0.7

Element Finance Process Quality Review

Score: 93%

Overview

This is a Element Finance Process Quality Review completed on August 12th 2021. It was performed using the Process Review process (version 0.7.3) and is documented here. The review was performed by Nic of DeFiSafety. Check out our Telegram.

The final score of the review is 93**%**, a **PASS**. The breakdown of the scoring is in Scoring Appendix. For our purposes, a pass is **70%.**

Summary of the Process

Very simply, the review looks for the following declarations from the developer's site. With these declarations, it is reasonable to trust the smart contracts.

- **Here are my smart contracts on the blockchain**
- **Here is the documentation that explains what my smart contracts do**
- **Here are the tests I ran to verify my smart contract**
- **Here are the audit(s) performed on my code by third party experts**
- **Here are the admin controls and strategies**

Disclaimer

This report is for informational purposes only and does not constitute investment advice of any kind, nor does it constitute an offer to provide investment advisory or other services. Nothing in this report shall be considered a solicitation or offer to buy or sell any security, token, future, option or other financial instrument or to offer or provide any investment advice or service to any person in any jurisdiction. Nothing contained in this report constitutes investment advice or offers any opinion with respect to the suitability of any security, and the views expressed in this report should not be taken as advice to buy, sell or hold any security. The information in this report should not be relied upon for the purpose of investing. In preparing the information contained in this report, we have not taken into account the investment needs, objectives and financial circumstances of any particular investor. This information has no regard to the specific investment objectives, financial situation and particular needs of any specific recipient of this information and investments discussed may not be suitable for all investors.

Any views expressed in this report by us were prepared based upon the information available to us at the time such views were written. The views expressed within this report are limited to DeFiSafety and the author and do not reflect those of any additional or third party and are strictly based upon DeFiSafety, its authors, interpretations and evaluation of relevant data. Changed or additional information could cause such

views to change. All information is subject to possible correction. Information may quickly become unreliable for various reasons, including changes in market conditions or economic circumstances.

Chain

This section indicates the blockchain used by this protocol.

> **Chain:** Ethereum

Guidance:

Ethereum
Binance Smart Chain
Polygon
Avalanche
Terra

Code and Team

This section looks at the code deployed on the Mainnet that gets reviewed and its corresponding software repository. The document explaining these questions is here. This review will answer the following questions:

1) Are the executing code addresses readily available? (%)
2) Is the code actively being used? (%)
3) Is there a public software repository? (Y/N)
4) Is there a development history visible? (%)
5) Is the team public (not anonymous)? (Y/N)

1) Are the executing code addresses readily available? (%)

> **Answer:** 100%

They are available at website https://github.com/element-fi/elf-contracts, as indicated in the Appendix.

Guidance:

100% Clearly labelled and on website, docs or repo, quick to find
70% Clearly labelled and on website, docs or repo but takes a bit of looking
40% Addresses in mainnet.json, in discord or sub graph, etc

20%	Address found but labeling not clear or easy to find
0%	Executing addresses could not be found

How to improve this score:

Make the Ethereum addresses of the smart contract utilized by your application available on either your website or your GitHub (in the README for instance). Ensure the addresses is up to date. This is a very important question towards the final score.

2) Is the code actively being used? (%)

> **Answer:** 100%

Activity is over 10 transactions a day on contract *Tranche.sol*, as indicated in the Appendix.

Guidance:

100%	More than 10 transactions a day
70%	More than 10 transactions a week
40%	More than 10 transactions a month
10%	Less than 10 transactions a month
0%	No activity

3) Is there a public software repository? (Y/N)

> **Answer:** Yes

GitHub: https://github.com/element-fi/elf-contracts.

Is there a public software repository with the code at a minimum, but also normally test and scripts. Even if the repository was created just to hold the files and has just 1 transaction, it gets a **"Yes"**. For teams with private repositories, this answer is **"No"**.

4) Is there a development history visible? (%)

> **Answer:** 100%

With 180 commits and 7 branches, this is a robust software repository.

This metric checks if the software repository demonstrates a strong steady history. This is normally demonstrated by commits, branches and releases in a software repository. A healthy history demonstrates a history of more than a month (at a minimum).

