Base-b & Fast Exponentiation

Review

1. Is $\gcd(11, 52) = 1$?
2. Solve $1 = s(11) + t(52)$ for s and t
3. Then s is the inverse of 11 mod 52

Let's review the Extended Euclidean Algorithm

Find the inverse of 11 mod 52

First, find $\gcd(11, 52)$:

$52 = 11(4) + 8$
$11 = 8(1) + 3$
$8 = 3(2) + 2$
$3 = 2(1) + \mathbf{1}$

$\gcd(11,52) = 1$

Next, rearrange each equation:

$8 = 52 - 4(11)$
$3 = 11 - 1(8)$
$2 = 8 - 2(3)$
$1 = 3 - 1(2)$

Finally, use substitution to solve $1 = s(11) + t(52)$ for s and t.

$1 = 3 - 1\big(8 - 2(3)\big)$	substitute 2
$1 = 3 - 1(8) + 2(3)$	simplify
$1 = 3(3) - 1(8)$	simplify
$1 = 3\big(11 - 1(8)\big) - 1(8)$	substitute 3
$1 = 3(11) - 3(8) - 1(8)$	simplify
$1 = 3(11) - 4(8)$	simplify
$1 = 3(11) - 4\big(52 - 4(11)\big)$	substitute 8
$1 = 3(11) - 4(52) + 16(11)$	simplify
$1 = 19(11) - 4(52)$	simplify

The inverse of 11 mod 52 is **19**.

Base-b

We can express any integer n using any base-b where $b > 1$

$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b^1 + a_0 b^0$ $\quad$ $523 = 5 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0$

$(a_k a_{k-1} \ldots a_1 a_0)_b$ $\quad$ base b expansion of n $\quad$ 523_{10}

Binary: $(10110100)_2 = 1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2 + 0 = 180$ (decimal)

Octal: $(264)_8 = 2 \cdot 8^2 + 6 \cdot 8^1 + 4 = 180$ (decimal)

Hexadecimal: $(\mathrm{B}4)_{16} = \mathrm{B} \cdot 16^1 + 4 = 11 \cdot 16 + 4 = 176 + 4 = 180$ (decimal)

Hexadecimal Numbers

Base-16 is called Hexadecimal or Hex

Hexadecimal uses 16 "digits" or symbols

Each hex digit corresponds to a unique sequence of 4 binary bits.

Two hex digits represent 8 binary bits, a byte.

$10110111 \ = \ \mathrm{B7}$

$10110111 \ = \ \mathrm{B7} = 11 \cdot 16 + 7 = 183_{10}$

$10110111 = 1 \cdot 2^7 + 1 \cdot 2^5 + 1 \cdot 2^4 + +1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 183_{10}$

What is 3F?

Binary: 00111111

Decimal: $3 \cdot 16 + 15 = 63_{10}$

Decimal	Hex	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Convert base-b to n

$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b^1 + a_0 b^0$

Convert each of the following to base-10 (without using a calculator):

$$
\begin{aligned}
1101011_2 &= 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 \\
&= 64 + 32 + 8 + 2 + 1 \\
&= 107
\end{aligned}
$$

$$
\begin{aligned}
1234_5 &= 1 \cdot 5^3 + 2 \cdot 5^2 + 3 \cdot 5^1 + 4 \cdot 5^0 \\
&= 125 + 50 + 15 + 4 \\
&= 194
\end{aligned}
$$

$$
\begin{aligned}
1221_3 &= 1 \cdot 3^3 + 2 \cdot 3^2 + 2 \cdot 3^1 + 1 \cdot 3^0 \\
&= 27 + 18 + 6 + 1 \\
&= 52
\end{aligned}
$$

Convert n to base$_b$

1. Divide n by b to get a quotient q and remainder a_0:

   $n = bq_0 + a_0$

2. The remainder is the rightmost "digit" in the base_b expansion of n

3. Divide the quotient q_0 by b to obtain a new quotient q_1 and remainder a_1.

   $q_0 = bq_1 + a_1$

4. The remainder a_1 is the next rightmost "digit".

5. Continue the process until the quotient is 0.

Example:
97_{10} to base$_3$

10121_3

$$97 = 3 \cdot (32) + (1)$$
$$32 = 3 \cdot (10) + (2)$$
$$10 = 3 \cdot (\ 3) + (1)$$
$$3 \quad = 3 \cdot (\ 1) + (0)$$
$$1 \quad = 3 \cdot (\ 0) + (1)$$

Rosen, Kenneth. *Discrete Mathematics and Its Applications, 7th Edition*. McGraw-Hill, 2012

Try it!

