Національний технічний університет України
«Київський політехнічний університет імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра Системного проектування

Лабораторна робота № 5
з предмету «Проектування інформаційних систем»

Виконав:
студент групи ДА-72
Кулик В.О.

Київ – 2020

Мета роботи: оволодіти навичками створення програмного забезпечення за метолологією TDD та ознайомитися з процедурами рефакторинга.

Хід роботи

1. Створимо функції для тестування.

На даному етапі нам потрібні функції, що будуть вираховувати кількість випитої користувачем води на даний момент, різницю між поточною кількістю води та денною нормою, а також частоту відправки повідомлень.

index.ts

index.ts > calculateNotificationFrequency

```
1   const calculateCurrentAmountOfWaterDrunk = (
2     prevAmount,
3     income,
4     globalAmount
5   ) => {
6     return 0;
7   };
8
9   const calculateWaterTillGlobal = (currAmount, global) => {
10    return 0;
11  };
12
13  const calculateNotificationFrequency = (notifications) => {
14    return 0;
15  };
16
17  module.exports = {
18    calcCurrWaterDrunk: calculateCurrentAmountOfWaterDrunk,
19    calcWaterTillGlobal: calculateWaterTillGlobal,
20    calcNotFreq: calculateNotificationFrequency,
21  };
```

2. Використовуючи парадигму TDD, створимо тести для даних функцій, що відображатимуть очікуваний функціонал:

index.ts | index.spec.ts | package.json

index.spec.ts > ...

```
1   const index = require('./index');
2   const waterDrunk = index.calcCurrWaterDrunk;
3   const waterTillGlobal = index.calcWaterTillGlobal;
4   const calcNotFreq = index.calcNotFreq;
5
6   test('should calculate amount of water drunk', () => {
7     expect(waterDrunk(2, 1, 6)).not.toBeFalsy();
8     expect(waterDrunk(2, 1, 6)).toBeGreaterThanOrEqual(3);
9     expect(waterDrunk(3, 2, 4)).toBe(4);
10    expect(waterDrunk(3, 2, 9)).toBe(5);
11  });
12
13  test('should calculate difference between current water level and daily global
    lvl', () => {
14    expect(waterTillGlobal(5, 9)).toBe(4);
15    expect(waterTillGlobal(8, 5)).toBe(0);
16    expect(waterTillGlobal(6, 6)).toBeGreaterThanOrEqual(0);
17  });
18
19  test('should calculate notifications frequency depending on user input data', ()
    => {
20    expect(calcNotFreq(5)).toBeTruthy();
21    expect(calcNotFreq('five')).toBeFalsy();
22    expect(calcNotFreq(6)).toBe(2);
23    expect(calcNotFreq(3.4)).toBe(4);
24  });
```

3. Запустимо дані тести:

```
FAIL ./index.spec.ts
  ✕ should calculate amount of water drunk (3 ms)
  ✕ should calculate difference between current water level and daily global lvl (1 ms)
  ✕ should calculate notifications frequency depending on user input data (1 ms)

  ● should calculate amount of water drunk

    expect(received).not.toBeFalsy()

    Received: 0

       5 |
       6 | test('should calculate amount of water drunk', () => {
    >  7 |   expect(waterDrunk(2, 1, 6)).not.toBeFalsy();
         |                                   ^
       8 |   expect(waterDrunk(2, 1, 6)).toBeGreaterThanOrEqual(3);
       9 |   expect(waterDrunk(3, 2, 4)).toBe(4);
      10 |   expect(waterDrunk(3, 2, 9)).toBe(5);

      at Object.<anonymous> (index.spec.ts:7:35)

  ● should calculate difference between current water level and daily global lvl

    expect(received).toBe(expected) // Object.is equality

    Expected: 4
    Received: 0

      12 |
      13 | test('should calculate difference between current water level and daily global lvl', () => {
    > 14 |   expect(waterTillGlobal(5, 9)).toBe(4);
         |                                 ^
      15 |   expect(waterTillGlobal(8, 5)).toBe(0);
      16 |   expect(waterTillGlobal(6, 6)).toBeGreaterThanOrEqual(0);
      17 | });

      at Object.<anonymous> (index.spec.ts:14:33)

  ● should calculate notifications frequency depending on user input data

    expect(received).toBeTruthy()

    Received: 0

      18 |
      19 | test('should calculate notifications frequency depending on user input data', () => {
    > 20 |   expect(calcNotFreq(5)).toBeTruthy();
         |                          ^
      21 |   expect(calcNotFreq('five')).toBeFalsy();
      22 |   expect(calcNotFreq(6)).toBe(2);
      23 |   expect(calcNotFreq(3.4)).toBe(4);

      at Object.<anonymous> (index.spec.ts:20:26)

Test Suites: 1 failed, 1 total
Tests:       3 failed, 3 total
Snapshots:   0 total
```

Як можемо побачити, всі тести не пройшли.

4. Додамо логіку дод методів, що буде задовільняти умовам тестів:

index.ts

```ts
const calculateCurrentAmountOfWaterDrunk = (
  prevAmount,
  income,
  globalAmount
) => {
  let afterIncome;
  if (prevAmount + income > globalAmount) {
    afterIncome = globalAmount;
  } else {
    afterIncome = prevAmount + income;
  }
  return afterIncome;
};

const calculateWaterTillGlobal = (currAmount, global) => {
  let difference;
  if (currAmount >= global) {
    difference = 0;
  } else {
    difference = global - currAmount;
  }
  return difference;
};

const calculateNotificationFrequency = (notifications) => {
  let frequency;
  if (typeof notifications === 'number' && notifications > 0) {
    frequency = Math.ceil(12 / notifications);
  } else {
    frequency = false;
  }
  return frequency;
};
```

5. Перевіримо тести:

```
PASS ./index.spec.ts
  ✓ should calculate amount of water drunk (2 ms)
  ✓ should calculate difference between current water level and daily global lvl
  ✓ should calculate notifications frequency depending on user input data (1 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        1.021 s
Ran all test suites.
```

6. Проведемо рефакторинг, згідно якого перепишемо внутрішню логіку функцій таким чином, щоб зробити код більш легким та зрозумілим (використаємо тернарні оператори та const об'явлення для змінних, що буде більш коректним згідно конвенції коду):

index.ts

index.ts > calculateNotificationFrequency

```ts
const calculateCurrentAmountOfWaterDrunk = (
  prevAmount,
  income,
  globalAmount
) => {
  const afterIncome =
    prevAmount + income > globalAmount ? globalAmount : prevAmount + income;
  return afterIncome;
};

const calculateWaterTillGlobal = (currAmount, global) => {
  const difference = currAmount >= global ? 0 : global - currAmount;
  return difference;
};

const calculateNotificationFrequency = (notifications) => {
  const frequency =
    typeof notifications === 'number' && notifications > 0
      ? Math.ceil(12 / notifications)
      : false;
  return frequency;
};
```

Проведемо тести:

```
PASS ./index.spec.ts
  ✓ should calculate amount of water drunk (3 ms)
  ✓ should calculate difference between current water level and daily global lvl
  ✓ should calculate notifications frequency depending on user input data (1 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        0.971 s, estimated 1 s
Ran all test suites.
```

Висновок:

В ході виконання лабораторної роботи було досліджено методологію TDD. Написані модульні тести, відповідно до них створений функціонал та проведений рефакторинг до коду проекту. Дані процеси покликані зробити розробку проекту більш надійною та менш ресурсопотребуючою.