compiles and runs programs when applicable.

Process Analyst Considers how the team could work and learn more effectively.

Fill in the following table showing which group member is performing each role:

Role	Person
Facilitator	
Quality Control	
Spokesperson	
Process Analyst	

2 Model 1: Binary Number Representation

Questions: 11 | Allocated Time: 15 minutes

We have all been trained to think about numbers in terms of a decimal system. We can look at a number like 872 and break it down into ones, tens, and hundreds places and almost instantly understand the quantity with which we are dealing. Unlike humans, computers don't tend to have ten fingers so they use a different representation of numbers: binary representation. Binary is very similar to decimal in many respects with the most obvious difference being that binary systems use only two characters (0 and 1) instead of 10 (0–9).

The following table shows the binary values for the integers zero to eleven.

Decimal	Binary	Decimal	Binary
0	0	6	110
1	1	7	111
2	10	8	1000
3	11	9	1001
4	100	10	1010
5	101	11	1011

Problem 1. How many *binary digits* (a.k.a. *bits*) are used in the representation of the following decimal numbers:

2 2

5 3

7 3

10 4

Problem 2. List the decimal numbers (between zero and eleven) where the binary representation requires one more *binary digit* (i.e. *bit*) than the previous number.

2 4 8

Problem 3. For the numbers you just listed, give the *binary representation* of the number that came before it. For example, if 4 (i.e. 100) was one of your answers, give the binary representation of the number 3 (i.e. 11).

2 1

4 11

8 111

Problem 4. What is the **smallest** number that will *require* a certain number of bits? Give your answer in decimal representation.

5 bits 16

6 bits 32

7 bits 64

Problem 5. Let X be the first number to require N bits. In terms of X, what will be the first number to require $N + 1$?

It will be $2X$.

Problem 6. Describe the pattern of the *rightmost* bit as you count up from zero to ten.

Alternating: 0, 1, 0, 1, 0, 1, 0, 1, . . .

Problem 7. Describe the pattern of the *second* rightmost bit as you count up from zero to ten. To help you see the pattern, add a 0 to the left of both zero and one, making them 00 and 01.

Two zeroes, then two ones, repeat: 0, 0, 1, 1, 0, 0, 1, 1, . . .

Problem 8. **How often** does the *third* rightmost bit transition from 0 to 1 and from 1 to 0? Again, add 0's to the left of the binary representation to get enough bits if needed.

It changes after every fourth number: 0, 0, 0, 0, 1, 1, 1, 1, . . .

Problem 9. Complete the following table to show the binary representation of the numbers 11 to 15, using your knowledge of binary patterns you explored in the previous questions.

Decimal	Binary
11	1011
12	1100
13	1101
14	1110
15	1111

Problem 10. The decimal number 64 is represented as 1000000 in binary. Fill in the following table with the binary representation of the given numbers, again using the patterns you have discovered up to this point.

Decimal	Binary
64	1000000
65	1000001
66	1000010
67	1000011
68	1000100

Problem 11. *(Advanced)* Imagine that you start a binary counter at 0 and then add 1 to it over and over again. Fill in the following table to indicate **how often** each bit position transitions from 0 to 1 or 1 to 0 (i.e. the frequency of transitions).

Bit	Transition Frequency
Rightmost	After every increment
2nd rightmost	After every other increment
3rd rightmost	" " 4th "
4th rightmost	" " 8th "
5th rightmost	" " 16th "
Nth rightmost	" " 2^Nth "

TIME REQUIRED:

3 Model 2: The Value of a Binary Number

Questions: 12 | Allocated Time: 20 minutes

The following example illustrates how we convert a decimal number to a value:

Digit	8	9	5	9
Index	3	2	1	0
Weight	10^3	10^2	10^1	10^0

$$8959 = 8 \times 10^3 + 9 \times 10^2 + 5 \times 10^1 + 9 \times 10^0$$

Problem 12. What *weighted* value is associated with the digit at index 2?

100

Problem 13. Does adding a `0` to the *left* of a decimal number (i.e. at the beginning) change its value? For example, changing 759 to 0759. Explain.

No. $0 \times 10^N = 0$ for any N.

Problem 14. Does adding a `0` to the *right* of a decimal number (i.e. at the end) change its value? For example, changing 759 to 7590. Explain.

Yes, it has the effect of multiplying the number by 10. This is because all the digits of the old number are shifted to the left by one place, so their weights are all multiplied by 10.

Problem 15. Complete the following table that will help convert a binary number to its equivalent decimal value.

