МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЛЬВІВСЬКА ПОЛІТЕХНІКА

Автоматизоване проектування комп'ютерних систем

Task 5. Implement automated tests

Виконав:
ст. гр КІ - 401
Гербей О.М.

Прийняв: Федак П. Р.

2024

Опис теми

Для виконання завдання №4 потрібно виконати наступні задачі:

1. Впровадити або використовувати існуючу тестову структуру;
2. Створити набір автоматизованих тестів;
3. Звіт про тестування повинен містити кількість усіх тестів, складених тестів, нескладених тестів, покриття;
4. Покриття повинно бути більше 80%

Теоретичні відомості

Automated tests — це тести, які виконуються автоматично за допомогою спеціальних інструментів або скриптів без необхідності ручного втручання. Вони використовуються для перевірки функціональності програмного забезпечення, щоб забезпечити його якість, швидкість і точність тестування, а також для автоматичного виявлення помилок.

AUnit — це популярний фреймворк для написання та виконання автоматизованих тестів для платформ Arduino. Він підтримує простий і зрозумілий синтаксис, що нагадує Google Test, та дозволяє легко створювати юніт-тести для функцій, класів або бібліотек, які працюють безпосередньо на мікроконтролерах. AUnit забезпечує зручний механізм для організації тестів, а також надає інструменти для перевірки результатів, допомагаючи виявляти помилки в логіці програмування вже на етапі розробки. Фреймворк інтегрується в стандартне середовище Arduino IDE та підтримує гнучкі можливості для налаштування тестового середовища прямо на мікроконтролері.

Виконання завдання

1. Написав автоматизовані тести:

test.ino

```
#include <AUnit.h>
#include <EEPROM.h>

test(EvaluateFunctionTest) {
    char testBoard1[3][3] = {
        {'X', 'X', 'X'},
        {' ', 'O', ' '},
        {' ', ' ', 'O'}
    };
    assertEqual(evaluate(testBoard1), -1);

    char testBoard2[3][3] = {
        {'O', 'O', 'O'},
        {' ', 'X', ' '},
        {' ', ' ', 'X'}
    };
    assertEqual(evaluate(testBoard2), 1);

    char testBoard3[3][3] = {
        {'X', 'O', 'X'},
        {'O', 'O', 'X'},
        {'X', 'X', 'O'}
    };
    assertEqual(evaluate(testBoard3), 0);
}
test(ResetBoardTest) {
    resetBoard();
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            assertEqual(board[i][j], ' ');
        }
    }
    assertEqual(moveCount, 0);
    assertFalse(gameOver);
}
test(AIMoveTest) {
    Serial.println("BlockingMove...");
    resetBoard();
    board[0][0] = 'X';
    board[1][1] = 'X';
    board[1][2] = 'O';

    makeAIMove();
    assertEqual(board[2][2], 'O');
```

```
    Serial.println("ForkMove...");
    resetBoard();
    board[1][0] = 'X';
    board[1][1] = 'O';
    board[0][0] = 'O';
    board[2][2] = 'X';

    makeAIMove();
    assertEqual(board[0][1], 'O');

    Serial.println("ForkMove2...");
    resetBoard();
    board[1][1] = 'O';
    board[1][2] = 'X';
    board[2][2] = 'O';
    board[0][0] = 'X';

    makeAIMove();
    assertEqual(board[2][0], 'O');

    Serial.println("WinningMove...");
    resetBoard();
    board[0][0] = 'X';
    board[1][1] = 'O';
    board[2][2] = 'X';
    board[0][1] = 'O';

    makeAIMove();
    assertEqual(board[2][1], 'O');

    Serial.println("Best pos...");
    resetBoard();
    makeAIMove();
    assertEqual(board[1][1], 'O');
}

test(ProcessCommandTest) {
    strcpy(receivedData, "reset");
    processCommand();
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            assertEqual(board[i][j], ' ');
        }
    }
    strcpy(receivedData, "BLed");
    processCommand();
    assertTrue(blueLedState);
}
```

```
test(MinimaxBlockTest) {
    char testBoard[3][3] = {
        {'X', 'X', ' '},
        {'O', ' ', ' '},
        {' ', ' ', ' '}
    };
    int score = minimax(testBoard, 0, true, -1000, 1000);
    assertEqual(score, 0);
}

test(MinimaxDrawTest) {
    char testBoard[3][3] = {
        {'O', 'X', 'O'},
        {'X', 'O', 'X'},
        {'X', 'O', ' '}
    };
    int score = minimax(testBoard, 0, true, -1000, 1000);
    assertEqual(score, 0);
}
```

