

# Упражнение: Нива на тестване и типове тестове

Упражнения и задачи за домашна работа към курса ["Основи на QA" @СофтУни](#).

Попълнете предоставения темплейт (**MS Word документ**). Поставете решенията си на съответните за всяка задача места. Преименувайте документа на **Test-Levels-Homework-FirstName-LastName.docx**. Прикачете документа като домашна работа.

## 1. Юнит тестване в реалния живот: Тестване на батерия

Дадени са ви няколко батерии 1,5V AA:



Как бихте могли да проверите дали батериите **работят според очакванията**? Попълнете в таблицата, какви тестове ще изпълните.

### Подсказки

- Проверете с волтметър или мултицет.
- Проверете **размера на батерията** (височина + диаметър). Съответства ли на стандарта за размер "AA"?
- Проверете **напрежението** на батерията: измерете напрежението с помощта на цифров мултиметър.
- Проверете **физическото състояние** на батерията: има ли повреди?
- Проверете **етикетите** на батерията. Точни ли са?

## 2. Юнит тестване в реалния живот: Тестване на крушка

Дадена ви е **1.5V крушка E10**:



Как можете да проверите дали **работи според очакванията**? Попълнете в таблицата, какви тестове ще изпълните.

## 3. Юнит тестване в света на софтуера: Проверка на възрастта

Нека си представим **функция** за "проверка на възрастта" (AgeChecke), която работи по следния начин:

- Ако възрастта е под 13 години, връща "дете"(child)
- Ако възрастта е между 13 (вкл.) и 19 (вкл.), връща "тийнейджър" (teenager)
- Ако възрастта е между 20 (вкл.) и 64 (вкл), връща "възрастен" (adult)
- Ако възрастта е равна или по-голяма от 65, връща "по-възрастен" (elder)
- Ако възрастта е отрицателна или над 150, връща "грешка"(error)

Ето няколко примера:

- AgeChecker(5) → child
- AgeChecker(19.5) → teenager
- AgeChecker(20) → adult

- AgeChecker(75.3) → elder
- AgeChecker(-5) → error

Какви биха били юнит тестовите за тази функция? Попълнете в таблицата тестовите, които ще изпълните.

Можете да разгледате функцията за проверка на възрастта тук: <http://softuni-qa-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/age-checker/>. Изпълнете тестове и докладвайте резултатите: **pass** / **fail** (успешно / неуспешно).

## Подсказки

Помислете за тест случаи, с които да **покриете всяка възможна възрастова група**. Помислете за **гранични случаи**.

## 4. Юнит тестване в света на софтуера: Проверка на доходите

Нека си представим **функция за проверка на дохода**, предназначена да категоризира определен **месечен доход** в една от следните категории: "нисък", "среден", "висок". Функцията за проверка на доходите работи по следния начин:

- Ако доходът е по-малък от 1000, връща "нисък" (low)
- Ако доходът е между 1000 (включително) и 2999, връща "среден" (mid)
- Ако доходът е равен или по-голям от 3000, връща "висок" (high)
- Ако доходът е отрицателен, връща "грешка" (error)

Ето няколко примера:

- IncomeChecker(250) → low
- IncomeChecker(1000) → mid
- IncomeChecker(2300.70) → mid
- IncomeChecker(7000) → high
- IncomeChecker(-5) → error

Какви биха били юнит тестовите за тази функция? Попълнете в таблицата тестовите, които ще изпълните.

Можете да разгледате функцията за проверка на доходите тук: <http://softuni-qa-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/income-checker/>. Изпълнете тестовите и докладвайте резултатите: **pass** / **fail** (успешно / неуспешно).