Guidance:

100% Any one of 100+ commits, 10+branches
70% Any one of 70+ commits, 7+branches
50% Any one of 50+ commits, 5+branches
30% Any one of 30+ commits, 3+branches
0% Less than 2 branches or less than 30 commits

5) Is the team public (not anonymous)? (Y/N)

> **Answer:** Yes

Location: https://www.element.fi/, at the bottom of the page.

For a **"Yes"** in this question, the real names of some team members must be public on the website or other documentation (LinkedIn, etc). If the team is anonymous, then this question is a **"No"**.

Documentation

This section looks at the software documentation. The document explaining these questions is here.

Required questions are;

6) Is there a whitepaper? (Y/N)
7) Are the basic software functions documented? (Y/N)
8) Does the software function documentation fully (100%) cover the deployed contracts? (%)
9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)
10) Is it possible to trace from software documentation to the implementation in code (%)

6) Is there a whitepaper? (Y/N)

> **Answer:** Yes

Location: https://docs.element.fi/.

7) Are the basic software functions documented? (Y/N)

> **Answer:** Yes

The Element Finance basic software functions are documented in the "Developers" section of their

documentation.

8) Does the software function documentation fully (100%) cover the deployed contracts? (%)

> **Answer:** 80%

The major Element Finance software functions are documented in the "Developers" section of their documentation.

Guidance:

100% All contracts and functions documented
80% Only the major functions documented
79-1% Estimate of the level of software documentation
0% No software documentation

How to improve this score:

This score can be improved by adding content to the software functions document such that it comprehensively covers the requirements. For guidance, refer to the SecurEth System Description Document. Using tools that aid traceability detection will help.

9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)

> **Answer:** 60%

Code examples are in the Appendix. As per the SLOC, there is 60% commenting to code (CtC).

The Comments to Code (CtC) ratio is the primary metric for this score.

Guidance:

100% CtC > 100 Useful comments consistently on all code
90-70% CtC > 70 Useful comment on most code
60-20% CtC > 20 Some useful commenting
0% CtC < 20 No useful commenting

How to improve this score

This score can improve by adding comments to the deployed code such that it comprehensively covers the code. For guidance, refer to the SecurEth Software Requirements.

10) Is it possible to trace from software documentation to the implementation in code (%)

>

> **Answer:** 60%

The Element Finance developer documentation lists and describes all the functions with non-explicit tracability.

Guidance:

100% Clear explicit traceability between code and documentation at a requirement level for all code
60% Clear association between code and documents via non explicit traceability
40% Documentation lists all the functions and describes their functions
0% No connection between documentation and code

How to improve this score:

This score can improve by adding traceability from documentation to code such that it is clear where each outlined function is coded in the source code. For reference, check the SecurEth guidelines on traceability.

Testing

This section looks at the software testing available. It is explained in this document. This section answers the following questions;

11) Full test suite (Covers all the deployed code) (%)
12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)
13) Scripts and instructions to run the tests (Y/N)
14) Report of the results (%)
15) Formal Verification test done (%)
16) Stress Testing environment (%)

11) Is there a Full test suite? (%)

> **Answer:** 100%

Code examples are in the Appendix. As per the SLOC, there is 308% testing to code (TtC).

This score is guided by the Test to Code ratio (TtC). Generally a good test to code ratio is over 100%. However the reviewers best judgement is the final deciding factor.

Guidance:

100% TtC > 120% Both unit and system test visible
80% TtC > 80% Both unit and system test visible
40% TtC < 80% Some tests visible
0% No tests obvious

12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)

> **Answer:** 92%

Element Finance's core contracts' software repository has 92% code coverage according to their coveralls report.

Guidance:

100%	Documented full coverage
99-51%	Value of test coverage from documented results
50%	No indication of code coverage but clearly there is a reasonably complete set of tests
30%	Some tests evident but not complete
0%	No test for coverage seen

How to improve this score:

This score can improved by adding tests that achieve full code coverage. A clear report and scripts in the software repository will guarantee a high score.

13) Scripts and instructions to run the tests (Y/N)

> **Answer:** Yes

Scrips/Instructions location: https://github.com/element-fi/elf-contracts/tree/main/scripts.