Bit	1	1	0	0	1
Index	4	3	2	1	0
Weight	2^4	2^3	2^2	2^1	2^0

Problem 16. What is the decimal representation of the binary number in the previous question (i.e. 11001)?

$$1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 16 + 8 + 1 = 25$$

Problem 17. Does adding a `0` to the *left* of a binary number (i.e. at the beginning) change its value? For example, changing 101 to 0101. Explain.

No, for the same reason as for decimal numbers (except now it's 0×2^N).

Problem 18. Does adding a `0` to the *right* of a binary number (i.e. at the end) change its value? For example, changing 101 to 1010. Explain.

Yes, for the same reason as for decimal numbers (except now it's equivalent to multiplying by 2, rather than 10).

Problem 19. Convert the following binary numbers to decimal:

- 100101 = 37
- 111011 = 59
- 1000001 = 65

Problem 20. We can calculate the value of an m digit decimal number according to the following equation:

$$\sum_{i=0..m} d_i \times 10^i$$

Where d_i is the digit at index i (e.g. d_3 in the example above is 8). Write a similar equation, except to handle binary instead of decimal numbers.

$$\sum_{i=0..m} d_i \times 2^i$$

Problem 21. One side effect of the equation you wrote in the last question is that any value can be represented as a sum of powers of 2. For example, 3 is $2^1 + 2^0$ (i.e. 2 + 1) What powers of 2 add together to equal the following numbers.

- 4 = 2^2
- 6 = $2^2 + 2^1$
- 7 = $2^2 + 2^1 + 2^0$
- 20 = $2^4 + 2^2$
- 48 = $2^5 + 2^4$

Problem 22. Fill in the following table of bits for the decimal number 48.

Bit	1	1	0	0	0	0	0
Index	6	5	4	3	2	1	0
Weight	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Problem 23. *(Advanced)* While it might be easy to remember combinations of the powers of 2 for smaller numbers, this approach does not scale well. Briefly describe an algorithm you could use to determine which powers of 2 add to equal a given decimal value.

Find the largest power of two that is smaller than the value; that power will be used. Subtract it from the value. Repeat with the smaller value, until you reach zero.

Problem 24. *(Advanced)* `unsigned char` is an integer datatype in C that is usually represented as a single *byte*, where a byte is defined as 8 bits. What is the maximum value of an `unsigned char`, in both binary and decimal representations?

Binary Representation 1111 1111

Decimal Representation 255

TIME REQUIRED:

3.1 Group Reflection

Problem 25. What is the range of numbers we can represent with a N bit binary number?

Zero through 2^{N-1}.

Problem 26. Convert the binary number 1110011 into its equivalent decimal representation.

115

Problem 27. Convert the decimal number 99 into its equivalent binary representation.

1100011

4 Model 3: Hexadecimal Representation

Questions: 13 | Allocated Time: 15 minutes

One of the downsides of binary numbers is that they require more digits to represent a number than decimal. Hexadecimal numbers are a sort of middle ground between binary and decimal numbers. Below is a table of the hexadecimal representation of the integers 0 to 19.

D	H	D	H	D	H	D	H
0	0	5	5	10	A	15	F
1	1	6	6	11	B	16	10
2	2	7	7	12	C	17	11
3	3	8	8	13	D	18	12
4	4	9	9	14	E	19	13

Note that we'll often use the C notation of adding a `0x` prefix to a hexadecimal number to differentiate it from decimal (e.g. 150 vs 0x150).

Problem 28. What possible values can a single hexadecimal digit have? For example, a single bit can have the value 0 or 1, while a single decimal digit can have values between 0 and 9.

From 0 to 15

Problem 29. How many distinct values can a single hexadecimal digit represent? For example, a decimal digit can have 10 distinct values while a binary digit (i.e. bit) can have 2 distinct values.

16

Problem 30. What is the *weight* of the *rightmost* hex digit? Recall that the rightmost decimal digit had a weight of 10^0.

$16^0 = 1$.

Problem 31. What is the weight of the hex digit just to the left of the rightmost digit (i.e. the second rightmost)?

$16^1 = 16$.

Problem 32. Give the hexadecimal and decimal representations of the first number to *require* 3 hex digits.

- Hexadecimal: 0x100
- Decimal: 256

Problem 33. Which representation will be more compact (i.e. require fewer digits) for large numbers: hexadecimal or decimal?

Hexadecimal, since each digit can represent more possible values.

Problem 34. What is the hexadecimal representation of the decimal number 21?

0x15

Problem 35. What is the hexadecimal representation of 32?