## 5. \* Интеграционно тестване в реалния свят: Запалване на крушката

Дадени са ви се следните компоненти:

- 1.5V AA батерия
- 1.5V ел. крушка E10
- Бутон за превключване
- Свързващи проводници

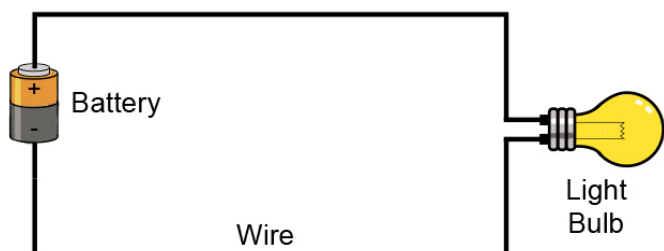
Всички компоненти вече са **тествани поотделно** и работят според очакванията.

Съставете **интеграционни тестове**, които да **проверяват дали крушката, батериите, бутонът за превключване и проводниците работят правилно заедно като една електрическа верига**.

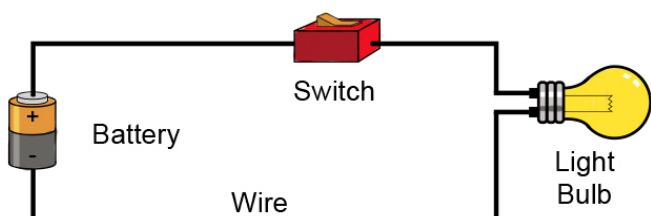


## Подсказки

Първият и най-прост интеграционен тест може да използва следната електрическа верига:



Вторият интеграционен тест може да включва бутон за превключване. Може да се използва следната електрическа верига:

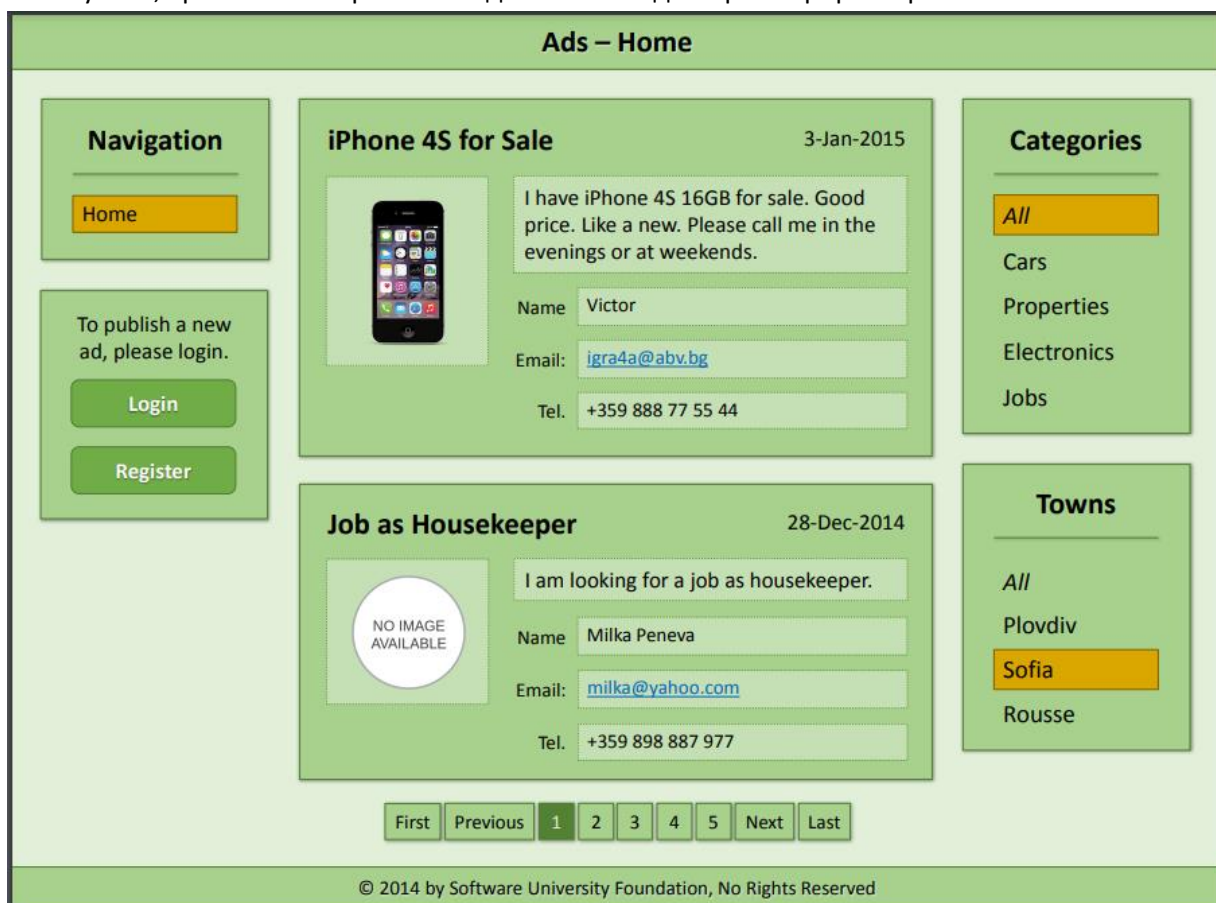


Тествайте и **отрицателни** случаи.

## 6. \* Интеграционно тестване в софтуерния свят: Обяви

Предоставено ви е уеб приложение за публикуване на обяви. То се състои от:

- **Начална страница (Home Page):** показва всички публикувани обяви. Потребителят може да преглежда обявите по категории и/или по градове. За да публикува обява, потребителят трябва да е регистриран. Има бутони, чрез които потребителят да влезе или да се регистрира в приложението.



- **Страница за влизане (Login Page):** осъществява влизането на потребителите. Потребителят може да влезе, като предостави правилно потребителско име и парола. Има и линк към формата за регистрация.

- **Начална страница на потребителя (User Home Page):** достъпна след успешно влизане в приложението → показва всички публикувани обяви. Потребителят може да преглежда обявите по категории и/или по градове. Има потребителско поле за навигация с опции за публикуване на нова обява, за преглед на собствените публикувани обяви и за редактиране на личния профил. Има бутон "Изход".

Горните страници са различни **компоненти** на приложението и вече **са тествани поотделно**. Вашата задача е да създадете интеграционни тестове, които проверяват дали **горните 3 компонента работят правилно заедно**.

## Подсказки

- Можете да проверите дали след влизане началната страница на потребителя се показва правилно.
- Помислете какво трябва да се случи, ако някой се опита да влезе с невалидни потребителски данни.
- Помислете какво трябва да се случи след излизане.

## 7. \* Интеграционно тестване в софтуерния свят: Кредитен риск

Разполагате с приложение за изчисляване на **кредитния риск** въз основа на **възрастта** и **дохода** на клиента.

<http://softuni-ga-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/credit-risk/>

То използва следните, вече **съществуващи компоненти** (функции):

- **AgeChecker**(age) → child / teenager / adult / elder
- **IncomeChecker**(monthlyIncome) → low / mid / high

Всички компоненти вече са тествани **поотделно** и работят според очакванията.

**Кредитният риск** се изчислява по следните формули:

- Възрастов риск =
  - дете → 100%, тинейджър → 60%, възрастен → 10%, по-възрастен → 20%
- Риск на дохода =
  - нисък → 50%, среден → 30%, висок → 10%
- Кредитен риск =
  - $\text{възрастовРиск} + \text{рискНаДохода} - (\text{възрастовРиск} * \text{рискНаДохода})$
- Age risk =
  - child → 100%, teenager → 60%, adult → 10%, elder → 20%
- Income risk =
  - low → 50%, mid → 30%, high → 10%
- Credit risk =
  - $\text{ageRisk} + \text{incomeRisk} - (\text{ageRisk} * \text{incomeRisk})$

