The problem

You’ve been tasked by humanity to stop global warming and, as a diligent engineer, you know

You need to measure things you want to change. Hence, you’re starting your earth-saving

Challenge by writing a tool that can estimate the power draw of a single dimmable smart light

Bulb.

When fully lit, the light consumes 5W, and this then drops linearly as the dimmer is turned down.

Internally, the light represents its dimmer value as a floating point number between 0.0 and 1.0,

Inclusive.

The light outputs a message whenever someone adjusts it. Each message contains a

Timestamp from the light bulb’s internal clock (in seconds since the start of 1970). There are two

Types of message. A Turnoff message indicates that the light has been turned off completely.

A Delta message indicates that the brightness has been adjusted; it includes a value for the

Change in the dimmer value (a floating point number between -1.0 and +1.0 inclusive).

Your tool consumes these messages, estimates the dimmer value over time, and uses that to

Estimate the energy consumed by the light.

But there is a catch. The protocol used to transmit the messages is unreliable. Your tool should

Deal with the messages being duplicated, lost and/or delivered out of order.

The solution

Your solution should take the form of a command line tool which reads messages from stdin

Until it reaches an EOF. It should then print the estimated energy consumed in watt-hours and

Exit.

Here is a more complex example:

$ Energy-estimator <<EOF

> 1544206562 Turnoff

> 1544206563 Delta +0.5

> 1544210163 Delta -0.25

> 1544211963 Delta +0.75

> 1544211963 Delta +0.75

> 1544213763 Turnoff

EOF

Estimated energy used: 5.625 Wh

You can assume that every stream starts and ends with a Turnoff message.

You can choose any reasonably mainstream language you like to implement the tool. We recommend using something that you are familiar with.

You should script a build for the tool which outputs the tool in an easily distributed format (single

File binary or similar). The build and tool itself should run on Linux. You can specify distro and

Version (as long as it's available as a Docker image) and we are happy to install dependencies.

If we can't get it running first time then we're happy to iterate with you.

Please note that there is no right answer here. There are a large number of edge cases which

The tool could encounter due to the unreliability of the system. We would like you to pick a

Subset of these to handle and decide how best to do so.