14) Report of the results (%)

> **Answer:** 100%

A full code coverage report is availible here at: https://github.com/element-fi/elf-contracts/blob/main/audits/test-report.md

Guidance:

100%	Detailed test report as described below
70%	GitHub code coverage report visible
0%	No test report evident

How to improve this score

Add a report with the results. The test scripts should generate the report or elements of it.

15) Formal Verification test done (%)

> ⚠ **Answer:** 0%

No evidence of a Element Finance Formal Verification test was found in their documentation or in further web research.

16) Stress Testing environment (%)

> ✓ **Answer:** 100%

There is evidence of Element Finance's Goerli testnet smart contract usage at https://github.com/element-fi/elf-deploy/blob/main/addresses/goerli.json.

Security

This section looks at the 3rd party software audits done. It is explained in this document. This section answers the following questions;

17) Did 3rd Party audits take place? (%)
18) Is the bounty value acceptably high?

17) Did 3rd Party audits take place? (%)

> ✓ **Answer:** 100%

PeckShield has published a Element Finance audit report on April 14th 2021, which was before their mainnet launch on June 30th.

Runtime Verification has published a Element Finance audit report on April 26th, which was before their mainnet launch on June 30th.

Note: The Element Finance team has successfully implemented most of the fix recommendations from both audit reports.

Guidance:

100% Multiple Audits performed before deployment and results public and implemented or not required

90% Single audit performed before deployment and results public and implemented or not required
70% Audit(s) performed after deployment and no changes required. Audit report is public

50% Audit(s) performed after deployment and changes needed but not implemented
20% No audit performed
0% Audit Performed after deployment, existence is public, report is not public and no improvements deployed OR smart contract address' not found, (where question 1 is 0%)

Deduct 25% if code is in a private repo and no note from auditors that audit is applicable to deployed code

18) Is the bounty value acceptably high (%)

> ⓘ **Answer: 7**0%

Element Finance's Immunefi Bug Bounty program rewards participating users with up to 100k for the most critical of finds, and it is an active program.

Guidance:

100% Bounty is 10% TVL or at least $1M AND active program (see below)
90% Bounty is 5% TVL or at least 500k AND active program
80% Bounty is 5% TVL or at least 500k
70% Bounty is 100k or over AND active program
60% Bounty is 100k or over
50% Bounty is 50k or over AND active program
40% Bounty is 50k or over
20% Bug bounty program bounty is less than 50k
0% No bug bounty program offered

An active program means that a third party (such as Immunefi) is actively driving hackers to the site. An inactive program would be static mentions on the docs.

Access Controls

This section covers the documentation of special access controls for a DeFi protocol. The admin access controls are the contracts that allow updating contracts or coefficients in the protocol. Since these contracts can allow the protocol admins to "change the rules", complete disclosure of capabilities is vital for user's transparency. It is explained in this document. The questions this section asks are as follow;

19) Can a user clearly and quickly find the status of the admin controls?
20) Is the information clear and complete?
21) Is the information in non-technical terms that pertain to the investments?
22) Is there Pause Control documentation including records of tests?

19) Can a user clearly and quickly find the status of the access controls (%)

> **Answer:** 100%

Admin control information is easily found on their documentation.
https://docs.element.fi/developers/security/admin-keys

Guidance:

100%	Clearly labelled and on website, docs or repo, quick to find
70%	Clearly labelled and on website, docs or repo but takes a bit of looking
40%	Access control docs in multiple places and not well labelled
20%	Access control docs in multiple places and not labelled
0%	Admin Control information could not be found

20) Is the information clear and complete (%)

> **Answer:** 100%

All contracts are immutable according to their admin keys documentation.

Guidance:

All the contracts are immutable -- 100% OR

a) All contracts are clearly labelled as upgradeable (or not) -- 30% AND
b) The type of ownership is clearly indicated (OnlyOwner / MultiSig / Defined Roles) -- 30% AND
c) The capabilities for change in the contracts are described -- 30%

How to improve this score:

Create a document that covers the items described above. An example is enclosed.

21) Is the information in non-technical terms that pertain to the investments (%)

> **Answer:** 100%

All contracts are immutable according to their admin keys documentation.

