Daily Coding Problem #105

Problem

This problem was asked by Facebook.

Given a function `f`, and `N` return a debounced `f` of `N` milliseconds.

That is, as long as the debounced `f` continues to be invoked, `f` itself will not be called for `N` milliseconds.

Solution

We can understand the debounced function as doing these two steps:

1. Cancelling a previously scheduled `f` to be cancelled (if it exists)
2. Re-scheduling `f` to be schedulled in `milliseconds`

To do this, we need to be able to remember how to cancel the previous `f`. So we take the result of `setTimeout` and store it under closure in `defer`. This gives us an id which we can use to cancel it.

```
const debounced = function(f, milliseconds) {
  let defer;

  return function() {
    if (defer) {
      clearTimeout(defer);

    }
```

```
    defer = setTimeout(f, milliseconds)
  }
}
```