**Using vm.roll and vm.warp**

**vm.roll and vm.wrap**

In lesson 19, we skipped testing one of the four steps of enterRaffle: 2. We check if the `RaffleState` is `OPEN`;

To rephrase it, a user should not be able to enter if the RaffleState is CALCULATING.

function testDontAllowPlayersToEnterWhileRaffleIsCalculating() public {

// Arrange

vm.prank(PLAYER);

raffle.enterRaffle{value: entranceFee}();

vm.warp(block.timestamp + interval + 1);

vm.roll(block.number + 1);

raffle.performUpkeep("");

// Act / Assert

vm.expectRevert(Raffle.Raffle\_\_RaffleNotOpen.selector);

vm.prank(PLAYER);

raffle.enterRaffle{value: entranceFee}();

}

We start our test exactly like the others. We prank the PLAYER and we call enterRaffle specifying the appropriate msg.value so our user registers properly.

The following step involves calling two new cheatcodes:

* [vm.warp](https://book.getfoundry.sh/cheatcodes/warp?highlight=warp#warp) which sets the block.timestamp;
* [vm.roll](https://book.getfoundry.sh/cheatcodes/roll?highlight=roll#roll) which sets the block.number;

Even though we don't use them here it's important to know that there are other block.timestamp manipulation cheatcodes that you'll encounter in your development/security path.

* [skip](https://book.getfoundry.sh/reference/forge-std/skip) which skips forward the block.timestamp by the specified number of seconds;
* [rewind](https://book.getfoundry.sh/reference/forge-std/rewind) which is the antonym of skip, i.e. it rewinds the block.timestamp by a specified number of seconds;

So we use the vm.warp and vm.roll to push the block.timestamp and block.number in the future.

We call performUpkeep to change the RaffleState to CALCULATING.

Following that we call the vm.expectRevert cheatcode, expecting to revert the next call with the Raffle\_\_RaffleNotOpen error.

The last step is pranking the PLAYER again and calling enterRaffle to check if it reverts as it should.

Run the test using forge test --mt testDontAllowPlayersToEnterWhileRaffleIsCalculating

Ran 1 test for test/unit/RaffleTest.t.sol:RaffleTest

[FAIL. Reason: InvalidConsumer()] testDontAllowPlayersToEnterWhileRaffleIsCalculating() (gas: 101956)

Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 2.70ms (206.20µs CPU time)

OH NO! [FAIL. Reason: InvalidConsumer()] ... we gonna fix this one soon, I promise!