0x20

Problem 36. Complete the following table by converting the hex numbers into their equivalent decimal representation:

Hex.	Dec.
0x30	48
0x5A	90
0x11D	285

Problem 37. One benefit of hexadecimal representation is the ease with which we can convert between it and binary representation. Complete the table below, showing the conversion between a hex digit and binary.

Dec.	Hex.	Bin.	Dec.	Hex.	Bin.	Dec.	Hex.	Bin.
0	0x0	0000	6	0x6	0110	12	0xC	1100
1	0x1	0001	7	0x7	0111	13	0xD	1101
2	0x2	0010	8	0x8	1000	14	0xE	1110
3	0x3	0011	9	0x9	1001	15	0xF	1111
4	0x4	0100	10	0xA	1010	16	0x10	10000
5	0x5	0101	11	0xB	1011	17	0x11	10001

Problem 38. There is a trivial algorithm to convert from hex to binary: just convert each hex digit to its equivalent 4-bit binary representation. For example, 0xA1 is 1010 (A) followed by 0001 (1), i.e. 10100001. What is the binary representation of the following hex numbers?

- $0xCAFE$ = 1100 1010 1111 1110
- $0x8BADF00D$ = 1000 1011 1010 1101 1111 0000 0000 1101

Problem 39. Devise an algorithm for converting any binary number into its equivalent hexadecimal representation. Your algorithm should be a direct conversion: you should **NOT** convert to any intermediate representation (e.g. decimal).

Divide the binary number into groups of four bits (sometimes known as "nybbles") and look each group up in the table at the beginning of this model.

Problem 40. Convert the following binary numbers into their hexadecimal representation:

- 1101 0001 1110 = $0xD1E$
- 110111 = 0011 0111 = $0x37$

TIME REQUIRED:

4.1 Group Reflection

Problem 41. We just saw that any whole number can be represented as a sum of powers of 2, 10, or 16. What about other "bases"? Are there any whole numbers n that *won't work* as a base—that is, you *cannot* represent any whole number as a sum of powers of n?

Zero and one won't work, because $0^n = 0$ and $1^n = 1$ for any n.

Problem 42. *(Advanced)* What if we tried to use powers of -2? Would it still be possible to represent any whole number? Are there numbers that can be represented as a sum of powers of -2 but that *can't* be represented as a sum of powers of 2?

This is called "nega-binary" and it can represent all *integers*, not just all whole numbers. For example: $101_{-2} = 5_{10}$ and $1111_{-2} = -5_{10}$.

5 Model 4: Adding Binary Numbers

Questions: 15 | Allocated Time: 15 minutes

The rules of binary addition are very similar to the rules you learned in elementary school for adding decimal numbers.

Problem 43. Perform the following decimal addition using the trusty gradeschool algorithm of adding digit-by-digit, carrying over from one column to the next when necessary. Make sure you write down all your work (including carries).

				1	
	8	1	5	2	9
+	1	2	2	5	6
	9	3	7	8	5

Problem 44. What is the maximum number that can result from adding two 1 decimal digit numbers together?

$9 + 9 = 18$

Problem 45. As a result of your answer to the previous question, decimal addition occasional requires that we *carry* a value from one column to the next. What is the maximum carry value?

1, since the largest possible sum of two decimal digits is still less than 20.

Problem 46. Complete the following table to indicate the result of adding two 1 bit numbers (b_1 and b_2). The result of addition should also be written in binary notation.

b_0	b_1	$b_0 + b_1$
0	0	0
0	1	1
1	0	1
1	1	10

Problem 47. What is the maximum number of bits required to represent the value of adding two 1-bit numbers?

Two (1 + 1 = 10).

Problem 48. Like decimal addition, binary addition includes the possibility of having to *carry* a 1 value over to the next column and therefore have to add three values together instead of two. Complete the following table to show all possible combinations of adding two 1-bit numbers (b_0 and b_1) with a carry bit (c).

c	b_0	b_1	$c + b_0 + b_1$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	10
1	0	0	1
1	0	1	10
1	1	0	10
1	1	1	11

Problem 49. What is the maximum number of bits required to represent the value of adding two 1-bit numbers and a carry?

Still just two; 1 + 1 + 1 = 11 (= 3).

Problem 50. Apply what you have learned from the previous questions to add the following two binary numbers together.

			1	
	1	0	0	1
+	0	1	0	1
	1	1	1	0

Problem 51. Convert the three binary numbers from the previous question (i.e. the two numbers being added together and the result) to their decimal representation. Was the result what you expected?

$1001_2 = 9_{10}$, $0101_2 = 5_{10}$, $1110_2 = 14_{10}$, as expected.

Problem 52. Add the following two binary numbers together.

