CS246: Programming Paradigms
Prof. Richard Eisenberg
Spring 2017

Bits!

Convert to binary:

1. 9
2. 17
3. 32
4. 0xA
5. 0x6DEA12
6. 07326

Convert from hexadecimal to octal and from octal to hexadecimal:

7. 0xAB38
8. 0x3333
9. 0xCAFE
10. 0275535
11. 0100000
12. 0777777

Assume we can write binary numbers like this: 0b10011, for example, would be 19. (Other languages indeed allow this notation. And `gcc` also supports it as an extension to the language. But it's not in standard C.) Write what each of the following expressions equal. Use the same numerical base that the input (the part being shifted, if there is a shift) is given in.

13. 0b0101 | 0b1010
14. 0b0101 & 0b1010
15. 0b1101 & 0b11
16. 0b1000 | 0b0010
17. 0xAB & 0x78
18. 0x1 | 0xC
19. 0b1011 ^ 0b0110
20. 0b1010 ^ 0b1111
21. $0b011000 \ll 1$
22. $0b011000 \gg 2$
23. $0x2 \ll 2$
24. $0x3 \ll 2$

Interpret the following bit patterns as 8-bit `signed char`s. Circle the larger number.

25. 0b00001000
    0b00000100

26. 0b00000000
    0b11111111

27. 0b00001111
    0b11110000

28. 0b11111110
    0b11111111

Let X be a 4-bit quantity. Solve for X If there are multiple solutions, any one solution will do.

29. $0b1100 \mid X = 0b1101$

30. $X \ll 2 = 0b0100$

31. $0b1101 \,\&\, X = 0b1001$