Convert 215_{10} to base-5

$215 = 5(43) + 0$
$43 = 5(8) + 3$
$8 = 5(1) + 3$
$1 = 5(0) + 1$

1330

Convert 57_{10} to base-16

$57 = 16(3) + 9$
$3 = 16(0) + 3$

39

Example:
97_{10} to base_3

$97 = 3 \cdot (32) + (1)$

$32 = 3 \cdot (10) + (2)$

$10 = 3 \cdot (3) + (1)$

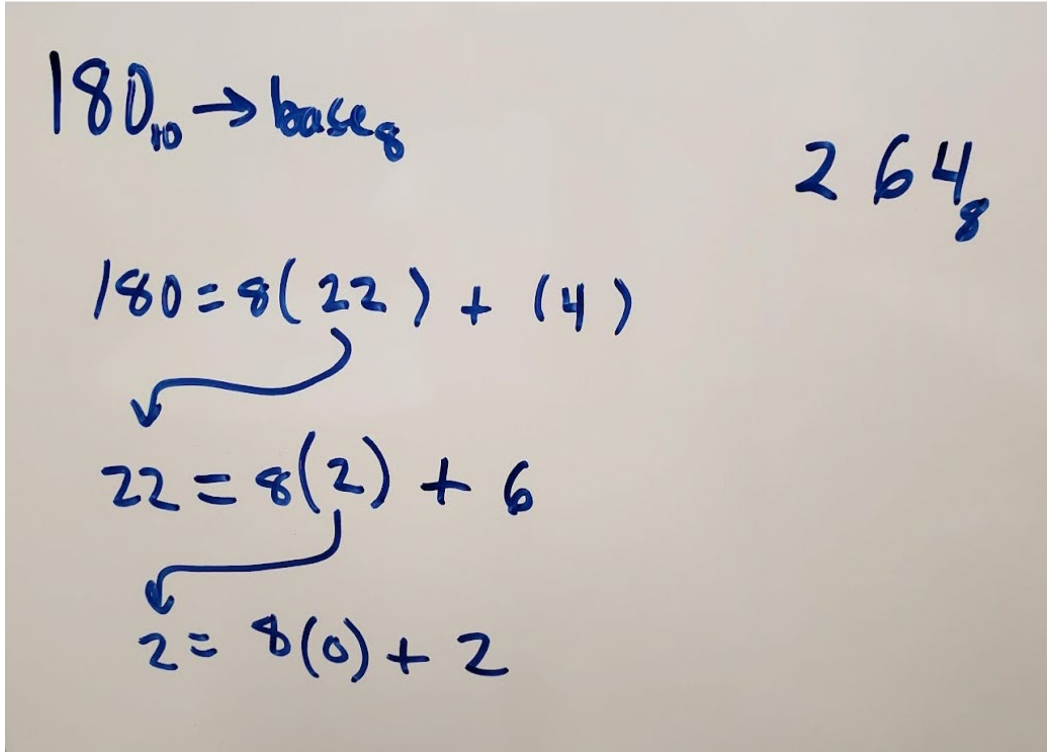
$3 = 3 \cdot (1) + (0)$

$1 = 3 \cdot (0) + (1)$

10121

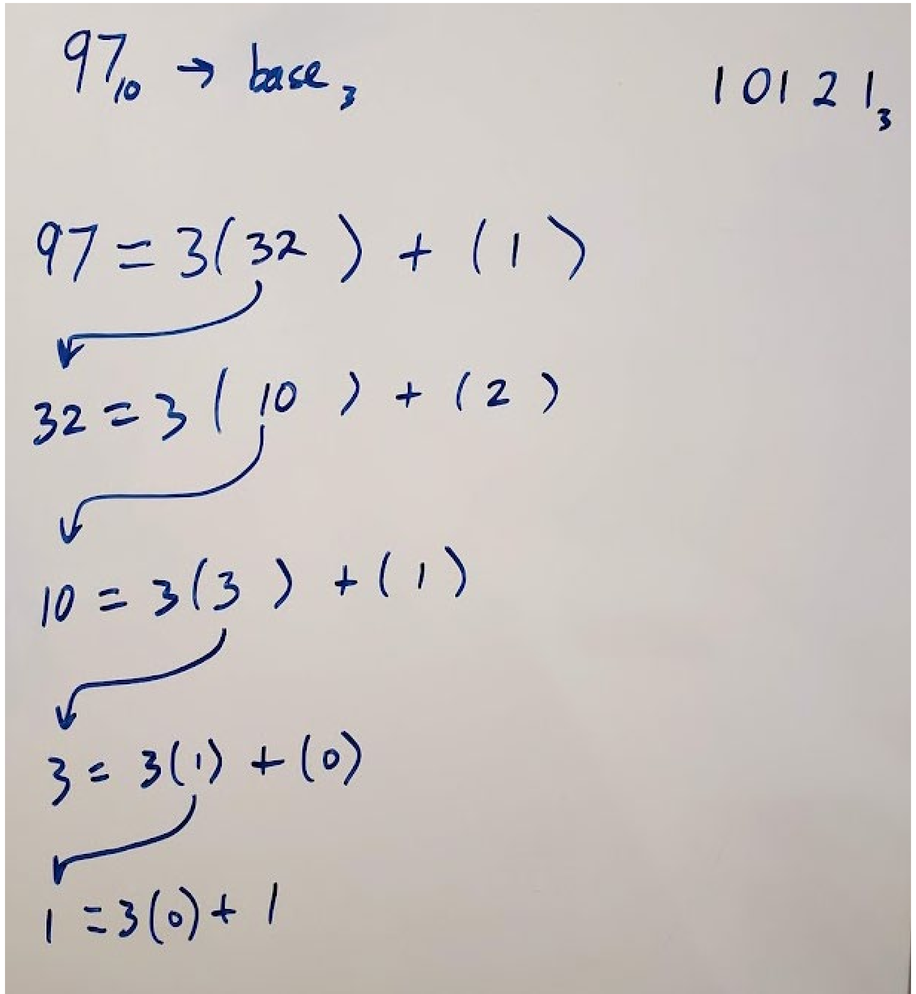
Try it:

180_{10} to base-8



Try it:

97_{10} to base-3



Additional Exercises:

9.6.1 (a-d)

9.6.2 (a-d)

Fast Modular Exponentiation

In cryptography, it is important to find $\boldsymbol{b^n}$ **mod** $\boldsymbol{m}$ quickly and efficiently, with b, n, and m being large integers.

b^n is a HUGE number, so it is not practical to compute it directly, then divide it by m to find the remainder. (n is typically 1024 or 2048 bits. That is between 308 and 617 digits!)

Instead, we use the fast modular exponentiation algorithm, which uses binary expansion of the exponent n

This makes computing x where $x \equiv b^n (\text{mod}\ m)$ very fast and efficient.

However, reversing this process and finding n if we know $x, b,$ and m is very difficult. This is the basis of some cryptographic algorithms.

