


```
fn main() {
    timely::execute_from_args(std::env::args(), |worker| {

        let input = worker.new_input();
        let probe = worker.dataflow(|scope| {
            input.to_stream(scope)
                .exchange(|&x| x)
                .inspect(|x| println!("{}", x))
                .probe()
        });

        for round in 0..10 {
            input.send(round);
            input.advance_to(round + 1);
            worker.step_while(|| probe.lt(input.time()));
        }

    });
}
```


A hand-drawn rectangular box with a thick, dark gray border. A small, sharp, upward-pointing triangle is attached to the top edge of the box, pointing towards the top right corner of the image.

Construct a dataflow graph using a new input and ending in a probe.

Adds a timestamp to each record

A hand-drawn diagram consisting of a rectangular box with a dark grey border. Inside the box, the text "Adds a timestamp to each record" is written in a black, sans-serif font. From the bottom center of the box, a dark grey arrow points downwards, ending in a small dot.

A hand-drawn dark gray box with a triangular pointer at the top left. The box contains the text "Reports on possibility of timestamps".

Reports on possibility of timestamps

```

fn main() {
    timely::execute_from_args(std::env::args(), |worker| {

        let input = worker.new_input();
        let probe = worker.dataflow(|scope| {
            input.to_stream(scope)
                .exchange(|&x| x)
                .inspect(|x| println!("{}", x))
                .probe()
        });

        for round in 0..10 {
            input.send(round * round);
            input.advance_to(round + 1);
            while probe.less_than(input.time()) {
                worker.step();
            }
        }

    });
}

```

```
fn main() {  
    timely::execute_from_args(std::env::args(), |worker| {  
  
        let input = worker.new_input();  
        let probe = worker.dataflow(|scope| {  
            input.to_stream(scope)  
                .exchange(|&x| x)  
                .inspect(|x| println!("{}", x))  
                .probe()  
        });  
  
    });  
}
```