Incremental testing

It is tempting to write the whole program, and then start debugging it.

Don't do that. It is much more effective to test every function after you have written it. Continue working on the next function only after the first one works correctly.

We can test functions using the interactive mode:

```
> ktc
>>> :load triangle.kts
```


Collatz problem

Consider a sequence of integers following the rule:

$$n_{i+1} = \begin{cases} 3n_i + 1 & \text{if } n_i \text{ is odd} \\ n_i/2 & \text{if } n_i \text{ is even} \end{cases}$$

It is conjectured that this sequence always arrives at 1.

```
5 16 8 4 2 1
34 17 52 26 13 40 20 10 5 16 8 4 2 1
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
672 336 168 84 42 21 64 32 16 8 4 2 1
```

Which starting value gives the longest chain?

Unit testing

For a large program, it is common to write a separate test program for every function or class. These test programs are called unit tests.

Some programmers write the tests before the code!

The unit tests remain useful even when the program is finished. All software needs to be maintained. Whenever a change is made to the software, we can run the unit tests again to get confidence that we didn't break anything.

On many software projects, unit tests are automated and run every night, to ensure that the changes made during the day didn't create any new bugs.

We will provide some unit tests for some projects.