Modular exponentiation

We can quickly compute $\boldsymbol{b^n}$ **mod** $\boldsymbol{m}$ if n is a power of 2 because:

$$b^2 \bmod m = (b * b) \bmod m = (b \bmod m) * (b \bmod m) \bmod m$$

Compute $7^{256} (\text{mod } 13)$:

$7 \bmod 13 = 7$

$7^2 \bmod 13 = (7 * 7) \bmod 13 = 49 \bmod 13 = 10$

$7^4 \bmod 13 = (7^2 * 7^2) \bmod 13 = (7^2 \bmod 13 \; * 7^2 \bmod 13) \bmod 13 = (10 * 10) \bmod 13 = 9$

$7^8 \bmod 13 = (7^4 * 7^4) \bmod 13 = (9 * 9) \bmod 13 = 81 \bmod 13 = 3$

$7^{16} \bmod 13 = (7^8 * 7^8) \bmod 13 = (3 * 3) \bmod 13 = 9 \bmod 13 = 9$

$7^{32} \bmod 13 = (7^{16} * 7^{16}) \bmod 13 = (9 * 9) \bmod 13 = 3$

$7^{64} \bmod 13 = (7^{32} * 7^{32}) \bmod 13 = (3 * 3) \bmod 13 = 9$

$7^{128} \bmod 13 = (7^{64} * 7^{64}) \bmod 13 = (9 * 9) \bmod 13 = 3$

$7^{256} \bmod 13 = (7^{128} * 7^{128}) \bmod 13 = (3 * 3) \bmod 13 = 9$

https://www.khanacademy.org/computing/computer-science/cryptography/modarithmetic/a/fast-modular-exponentiation

If n is not a power of 2, we convert it into multiples of powers of 2:

$$b^n = b^{a_{k-1}*2^{k-1}+a_{k-2}*2^{k-2}+\ldots+a_1 2^1+a_0} = b^{a_{k-1}*2^{k-1}} * \cdots * b^{a_1*2} * b^{a_0}$$

Example:

b^{117}

$117 = 1\ 1\ 1\ 0\ 1\ 0\ 1_2$

64 32 16 4 1

$117 = 64 + 32 + 16 + 4 + 1$

$b^{117} = b^{(64+32+16+4+1)} = b^{64} * b^{32} * b^{16} * b^4 * b$

Now we can find something like $5^{117} \bmod 19$

Rosen, Kenneth. *Discrete Mathematics and Its Applications, 7th Edition*. McGraw-Hill, 2012
https://www.khanacademy.org/computing/computer-science/cryptography/modarithmetic/a/fast-modular-exponentiation

$$b^2 \bmod m = (b * b) \bmod m = (b \bmod m) * (b \bmod m) \bmod m$$

$5^{117} \bmod 19$

$5^{117} \bmod 19 = 5^{64+32+16+4+1} \bmod 19 = 5^{64} * 5^{32} * 5^{16} * 5^{4} * 5^{1} \bmod 19$

$5^{1} \bmod 19 = \boxed{5}$

$5^{2} \bmod 19 = (5 * 5) \bmod 19 = 25 \bmod 19 = 6$

$5^{4} \bmod 19 = (5^2 * 5^2) \bmod 19 = (6 * 6) \bmod 19 = 36 \bmod 19 = \boxed{17}$

$5^{8} \bmod 19 = (5^4 * 5^4) \bmod 19 = (17 * 17) \bmod 19 = 289 \bmod 19 = 4$

$5^{16} \bmod 19 = (5^8 * 5^8) \bmod 19 = (4 * 4) \bmod 19 = 16 \bmod 19 = \boxed{16}$

$5^{32} \bmod 19 = (16 * 16) \bmod 19 = 256 \bmod 19 = \boxed{9}$

$5^{64} \bmod 19 = (9 * 9) \bmod 19 = 81 \bmod 19 = \boxed{5}$

$5^{117} \bmod 19 = (5 * 9 * 16 * 17 * 5) \bmod 19 = 61200 \bmod 19 = \mathbf{1}$

$(5 * 9) \bmod 19 = 7$
$(7 * 16) \bmod 19 = 17$
$(17 * 17) \bmod 19 = 4$
$(4 * 5) \bmod 19 = 1$

https://www.khanacademy.org/computing/computer-science/cryptography/modarithmetic/a/fast-modular-exponentiation

Practice

Compute the following using fast modular exponentiation:

$5^{35} \bmod 11$

$5^{68} \bmod 7$

$53^{27} \bmod 12$

$46^{39} \bmod 11$

See additional exercises 9.7.2 a-d for sample solutions.