TE002 Test Report on the Module codecs_lib.vigenere

Conventions

Each test is defined following the same format. Each test receives a unique test identifier and a reference to the ID(s) of the requirements it covers (if applicable). The goal of the test is described to clarify what is to be tested. The test steps are described in brief but clear instructions. For each test it is defined what the expected results are for the test to pass. Finally, the test result is given, this can be only pass or fail.

The test format is as follows:

Test Identifier: TEST-[I/A/D/T]-XYZ

Requirement ID(s): REQ-uvw-xyz

Verification method: I/A/D/T

Test goal: Description of what is to be tested

Expected result: What test result is expected for the test to pass

Test steps: Step by step instructions on how to perform the test

Test result: PASS/FAIL

The test ID starts with the fixed prefix 'TEST'. The prefix is followed by a single letter, which defines the test type / verification method. The last part of the ID is a 3-digits *hexadecimal* number (0..9|A..F), with the first digit identifing the module, the second digit identifing a class / function, and the last digit - the test ordering number for this object. E.g. 'TEST-T-112'. Each test type has its own counter, thus 'TEST-T-112' and 'TEST-A-112' tests are different entities, but they refer to the same object (class or function) within the same module.

The verification method for a requirement is given by a single letter according to the table below:

Term	Definition
Inspection (I)	Control or visual verification
Analysis (A)	Verification based upon analytical evidences
Test (T)	Verification of quantitative characteristics with quantitative measurement
Demonstration (D)	Verification of operational characteristics without quantitative measurement

Test definitions (Test)

Test Identifier: TEST-T-200

Requirement ID(s): REQ-FUN-202, REQ-FUN-210 and REQ-FUN-220

Verification method: T

Test goal: Test the continous rotation of the passphrase with each encoded or decoded character.

Expected result: Encoding an decoding results are as expected, no exception is raised.

Test steps:

- Generate an arbitrary passphrase as a bytestring and convert it into a bytes array
- Instantiate a tested **CircularList** class and set its content by that bytes array
- Instantiate the Vignere codec and set its passphrase by that bytes array
- Generate an arbitrary ASCII only bytestring as a test value
- Encode the entire test bytestring and call *CircularList.getElement*() the same number of times
- For each byte *InByte* in the test bytestring do:
    - Encode it with the codec instance chr(*InByte*) -> *OutByte*
    - Get one (current) element stored in the **CircularList** instance as *Code*
    - Check that ord(*OutByte*) = (*InByte* + *Code*) mod 256
    - Append ord(*OutByte*) to a temporary bytes array *Buffer*
- Reset the internal indexes in the **CircularList** instance and the Vignere codec
- Encode the entire test bytestring and call *CircularList.getElement*() the same number of times
- For each byte *OutByte* in the *Buffer* do:
    - Decode bytearray([*OutByte*]) with the codec instance into a string (1 character) *NewChar*
    - Get one (current) element stored in the **CircularList** instance as *Code1*
    - Check that ord(*NewChar*) = (*OutByte* - *Code1*) mod 256
- Reset the internal indexes in the both codec and **CircularList** instances and repeat the check above
- Generate a new passphrase, set it into the both codec and **CircularList** instances and repeat the check above

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-201

Requirement ID(s): REQ-FUN-210, REQ-FUN-220 and REQ-FUN-230

Verification method: T

Test goal: Test that the implementation of the codec is according the specifications, and the decoding is *true* inverse operation for *encoding*.

Expected result: Encoding an decoding results are as expected, no exception is raised.

Test steps::

- Instantiate the Vignere codec
- Generate an arbitrary passphrase as an Unicode string and set it for the codec
- Generate an arbitrary test Unicode string *InData*
- Encode it without indication for the Unicode codec with Vignere codec into *OutData* and reset the index / counter of the Vigenere codec

- Check that *OutData* is a bytestring
- Decode *OutData* without indication for the Unicode codec with Vignere codec into *NewData* and reset the index / counter of the Vigenere codec
- Check that *NewData* is a string and *NewData* == *InData*
- Convert *OutData* into a bytes array and decode without indication for the Unicode codec with Vignere codec into *NewData1* and reset the index / counter of the Vigenere codec
- Check that *NewData1* is a string and *NewData1* == *InData*
- Repeat the encoding + decoding test cycle with specific indication of a proper and compatible Unicode codec
- Generate an arbitrary passphrase as a bytestring, set it and repeat the tests
- Generate an arbitrary passphrase as a bytes array, set it and repeat the tests

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-202

Requirement ID(s): REQ-AWM-200

Verification method: T

Test goal: Tests that use of non-existent or incompatible Unicode codec results in **ValueError**-type exception.

Expected result: A sub-class of **ValueError** exception is raised with each inappropriate call.

