



## BLOCK NONCE #5: Making Peace With Snapshot Tests

### What?

Although not ideal, sometimes we want to use hard-coded values in tests — these are called **snapshot tests**. It's helpful when certain outputs are expected to rarely change, like the gas cost of a fixed transaction.

### Why?

When hard-coded tests break, the correct values are often already known — the computer just printed them! So why should we fix them manually?

### How?

All rise for the **expect-test** crate 🎉 — from now, hard-coded values fix themselves! Instead of:

```
assert_eq!(computed_value, 7)
```

Just do:

```
expect!["7"].assert_debug_eq(computed_value);
```

**Oh, you've got multiple test cases? No worries!**

```
fn my_gas_test(#[values(2, 3)] gas_price: u8) {  
    let result = execute_tx(gas_price).gas_price;  
    if gas_price == 2 {  
        expect!["18"].assert_debug_eq(result);  
    } else {  
        expect!["27"].assert_debug_eq(result);  
    }  
}
```

**Wait, the expected output is in a file ? No problem!**

```
expect_file!["path_to_file.txt"].assert_debug_eq(computed_value);
```

#### Call For Action! 📢

Remember that time you were copy-pasting expected values into a test, while quietly wondering why you spent 4 years studying for this?

**Do your teammates a favor—go fix this code now!**