1				
	1	0	1	0
+	1	1	0	0
1	0	1	1	0

Problem 53. How many bits were required for the *result* in the previous question?

Five

Problem 54. How does the number of bits needed for the result compare to the number of bits in the numbers you added together?

One more than the number of bits in the addends.

Problem 55. Come up with an equation that describes the maximum number of bits required to hold the result of adding two N-bit numbers together.

It's always at most $N + 1$, since the "carry out" from the leftmost pair of bits in the addends cannot be more than 1.

Problem 56. What happens if you only have N bits to hold the result of two N-bit numbers?

Most computers just throw away the $N + 1$'th bit! Mathematically, this is the same as reducing the result modulo 2^N.

TIME REQUIRED:

5.1 Group Reflection

Problem 57. Convert the hexadecimal number 0x6FA into its equivalent binary representation.

$0x6FA = 0110\,1111\,1010$

Problem 58. Convert the binary number 1111011 to its equivalent hex representation.

$0111\,1011 = 0x7B$

6 Model 5: Representing Negative Values in Binary

Questions: 20 | Allocated Time: 20 mins

The binary representation we have seen thus far is exactly the representation used in C for "unsigned" data types (e.g. `unsigned int`). The downside of this representation—and consequently unsigned numbers—is that we can only represent non-negative integers. In this model we'll look at how to represent *both* positive *and* negative integers.

In our decimal system a negative number is indicated by the presence of a special symbol, the "minus sign." In our binary system we do not have the luxury of adding another symbol: we are stuck with just `0` and `1`. There are several different ways to encode negative integers in binary, but we will focus on the most common choice, called *two's complement*.

The two's complement representation is capable of representing both positive and negative numbers. Getting the two's complement representation of a non-negative number is a simple two step process.

- **Step 1:** Start by finding the binary representation of the *magnitude* of the number (like we've done before).
- **Step 2:** Place a 0 to the left of the number.

For example, to get the two's complement representation of 5 we first convert to binary like we did before. This gives us 101. Next, we place a 0 to the left of 101, giving us 0101. That's it!

Problem 59. What is the leftmost bit in a non-negative two's complement number?

Always zero.

Problem 60. What is the *two's complement* representation of the following numbers?

- 32 = 0 10 0000
- 42 = 0 10 1010

The process of getting the two's complement representation of a negative number is slightly more complex.

- **Step 1:** Start by finding the binary representation of the *absolute value* of the number (like we've done before).
- **Step 2:** Place a 0 to the left of the number. This gives you the **positive** representation of the number.
- **Step 3:** Invert all the bits ($0 \to 1$, and $1 \to 0$).
- **Step 4:** Add 1. If this addition has a "carry out" from the leftmost pair of bits, discard it. This gives you the **negative** representation of the number.

You might have noticed that the first two steps are the same for both negative and non-negative values, so it's really just two more steps to go from non-negative to negative.

The following table shows the decimal numbers 0 to 10, their magnitude representation, their positive binary representation, the positive number with bits inverted, and the negative representation (i.e. inverted + 1). These columns correspond to steps 1 to 4 from above.

Decimal	Magnitude	Positive	Invert	Negative
0	0	00	11	00
1	1	01	10	11
2	10	010	101	110
3	11	011	100	101
4	100	0100	1011	1100
5	101	0101	1010	1011
6	110	0110	1001	1010
7	111	0111	1000	1001
8	1000	01000	10111	11000
9	1001	01001	10110	10111
10	1010	01010	10101	10110

Problem 61. What is the two's complement representation of the following decimal numbers?

- 3 = 011

- 8 = 11000

Problem 62. Based on the table above, is there an easy way to determine the sign of a two's complement number?

If the leftmost bit of a two's complement number is 0, the number is non-negative. If it is 1, the number is negative.

Problem 63. Follow the four steps above to find the representation(s) of -15 in two's complement.

- Step 1: 1111
- Step 2: 0 1111
- Step 3: 1 0000
- Step 4: 1 0001

Problem 64. Fill in the following table by performing steps 3 and 4 from above (i.e. invert bits and add 1) but this time *starting with* the **negative** representation.

Decimal	Negative	Invert	Add 1
1	11	00	01
2	110	001	010
3	101	000	011
4	100	011	100
5	1011	0100	0101

Problem 65. Compare the "Add 1" column of this table with the table of decimal numbers from 0 to 10, above. Do you notice any correspondence between them?

The "Add 1" column of this table is the same as the "Positive" column of the earlier table.