Test steps::

- Instantiate the Vignere codec and set b'\x00' as its passphrase (no data scrambling, actually)
- Generate a test string containing ASCII printable characters and non-ASCII characters, including 'U2600' ('sun' code)
- Try to encode this string with the Vigenere codec explicitely indicating Unicode codecs - a sub-class of **ValueError** exception should be raised each time:
    - 'ascii' (incompatible)
    - 'cp1251' (incompatible)
    - 'whatever' (non-existing)
- Encode the genereated test string without any Unicode codec being indicated (using the default UTF-8)
- Try to decode that new bytestring with the Vigenere codec explicitely indicating Unicode codecs - a sub-class of **ValueError** exception should be raised each time:
    - 'ascii' (incompatible)
    - 'cp1251' (incompatible)
    - 'utf_32' (incompatible)
    - 'whatever' (non-existing)

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-203

Requirement ID(s): REQ-AWM-201

Verification method: T

Test goal: Tests that use of any type but a string for Unicode codec argument results in **TypeError**-type exception.

Expected result: A sub-class of **TypeError** exception is raised with each inappropriate call.

Test steps::

- Instantiate the Vignere codec and set b'\x00' as its passphrase (no data scrambling, actually)
- Generate an arbitrary test string
- Try to encode this string with the Vigenere codec explicitely indicating a number of different data types except a string value for the Unicode codec - a sub-class of **TypeError** exception should be raised each time:
- Convert the test string into a valid bytestring using UTF-8 codec
- Try to decode that new bytestring with the Vigenere codec explicitely indicating a number of different data types except a string value for the Unicode codec - a sub-class of **TypeError** exception should be raised each time

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-204

Requirement ID(s): REQ-AWM-202

Verification method: T

Test goal: Test that an exception is raised if encoding or decoding is attempted before setting the passphrase.

Expected result: Exception (sub-class of) exception is raised with each inappropriate call.

Test steps::

- Instantiate the Vignere codec w/o setting a passphrase
- Generate an arbitrary test string
- Try to encode this string with the Vigenere codec - check for an exeption being raised
- Generate an arbitrary bytestring
- Try to decode that new bytestring with the Vigenere codec- check for an exeption being raised

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-210

Requirement ID(s): REQ-AWM-210

Verification method: T

Test goal: Test that an improper type of the argument passed into the encoding method results in an exception of the **TypeError** type.

Expected result: TypeError (sub-class of) exception is raised with each inappropriate call.

Test steps::

- Instantiate the Vignere codec and set an arbitrary bytestring as the passphrase
- Try to encode an arbitrary value of the various improper types - anything except a string

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-220

Requirement ID(s): REQ-AWM-220

Verification method: T

Test goal: Test that an improper type of the argument passed into the decoding method results in an exception of the **TypeError** type.

Expected result: TypeError (sub-class of) exception is raised with each inappropriate call.

Test steps::

- Instantiate the Vignere codec and set an arbitrary bytestring as the passphrase
- Try to decode an arbitrary value of the various improper types - anything except a bytestring or bytes array

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test Identifier: TEST-T-230

Requirement ID(s): REQ-AWM-230

Verification method: T

Test goal: Test that an improper type of the argument passed into the 'set passphrase' method results in an exception of the **TypeError** type.

Expected result: TypeError (sub-class of) exception is raised with each inappropriate call.

Test steps::

- Instantiate the Vignere codec w/o setting a passphrase

- Try to set the passphrase using various inappropriate values of the different types - except a string, a bytestring or bytes array

N.B. implemented as a test case in the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS / FAIL

Test definitions (Analysis)

Test Identifier: TEST-A-200

Requirement ID(s): REQ-FUN-200, REQ-FUN-201

Verification method: A

Test goal: Test that the module implement all of the required functionality and it performs according to the requirements and specifications.

Expected result: All of the unit tests defined by the test cases TEST-T-200, TEST-T-201, TEST-T-202, TEST-T-203, TEST-T-210, TEST-T-220 and TEST-T-230 must pass.

Test steps: Run the test suit module codecs_lib.tests.ut002_vigenere.py

Test result: PASS

Traceability

For traceability the relation between tests and requirements is summarized in the table below:

Requirement ID	Covered in test(s)	Verified [YES/NO])
REQ-FUN-200	TEST-A-200	YES
REQ-FUN-201	TEST-A-200	YES
REQ-FUN-202	TEST-T-200	YES
REQ-FUN-210	TEST-T-200, TEST-T-201	YES
REQ-FUN-220	TEST-T-200, TEST-T-201	YES
REQ-FUN-230	TEST-T-201	YES
REQ-AWM-200	TEST-T-202	YES
REQ-AWM-201	TEST-T-203	YES
REQ-AWM-202	TEST-T-204	YES
REQ-AWM-210	TEST-T-210	YES
REQ-AWM-220	TEST-T-220	YES
REQ-AWM-230	TEST-T-230	YES
Software ready for production [YES/NO]	**Rationale**	

Software ready for production [YES/NO]	Rationale
YES	All tests are passed