Guidance:

100%	All the contracts are immutable
90%	Description relates to investments safety and updates in clear, complete non-software l language

30% Description all in software specific language
0% No admin control information could not be found

How to improve this score:

Create a document that covers the items described above in plain language that investors can understand. An example is enclosed.

22) Is there Pause Control documentation including records of tests (%)

> **Answer:** 80%

Pause control is mentioned in their admin keys documentation.

Guidance:

100% All the contracts are immutable or no pause control needed and this is explained OR
100% Pause control(s) are clearly documented and there is records of at least one test within 3 months

80% Pause control(s) explained clearly but no evidence of regular tests
40% Pause controls mentioned with no detail on capability or tests
0% Pause control not documented or explained

How to improve this score**:**

Create a document that covers the items described above in plain language that investors can understand. An example is enclosed.

Appendices

Author Details

The author of this review is Rex of DeFi Safety.

Email : rex@defisafety.com Twitter : @defisafety

I started with Ethereum just before the DAO and that was a wonderful education. It showed the importance of code quality. The second Parity hack also showed the importance of good process. Here my aviation background offers some value. Aerospace knows how to make reliable code using quality processes.

I was coaxed to go to EthDenver 2018 and there I started SecuEth.org with Bryant and Roman. We created guidelines on good processes for blockchain code development. We got EthFoundation funding to assist in their development.

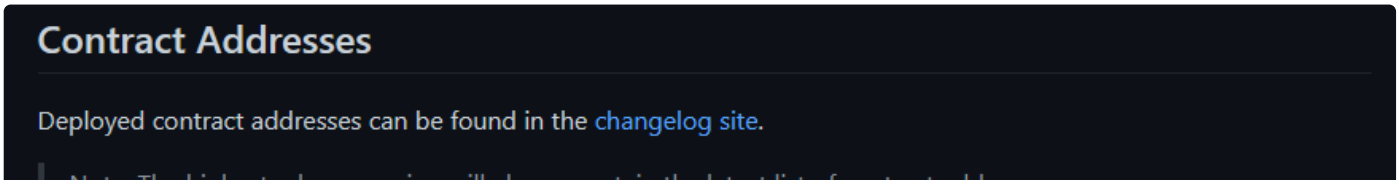
Process Quality Reviews are an extension of the SecurEth guidelines that will further increase the quality processes in Solidity and Vyper development.

DeFiSafety is my full time gig and we are working on funding vehicles for a permanent staff.

Scoring Appendix

	Total	Element Finance	
PQ Audit Scoring Matrix (v0.7)	**Points**	**Answer**	**Points**
Total	**260**		240.6
Code and Team			**93%**
1) Are the executing code addresses readily available? (%)	**20**	100%	20
2) Is the code actively being used? (%)	**5**	100%	5
3) Is there a public software repository? (Y/N)	**5**	Y	5
4) Is there a development history visible? (%)	**5**	100%	5
5) Is the team public (not anonymous)? (Y/N)	**15**	Y	15
Code Documentation			
6) Is there a whitepaper? (Y/N)	**5**	Y	5
7) Are the basic software functions documented? (Y/N)	**10**	Y	10
8) Does the software function documentation fully (100%) c	**15**	80%	12
9) Are there sufficiently detailed comments for all function	**5**	60%	3
10) Is it possible to trace from software documentation to th	**10**	60%	6
Testing			
11) Full test suite (Covers all the deployed code) (%)	**20**	100%	20
12) Code coverage (Covers all the deployed lines of code, or	**5**	92%	4.6
13) Scripts and instructions to run the tests? (Y/N)	**5**	Y	5
14) Report of the results (%)	**10**	100%	10
15) Formal Verification test done (%)	**5**	0%	0
16) Stress Testing environment (%)	**5**	100%	5
Security			
17) Did 3rd Party audits take place? (%)	**70**	100%	70
18) Is the bug bounty acceptable high? (%)	**10**	70%	7
Access Controls			
19) Can a user clearly and quickly find the status of the admi	**5**	100%	5
20) Is the information clear and complete	**10**	100%	10
21) Is the information in non-technical terms	**10**	100%	10
22) Is there Pause Control documentation including records	**10**	80%	8
Section Scoring			
Code and Team	50	100%	
Documentation	45	80%	
Testing	50	89%	
Security	80	96%	
Access Controls	35	94%	

Executing Code Appendix



> Note: The highest release version will always contain the latest list of contract addresses.
>
> Additionally, the latest deployed contract addresses for Goerli and Mainnet can be found here and here respectively.