Problem 66. What is the decimal representation of the following two's complement binary numbers:

- 010111 = 23_{10}
- 111010 = -6_{10}

Problem 67. Complete the following table by converting the two's complement representation to the equivalent decimal representation.

Two's Comp	Decimal
011	3
0011	3
00011	3
101	−3
1101	−3
11101	−3

Problem 68. How would you make an N-bit two's complement number into an $(N+1)$-bit two's complement number *without* changing its value?

Add one bit to the left that is the same as the sign bit (leftmost bit) of the original number.

Problem 69. Some of the entries in the "Negative" column in the first table have more bits than *required* to represent that number. Identify these entries and give the minimal two's complement representation (i.e. minimize the number of bits without changing the value that it represents).

Entries 0, 1, 2, 4, and 8 have more bits than required to represent that number. The minimal two's complement representation is as follows:

Decimal	Bits req'd	Negative
0	0	''
1	1	1
2	2	10
4	3	100
8	4	1000

Problem 70. Complete the following table to indicate the most positive (i.e. largest) and most negative (i.e. smallest) integer that can be represented with a given number of bits *when using two's complement representation*. Use the table above (which you simplified by removing unnecessary bits) to help you answer this question.

Bits	Most Positive	Most Negative
1	0	−1
2	1	−2
3	3	−4
4	7	−8

Problem 71. Use your answer from the previous question to find an expression that gives the *most positive* integer that can be represented by a N-bit two's complement number. Hint: This will be related to a power of two in some way.

The most positive integer that can be represented by a N-bit two's complement number is $2^{N-1} - 1$.

Problem 72. Use your answer from the previous questions to find an expression that gives the *most negative* integer that can be represented by a N-bit two's complement number. Hint: This will be related to a power of two in some way.

The most negative integer that can be represented by a N-bit two's complement number is -2^{N-1}.

Problem 73. In general, given an N-bit two's complement number, how does the *magnitude* of the most positive integer it can represent compare with the *magnitude* of the most negative integer it can represent? Why might this be?

The magnitude of the most positive integer that can be represented is always one less than the magnitude of the most negative integer that can be represented. This is because of zero. The range of integers represented with a leading 1-bit is $[-2^N, -1]$, and the range of integers represented with a leading 0-bit is $[0, 2^{N-1} - 1]$; there are exactly 2^{N-1} of each.

Problem 74. *How many* distinct integers can be represented by an N-bit two's complement representation (i.e. what is the difference between the most positive and the most negative)?

2^N distinct integers can be represented.

Problem 75. Perform binary addition on the two's complement numbers below:

$$\begin{array}{rr} & 1001 \\ + & 0011 \\ \hline & 1100 \end{array}$$

Problem 76. What are the decimal representations of your inputs (1001 and 0011) and of your solution from the previous question? Is this the answer you expected?

$1001_2 = -7_{10}$, $0011_2 = 3_{10}$, $1100_2 = -4_{10}$, as expected. Notice that if 1001 and 1100 are interpreted as +9 and +12 instead of −7 and −4, the sum is still correct. Binary addition of two's complement numbers is the *same operation* as binary addition of unsigned numbers.

Problem 77. Devise an algorithm for performing *binary subtraction* but using only addition.

Negate the subtrahend (the value on the right side of the binary −) and then add the two numbers as if they were unsigned, discarding the carry (if any).

Problem 78. An alternative to two's complement is *sign-magnitude* representation. In this representation, the leftmost bit of the number still indicates the sign, but the remainder of the bits directly represent the magnitude of the number. To negate a sign-magnitude number, you invert *only* the sign bit, leaving all the other bits unchanged. Negation is simpler this way, but can you see any potential drawbacks to this representation? There must be a reason nobody uses it anymore, right?

Addition of sign-magnitude numbers is not the same operation as addition of unsigned numbers; it is more complicated to deal with signed overflow; and perhaps most importantly, sign-magnitude numbers have *two* ways to represent zero ("positive" and "negative" zero).

TIME REQUIRED:

6.1 Group Reflection

Problem 79. Add the following two binary numbers together. Confirm that the result is correct by converting all the binary numbers to their decimal representation.

	1111000		120
+	0100111	+	39
	1001111		159

Problem 80. Give the decimal representation of the following two's complement binary numbers:

- 11 1111 1111 = −1
- 01 1010 0010 = 418

Problem 81. Give the two's complement representation of the following binary numbers:

- 105 = 0110 1001
- −482 = 10 0001 1110

Problem 82. What is the range of numbers that can be represented by a 32-bit two's complement number?

-2^{31} to $+2^{31} - 1$. In decimal: $-2,147,483,648$ to $2,147,483,647$.