

```

1  """
2  This module contains all functions which are used to test
   the program code.
3  """
4
5  import main_module
6  import text_module
7
8
9  # *** tests for the text_module ***
10 # * Caesar *
11 def test_caesar():
12     """
13     tests if encrypt_caesar and decrypt_caesar work
14     :return: True if the former encrypted text decrypted
   is equal to the original text
15     False if not (-> Something went wrong)
16     """
17     print("*** TEST_CAESAR STARTED ***")
18     key = 4
19     original_text = "Hello world"
20     # format text
21     formatted_text = main_module.format_text(original_text
22 )
23     # encrypt text
24     encrypted_text = text_module.encrypt_caesar(
   formatted_text, key)
25     # check if decrypted encrypted text is equal to
   formatted text
26     if text_module.decrypt_caesar(encrypted_text, key) ==
   formatted_text:
27         return True
28     else:
29         return False
30
31 # * Vigenère *
32 def test_create_vigenere_table():
33     """
34     tests to see if create_table_of_vigenere is working, i
   .e. test some letters in the table at specific fields
35     compare result with wikipedia
36     :return: table of vigenere
37     """
38     print("*** TEST_CREATE_VIGENERE_TABLE STARTED ***")

```

```

39     text_module.create_vigenere_table()
40
41
42 def test_vigenere():
43     """
44     tests if encrypt_vigenere and decrypt_vigenere work
45     :return: True if the former encrypted text decrypted
46     is equal to the original text
47     False if not (-> Something went wrong)
48     """
49     print("*** TEST_VIGENERE STARTED ***")
50     key = "magique"
51     original_text = "Hello world"
52     # format text
53     formatted_text = main_module.format_text(original_text
54 )
55     # encrypt text
56     encrypted_text = text_module.encrypt_vigenere(
57 formatted_text, key)
58     # check if decrypted encrypted text is equal to
59 formatted text
60     if text_module.decrypt_vigenere(encrypted_text, key)
61 == formatted_text:
62         return True
63     else:
64         return False
65
66
67 # * Enigma *
68 def test_enigma():
69     """
70     tests if enigma works
71     :return: True if the former encrypted text decrypted
72     is equal to the original text
73     False if not (-> Something went wrong)
74     """
75     print("*** TEST_ENIGMA STARTED ***")
76     key1 = "AAA"
77     key2 = "ADB"
78     original_text = "
79 AABBCCDDEEFFGGHHIIJJKKLLMMNNOOPPQQRRSSTTUUVVWWXXYYZZ"
80
81     other_text = "Ayant le choix ou d'être serf ou d'être
82 libre, quitte la franchise et prend le joug, qui consent à
83 son mal."
84
85     # format text

```

```

75     formatted_text1 = main_module.format_text(
        original_text)
76     formatted_text2 = main_module.format_text(other_text)
77     # check if decrypted encrypted text is equal to
        formatted text
78     # encrypt
79     encrypt1 = text_module.enigma(formatted_text1, key1)
80     encrypt2 = text_module.enigma(formatted_text2, key2)
81     # decrypt
82     decrypt1 = text_module.enigma(encrypt1, key1)
83     decrypt2 = text_module.enigma(encrypt2, key2)
84     if encrypt1 == decrypt1 and encrypt2 == decrypt2:
85         return True
86     else:
87         return False
88
89
90 # *** tests for the main_module ***
91 def test_format_and_normalise():
92     """
93     tests if normalise_letter and format_text work
94     :return: True if there are only capital letters left
        in the formatted text
95     False if not (-> Something went wrong)
96     """
97     print("*** TEST_FORMAT_AND_NORMALISE STARTED ***")
98     original_text = """àâ test1t!? æ test2test,. ç
        test3test;+ èéêë test4test-% îï t5t$& ô
99     test6test\"/ ùûü t7t{} ÿ t8t[] œ t9t= """
100    # format text
101    formatted_text = main_module.format_text(
        original_text)
102    print(formatted_text)
103    # check if only capital letters
104    if formatted_text.isalpha() and formatted_text.
        isupper():
105        return True
106    else:
107        return False
108
109
110 def test_run():
111     print("*** TEST_RUN STARTED ***")
112     print("")
113

```

```
114
115 # run all tests
116 def run_all_tests():
117     """
118     general function to run all tests
119     :return: None
120     """
121     test_caesar()
122     test_vigenere()
123     test_format_and_normalise()
124     test_create_vigenere_table()
125     test_enigma()
126
127
128 # run all tests
129 run_all_tests()
130
```