Code Used Appendix

Parent Txn Hash	Block	Age	From	To	Value
0xd77e2d692435612ef5...	13006587	20 hrs 11 mins ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether
0xae49b920af7b3dcc1b...	13005359	1 day 45 mins ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether
0x82150ff480d2b27f52c...	13005301	1 day 56 mins ago	0x26941c63f4587796ab...	0x94046274b5aa816ab...	0 Ether
0x82150ff480d2b27f52c...	13005301	1 day 56 mins ago	0x26941c63f4587796ab...	0xb3295e739380bd68d...	0 Ether
0x82150ff480d2b27f52c...	13005301	1 day 56 mins ago	0xee4e158c03a10cbc82...	0x26941c63f4587796ab...	0 Ether
0x5d5e7329fd0ab9a64c...	13005290	1 day 58 mins ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether
0xcab35550b81ab0e990...	13005278	1 day 1 hr ago	0x26941c63f4587796ab...	0x94046274b5aa816ab...	0 Ether
0xcab35550b81ab0e990...	13005278	1 day 1 hr ago	0x26941c63f4587796ab...	0xb3295e739380bd68d...	0 Ether
0xcab35550b81ab0e990...	13005278	1 day 1 hr ago	0xee4e158c03a10cbc82...	0x26941c63f4587796ab...	0 Ether
0x1b4475c6b6c4dba904...	13005266	1 day 1 hr ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether
0x0e11c2127919e4c0e3...	13004924	1 day 2 hrs ago	0x26941c63f4587796ab...	0x94046274b5aa816ab...	0 Ether
0x0e11c2127919e4c0e3...	13004924	1 day 2 hrs ago	0x26941c63f4587796ab...	0xb3295e739380bd68d...	0 Ether
0x0e11c2127919e4c0e3...	13004924	1 day 2 hrs ago	0xee4e158c03a10cbc82...	0x26941c63f4587796ab...	0 Ether
0x0e11c2127919e4c0e3...	13004924	1 day 2 hrs ago	0xee4e158c03a10cbc82...	0x26941c63f4587796ab...	0 Ether
0x17f4bacdd272365241...	13004745	1 day 3 hrs ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether
0xd7bf16170ef65cae9cb...	13004651	1 day 3 hrs ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether
0x877512a79cb516284b...	13004521	1 day 3 hrs ago	0xba12222222228d8ba...	0x26941c63f4587796ab...	0 Ether

Example Code Appendix

```
/// @author Element Finance
/// @title Tranche
contract Tranche is ERC20Permit, ITranche {
    IInterestToken public immutable override interestToken;
    IWrappedPosition public immutable position;
    IERC20 public immutable underlying;
    uint8 internal immutable _underlyingDecimals;

    // The outstanding amount of underlying which
    // can be redeemed from the contract from Principal Tokens
    // NOTE - we use smaller sizes so that they can be one storage slot
    uint128 public valueSupplied;
    // The total supply of interest tokens
    uint128 public override interestSupply;
    // The timestamp when tokens can be redeemed.
    uint256 public immutable unlockTimestamp;
    // The amount of slippage allowed on the Principal token redemption [0.1 basis points]
    uint256 internal constant _SLIPPAGE_BP = 1e13;
    // The speedbump variable records the first timestamp where redemption was attempted t
    // performed on a tranche where loss occurred. It blocks redemptions for 48 hours afte