2. Звіт про виконаття тестів:

```
3. →
4. Симуляція
5. ONL TX RX  DIGITAL (PWM
   ~)AREF GND 13 12 ~11 ~10 ~9 8 7 ~6 ~5 4 ~3 2 TX→1 RX←0  POWER ANALOG
   INIOREF RESET 3.3V 5V GND GND Vin A0 A1 A2 A3 A4 A5  ARDUINOUNO
6.
7. Starting AUnit tests...
8. TestRunner started on 6 test(s).
9. Starting AUnit tests...
10. BlockingMove...
11.
12. /tmp/build-Fme7SM/sketch/test.ino:50: Assertion passed: (O) == (O).
13. ForkMove...
14.
15. /tmp/build-Fme7SM/sketch/test.ino:60: Assertion passed: (O) == (O).
16. ForkMove2...
17.
18. /tmp/build-Fme7SM/sketch/test.ino:70: Assertion passed: (O) == (O).
19. WinningMove...
20.
21. /tmp/build-Fme7SM/sketch/test.ino:80: Assertion passed: (O) == (O).
22. Best pos...
23.
24. /tmp/build-Fme7SM/sketch/test.ino:85: Assertion passed: (O) == (O).
25. Starting AUnit tests...
26. Starting AUnit tests...
27. Test AIMoveTest passed.
28. Starting AUnit tests...
29. Starting AUnit tests...
30. /tmp/build-Fme7SM/sketch/test.ino:12: Assertion passed: (-1) == (-1).
31. /tmp/build-Fme7SM/sketch/test.ino:19: Assertion passed: (1) == (1).
32. /tmp/build-Fme7SM/sketch/test.ino:26: Assertion passed: (0) == (0).
33. Starting AUnit tests...
34. Starting AUnit tests...
35. Test EvaluateFunctionTest passed.
36. Starting AUnit tests...
37. Starting AUnit tests...
```

38. /tmp/build-Fme7SM/sketch/test.ino:111: Assertion passed: (0) == (0).
39. Starting AUnit tests...
40. Starting AUnit tests...
41. Test MinimaxBlockTest passed.
42. Starting AUnit tests...
43. Starting AUnit tests...
44. /tmp/build-Fme7SM/sketch/test.ino:122: Assertion passed: (0) == (0).
45. Starting AUnit tests...
46. Starting AUnit tests...
47. Test MinimaxDrawTest passed.
48. Starting AUnit tests...
49. Starting AUnit tests...
50.
51. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
52. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
53. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
54. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
55. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
56. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
57. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
58. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
59. /tmp/build-Fme7SM/sketch/test.ino:94: Assertion passed: ( ) == ( ).
60. /tmp/build-Fme7SM/sketch/test.ino:100: Assertion passed: (true) is true.
61. Starting AUnit tests...
62. Starting AUnit tests...
63. Test ProcessCommandTest passed.
64. Starting AUnit tests...
65. Starting AUnit tests...
66.
67. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
68. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
69. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
70. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
71. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
72. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
73. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
74. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
75. /tmp/build-Fme7SM/sketch/test.ino:34: Assertion passed: ( ) == ( ).
76. /tmp/build-Fme7SM/sketch/test.ino:37: Assertion passed: (0) == (0).

```
77. /tmp/build-Fme7SM/sketch/test.ino:38: Assertion passed: (false) is false.
78. Starting AUnit tests...
79. Starting AUnit tests...
80. Test ResetBoardTest passed.
81. Starting AUnit tests...
82. TestRunner duration: 3.228 seconds.
83. TestRunner summary: 6 passed, 0 failed, 0 skipped, 0 timed out, out of 6 test(s).
```

Висновок

Під час виконання завдання №5 було написано автоматизовані тетси та згенеровано звіти.

Список використаних джерел

1. Wikipedia. "Automated testing". https://en.wikipedia.org/wiki/Automated_testing.
2. ArduinoUnit https://github.com/mmurdoch/arduinounit/blob/master/README.md.