TESTS ENIGMA

NATHAN HALLEZ – ALEXANDRE HERSSENS

FONCTIONS DE TEST

```
    // ----------------------------------------------------
    //  Fonction test pour lettreEnNombre et nombreEnLettre
    // ----------------------------------------------------
    void testConversionsLettresNombres() {
        for (int i = 0; i < 26; i++){
            assertEquals(i, lettreEnNombre(charAt(ALPHABET, i)));
            assertEquals(charAt(ALPHABET, i), nombreEnLettre(i));
        }
    }
```

```
    // ----------------------------------------------------
    //  Fonction test pour choixRotor
    // ----------------------------------------------------
    void testChoixRotor() {
        assertEquals(ROTOR1, choixRotor(1));
        assertEquals(ROTOR2, choixRotor(2));
        assertEquals(ROTOR3, choixRotor(3));
        assertEquals(ROTOR4, choixRotor(4));
        assertEquals(ROTOR5, choixRotor(5));
    }
```

```
    // ----------------------------------------------------
    //  Fonction test pour decalageUnRang et positionInitialeRotor
    // ----------------------------------------------------
    void testDecalageRotors() {
        assertEquals("BCDEFGHIJKLMNOPQRSTUVWXYZA",
decalageUnRang("ABCDEFGHIJKLMNOPQRSTUVWXYZ"));
        assertEquals("FHUQSMDVHNQOIVHZI", decalageUnRang("IFHUQSMDVHNQOIVHZ"));

        assertEquals("DEFGHIJKLMNOPQRSTUVWXYZABC",
positionInitialeRotor("ABCDEFGHIJKLMNOPQRSTUVWXYZ", 3));
        assertEquals("SMDVHNQOIVHZIFHUQ", positionInitialeRotor("IFHUQSMDVHNQOIVHZ", 5));
    }
```

```
    // ----------------------------------------------------
    //  Fonction test pour indiceLettre
    // ----------------------------------------------------
    void testIndiceLettre() {
        assertEquals(2, indiceLettre('C',"ABCDE"));
        assertEquals(0, indiceLettre('A',"ABCDE"));
        assertEquals(4, indiceLettre('E',"ABCDE"));
        assertEquals(-1, indiceLettre('F',"ABCDE"));
        assertEquals(3, indiceLettre('F',ROTOR1));
    }
```

```
// ------------------------------------------------------
//  Fonction test pour valeurApresCablageDeDepart
// ------------------------------------------------------
void testValeurApresCablageDeDepart() {
    assertEquals('O', valeurApresCablageDeDepart('H',"AVDEHOJKLSXQ"));
    assertEquals('B', valeurApresCablageDeDepart('A',"ABCDEFGHIJKL"));
    assertEquals('A', valeurApresCablageDeDepart('B',"ABCDEFGHIJKL"));
    assertEquals('L', valeurApresCablageDeDepart('K',"ABCDEFGHIJKL"));
    assertEquals('C', valeurApresCablageDeDepart('D',"ABCDEFGHIJKL"));
    assertEquals('M', valeurApresCablageDeDepart('M',"ABCDEFGHIJKL"));
}
```

```
// ------------------------------------------------------
//  Fonction test pour passageDansLeReflecteur et passageDansUnRotor
// ------------------------------------------------------
void testPassageDansUnRotorOuReflecteur() {
    assertEquals('E', passageDansUnRotor('A', ROTOR1));
    assertEquals('K', passageDansUnRotor('B', ROTOR1));
    assertEquals('J', passageDansUnRotor('Z', ROTOR1));
    assertEquals('S', passageDansUnRotor('E', "AJDKSIRUXBLHWTMCQGZNPYFVOE"));

    assertEquals('Y', passageDansLeReflecteur('A', REFLECTEURA));
    assertEquals('R', passageDansLeReflecteur('B', REFLECTEURA));
    assertEquals('T', passageDansLeReflecteur('Z', "YRUHQSLDPXNGOKMIEBFZCWVJAT"));
}
```

```
// ------------------------------------------------------
//  Fonction test pour inverseRotor
// ------------------------------------------------------
void testInverseRotor() {
    assertEquals('A', inverseRotor('E', ROTOR1));
    assertEquals('B', inverseRotor('K', ROTOR1));
    assertEquals('Z', inverseRotor('J', ROTOR1));
    assertEquals('E', inverseRotor('S', "AJDKSIRUXBLHWTMCQGZNPYFVOE"));
}
```

```
// ------------------------------------------------------
//  Fonction test pour enMajuscule
// ------------------------------------------------------
void testEnMajuscule() {
    assertEquals("HELLO WORLD!", enMajuscule("hello world!"));
    assertEquals("ENIGMA", enMajuscule("EnIgMa"));
    assertEquals("ALAN TURING", enMajuscule("alan turing"));
}
```

```
No function 'void algorithm()' found, will try to run tests ...
    1> testConversionsLettresNombres
    2> testChoixRotor
    3> testDecalageRotors
    4> testIndiceLettre
    5> testValeurApresCablageDeDepart
    6> testPassageDansUnRotorOuReflecteur
    7> testInverseRotor
    8> testEnMajuscule
    8 test(s) verified on 8 tests (100% success).
```