```

```
    // it is triggered in order to (1) prevent atomic flash loan price manipulation (2)
    // give 48 hours to remediate any other loss scenario before allowing withdraws
    uint256 public speedbump;
    // Const which is 48 hours in seconds
    uint256 internal constant _FORTY_EIGHT_HOURS = 172800;
    // An event to listen for when negative interest withdraw are triggered
    event SpeedBumpHit(uint256 timestamp);

    /// @notice Constructs this contract
    constructor() ERC20Permit("Element Principal Token ", "eP") {
        // Assume the caller is the Tranche factory.
        ITrancheFactory trancheFactory = ITrancheFactory(msg.sender);
        (
            address wpAddress,
            uint256 expiration,
            IInterestToken interestTokenTemp,
            // solhint-disable-next-line
            address unused
        ) = trancheFactory.getData();
        interestToken = interestTokenTemp;

        IWrappedPosition wpContract = IWrappedPosition(wpAddress);
        position = wpContract;

        // Store the immutable time variables
        unlockTimestamp = expiration;
        // We use local because immutables are not readable in construction
        IERC20 localUnderlying = wpContract.token();
        underlying = localUnderlying;
        // We load and store the underlying decimals
        uint8 localUnderlyingDecimals = localUnderlying.decimals();
        _underlyingDecimals = localUnderlyingDecimals;
        // And set this contract to have the same
        _setupDecimals(localUnderlyingDecimals);
    }

    /// @notice We override the optional extra construction function from ERC20 to change r
    function _extraConstruction() internal override {
        // Assume the caller is the Tranche factory and that this is called from constructo
        // We have to do this double load because of the lack of flexibility in constructo
        ITrancheFactory trancheFactory = ITrancheFactory(msg.sender);
        (
            address wpAddress,
            uint256 expiration,
            // solhint-disable-next-line
            IInterestToken unused,
            address dateLib
        ) = trancheFactory.getData();

        string memory strategySymbol = IWrappedPosition(wpAddress).symbol();

        // Write the strategySymbol and expiration time to name and symbol

```

```
        // This logic was previously encoded as calling a library "DateString"
        // in line and directly. However even though this code is only in the constructor
        // it both made the code of this contract much bigger and made the factory
        // un deployable. So we needed to use the library as an external contract
        // but solidity does not have support for address to library conversions
        // or other support for working directly with libraries in a type safe way.
        // For that reason we have to use this ugly and non type safe hack to make these
        // contracts deployable. Since the library is an immutable in the factory
        // the security profile is quite similar to a standard external linked library.

        // We load the real storage slots of the symbol and name storage variables
        uint256 namePtr;
        uint256 symbolPtr;
        assembly {
            namePtr := name.slot
            symbolPtr := symbol.slot
        }
        // We then call the 'encodeAndWriteTimestamp' function on our library contract
        (bool success1, ) = dateLib.delegatecall(
            abi.encodeWithSelector(
                DateString.encodeAndWriteTimestamp.selector,
                strategySymbol,
                expiration,
                namePtr
            )
        );
        (bool success2, ) = dateLib.delegatecall(
            abi.encodeWithSelector(
                DateString.encodeAndWriteTimestamp.selector,
                strategySymbol,
                expiration,
                symbolPtr
            )
        );
        // Assert that both calls succeeded
        assert(success1 && success2);
    }

    /// @notice An aliasing of the getter for valueSupplied to improve ERC20 compatibility
    /// @return The number of principal tokens which exist.
    function totalSupply() external view returns (uint256) {
        return uint256(valueSupplied);
    }

    /**
    @notice Deposit wrapped position tokens and receive interest and Principal ERC20 tokens
            If interest has already been accrued by the wrapped position
            tokens held in this contract, the number of Principal tokens minted is
            reduced in order to pay for the accrued interest.
    @param _amount The amount of underlying to deposit
    @param _destination The address to mint to
    @return The amount of principal and yield token minted as (pt, yt)
     */
```

```
127     function deposit(uint256 _amount, address _destination)
128         external
129         override
130         returns (uint256, uint256)
131     {
132         // Transfer the underlying to be wrapped into the position
133         underlying.transferFrom(msg.sender, address(position), _amount);
134         // Now that we have funded the deposit we can call
135         // the prefunded deposit
136         return prefundedDeposit(_destination);
137     }
138
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	12	2524	189	871	1464	121

Comments to Code 871/1464 = 60%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	34	5658	569	574	4515	76

Tests to Code 4515/1464 = 308%