Solution: Bit-by-bit

Author: Misha Wolfson

The key to this puzzle was understanding the flavor: it hinted at bits (binary) in several ways: the name "bit-by-bit," the binary numbers in front of the questions, and the word "bit-coin" in the flavor text.

To find the answer, first fill in the information and solve the math problems in the questions:

```
1. (000001) 14 = 2 * 7             = first nontrivial prime * Snow White's dwarves
2. (000010) 30 = 3 * 10            = NAFTA countries * toes
3. (000011) 34 = 29 + 5            = days in a leap month + guys' burgers
4. (000100) 35 = 2 ^ 2 ^ 2 * 2 + 3 = (live crew ^ legit to quit ^ _pac Shakur) * taken
                                          to tango + stooges
5. (000101) 47 = 42 + 5            = answer to the meaning of life + work week days
6. (000110) 50 = 23 * 2 + 2 + 2    = Michael Jordan * nostrils + kidneys + lungs
7. (000111) 61 = 50 + 11           = states + last possible hour
8. (001000) 63 = 8 * 8 - 1         = super motel * black ball - singular sensation
```

The question numbers corresponded to the numbers on the coins: the answer to a math problem went on the coin whose number corresponded to that problem.

Next, the numbers on the coins needed to be converted back to binary (as clued by matching the 6-bit binary problem numbers to the decimal numbers on the coins). This yielded a grid-like structure of binary numbers.

The a-ha moment of the puzzle involved taking every binary number bit-by-bit and looking at the grid's appearance. The Deck Hands version of the puzzle came with a set of 6 empty grids to help clue this transformation. Taking the binary numbers in each grid bit-by-bit resulted in the following grids, where "x"s represent 1s (to the right):

```
 xxxx
x
x
x
 xxxx

x   x
x   x
xxxxx
x   x
x   x

 xxx
x   x
x   x
x   x
 xxx

 xxx
x   x
x   x
x   x
 xxx

 xxx
x
 xxx
    x
 xxx

xxxxx
x
xxxx
x
xxxxx
```

The grids spelled out the word CHOOSE which was the solution.

(The commented Perl code used to generate the sums from the grids is attached for reference.)

```
#!/usr/bin/env perl

use v5.16;

# Encode the answer (CHOOSE) into an array of strings;
# One string per letter, has only the ‘x’/’ ‘ parts, no whitespace.
my @letters = map { s/\t//gr } split /-\n/, (q{
	 xxxx
	x
	x
	x
	 xxxx
-
	x   x
	x   x
	xxxxx
	x   x
	x   x
-
	 xxx
	x   x
	x   x
	x   x
	 xxx
-
	 xxx
	x   x
	x   x
	x   x
	 xxx
-
	 xxx
	x
	 xxx
	    x
	 xxx
-
	xxxxx
	x
	xxxx
	x
	xxxxx
} =~ s/^\n//r);

my @sum;

for my $letterno (0 .. $#letters) {
	my $letter = $letters[$letterno];

	# Set the appropriate bit value: powers of 2.
	my $bit = 1 << $letterno;

	# For each x in the letter, add the appropriate bit to the sum in the
	# same coordinate where x is in the letter.
	my @lines = split /\n/, $letter;
	for my $lineno (0 .. $#lines) {
		my $line = $lines[$lineno];
		my @chars = split //, $line;
		for my $charno (0 .. $#chars) {
			my $char = $chars[$charno];
			if ($char eq ‘x’) {
				$sum[$lineno][$charno] += $bit;
			}
		}
	}
};

# Output the final 2d grid of sums
for my $row (@sum) {
	say join(“ “, map { sprintf “%2d”, $_ } @$row);
}
```