--- Day 9: Sensor Boost ---

You've just said goodbye to the rebooted rover and left Mars when you receive a faint distress signal coming from the asteroid belt. It must be the Ceres monitoring station!

In order to lock on to the signal, you'll need to boost your sensors. The Elves send up the latest BOOST program - Basic Operation Of System Test.

While BOOST (your puzzle input) is capable of boosting your sensors, for tenuous safety reasons, it refuses to do so until the computer it runs on passes some checks to demonstrate it is a complete Intcode computer.

Your existing Intcode computer is missing one key feature: it needs support for parameters in relative mode.

Parameters in mode `2`, relative mode, behave very similarly to parameters in position mode: the parameter is interpreted as a position. Like position mode, parameters in relative mode can be read from or written to.

The important difference is that relative mode parameters don't count from address `0`. Instead, they count from a value called the relative base. The relative base starts at `0`.

The address a relative mode parameter refers to is itself plus the current relative base. When the relative base is `0`, relative mode parameters and position mode parameters with the same value refer to the same address.

For example, given a relative base of `50`, a relative mode parameter of `-7` refers to memory address `50 + -7 = 43`.

The relative base is modified with the relative base offset instruction:

- Opcode `9` adjusts the relative base by the value of its only parameter. The relative base increases (or decreases, if the value is negative) by the value of the parameter.

For example, if the relative base is `2000`, then after the instruction `109,19`, the relative base would be `2019`. If the next instruction were `204,-34`, then the value at address `1985` would be output.

Your Intcode computer will also need a few other capabilities:

- The computer's available memory should be much larger than the initial program. Memory beyond the initial program starts with the value `0` and can be read or written like any other memory. (It is invalid to try to access memory at a negative address, though.)
- The computer should have support for large numbers. Some instructions near the beginning of the BOOST program will verify this capability.

Here are some example programs that use these features:

- `109,1,204,-1,1001,100,1,100,1008,100,16,101,1006,101,0,99` takes no input and produces a copy of itself as output.
- `1102,34915192,34915192,7,4,7,99,0` should output a 16-digit number.
- `104,1125899906842624,99` should output the large number in the middle.

The BOOST program will ask for a single input; run it in test mode by providing it the value `1`. It will perform a series of checks on each opcode, output any opcodes (and the associated parameter modes) that seem to be functioning incorrectly, and finally output a BOOST keycode.

Once your Intcode computer is fully functional, the BOOST program should report no malfunctioning opcodes when run in test mode; it should only output a single value, the BOOST keycode. What BOOST keycode does it produce?

To begin, get your puzzle input.

Answer: [Submit]

You can also [Share] this puzzle.