**Примери:**

- Кредитният риск за **14-годишно** лице със **700 месечен доход** се изчислява по следния начин:
  - ПроверкаНаВъзрастта(14) → тинейджър → **възрастовРиск = 60%**
  - ПроверкаНаДохода(700) → нисък → **рискНаДохода = 50%**
  - Кредитен риск =  $60\% + 50\% - (60\% * 50\%) = 110\% - 30\% = 80\%$
- Кредитният риск за **85-годишно** лице с **1600 месечен доход** се изчислява, както следва:
  - ПроверкаНаВъзрастта(85) → по-възрастен → **възрастовРиск = 20%**
  - ПроверкаНаДохода(1600) → среден → **рискНаДохода = 30%**
  - Кредитен риск =  $20\% + 30\% - (20\% * 30\%) = 50\% - 6\% = 44\%$
- Кредитният риск за **30-годишно** лице с **3500 месечен доход** се изчислява, както следва:
  - ПроверкаНаВъзрастта(30) → възрастен → **възрастовРиск = 10%**
  - ПроверкаНаДохода (3500) → висок → **рискНаДохода = 10%**
  - Кредитен риск =  $10\% + 10\% - (10\% * 10\%) = 20\% - 1\% = 19\%$

- Кредитният риск за **20-годишно** лице с **-50 месечен доход** дава **"грешка"**, тъй като доходът не може да бъде отрицателен.

Съставете **интеграционни тестове**, които проверяват дали калкулаторът за кредитния риск работи според очакванията. Попълнете в таблицата какви тестове ще изпълните.

Изпълнете тестовете и докладвайте резултатите: **pass** / **fail** (успешно / неуспешно).

## 8. Системно тестване в реалния живот: Фенерче

Дадено ви е класическо **електрическо фенерче**. Създайте набор от **системни тестове**, за да тествате устройството.

### Подсказки

- Тествайте включване / изключване на светлината
- Тествайте смяната на батерията
- Пробна смяна на крушката
- Тествайте колко издържа батерията
- Тествайте разстоянието на осветяване
- Тест за устойчивост на удар
- Работа при висока/ниска температура



## 9. Системно тестване в реалния живот:

### Дигитален кантар

Даден ви е класически **дигитален кантар** за измерване на телесно тегло.

Съставете **системни тестове**, за да тествате устройството.



## 10. Системно тестване в софтуерния свят: Калкулатор на числа

Дадено ви е **приложение "Калкулатор на числа"**:

<http://softuni-ga-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/number-calculator/>

Number Calculator

First Number:

Operation:

Second Number:

Result: 8

Number Calculator

First Number:

Operation:

Second Number:

Result: *invalid operation*

Създайте **системни тестове**, за да тествате приложението. Изпълнете тестовете и докладвайте резултатите: **pass** / **fail** (успешно / неуспешно).



## Подсказки:

Можете да обмислите тестване с:

- Цели числа
- Десетични числа
- Експоненциални числа
- Безкрайност
- Много големи числа
- Много малки числа
- Невалидни въведени данни
- Невалидни операции

## 11. Приемно тестване в реалния живот: Фенерче

Получавате класическо **електрическо фенерче**. Напишете тестове за приемане от крайния потребител. Как ще тествате устройството от гледна точка на клиента?

## Подсказки:

- Клиентът може да вземе фенерчето, **да включи / изключи** светлината и да се увери, че работи.
- Клиентът може да провери **доколко силна е светлината**
- Клиентът може да провери колко лесно се **сменят батериите**.

## 12. Приемно тестване в реалния живот: Дигитален кантар

Даден ви е класически **дигитален кантар** за измерване на телесно тегло.

Напишете тестове за приемане от крайния потребител. Как ще тествате устройството от гледна точка на клиента?



## 13. Приемно тестване в софтуерния свят: Калкулатор на числа

Дадено ви е **приложение "Калкулатор на числа"**:

<http://softuni-ga-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/number-calculator/>

Number Calculator

First Number:

Operation:

Second Number:

Result: 8

Number Calculator

First Number:

Operation:

Second Number:

Result: *invalid operation*

Създайте тестове за приемане от крайния потребител. **Изпълнете** тестовете и **докладвайте резултатите**: **pass** / **fail** (успешно / неуспешно).

## 14. Функционални и нефункционални тестове: Фенерче

Разгледайте всички тестове, които сте съставили за фенерчето в предните упражнения (системно тестване и приемно тестване).



Можете ли да ги разделите на **функционални** и **нефункционални**?