Automation Test

What is Automated testing?

- **Automated testing: The management and performance of test activities, to include the development and execution of test scripts having as objective the verification of test requirements, using an automated test tool.**
- **The automation of test activities reveals its greatest values in instances where test scripts are repeated or where test scripts subroutines are created and then invoked repeatedly by a number of test scripts.**
- **Given the continual changes and additions to requirements and software, automated tests serves as an important control mechanism to ensure accuracy and stability of the software through each build.**

Automated testing advantages:

- **Costs and efficiency**
    - **Detection of the errors that reached production phase (with regression tests)**
    - **Multiple users simulation**
    - **Reusable of the old scripts -> creation of the new scripts is reduce**
    - **automatic execution of performance tests in the beginning of the production ->less costs for improving the performance**
- **Time economy**
    - **quick analysis in case of changing of environment parameters**
    - **short duration of the testing cycles**
    - **better estimation for test planning**

    - **a large number of tests can be executed over night**
    - **quick generation of testing preconditions**
- **Quality increase**
    - **automatic compare of results**
    - **more consistent results due to repeating tests**

Limitation of Automated Testing:

- **Most of the times an Automated Testing system can't tell if something "looks good" on the screen or when a pictogram or a window is not displayed well**
- **There are a bunch of problems that can appear when trying to automate the testing process:**
    - **Unrealistic expectations (e.g. expectation that automated tests will find a lot of errors)**
    - **Poor testing experience**
    - **Maintenance of automated tests**
- **Automated testing will never replace definitely the manual testing**
- **Tests that should not be automated are:**
    - **tests that are executed very rare**
    - **where the system is very unstable**
    - **tests that can be verified easily manually but hardly automated**
    - **tests that need physical interaction**

Automated testing vs. Manual testing:

- **Pros of Automated testing**
    - **If a set of tests must be ran repeatedly, automation is a huge win**
    - **It offers the possibility to run automation against code that frequently change to catch regressions**
    - **Offers the possibility to add a large test matrix (e.g. different languages on different OS platforms)**
    - **Automated tests can be run the same time on different machines, whereas manual tests must be run sequentially**
    - **It offers more time for the test engineer to invoke greater depth and breadth of testing, focus on problem analysis, and verify proper performance of software following modifications and fixes**
    - **Combined with the opportunity to perform programming tasks, this flexibility promotes test engineer retention and improves his morale**
- **Cons of Manual testing**
    - **Running tests manually can be very time consuming**
    - **The manual tests are requiring more people and hardware**
    - **Each time there is a new build, the tester must rerun all required tests - which after a while would become very boring and tiresome.**

Automation test for chess application:

In this example I try to make an automation test for "Get Suggested Moves" button using python as programming language and "selenium" as a tool for automating this test.

What Is Selenium?

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms. It is quite similar to HP Quick Test Pro (QTP now UFT) only that Selenium focuses on automating web-based applications. Testing done using Selenium tool is usually referred as Selenium Testing.

Selenium is not just a single tool but a suite of software's, each catering to different testing needs of an organization. It has four components.

- **Selenium Integrated Development Environment (IDE)**
- **Selenium Remote Control (RC)**
- **WebDriver**
- **Selenium Grid**

1. **We need a web driver: (in this example we use "Chrome")**
   After you choose this web driver, you need to set some attributes.

```
driver = webdriver.Chrome('./drivers/chromedriver.exe')

driver.get('http://127.0.0.1:5000/')
```

2. **We need to get the "Get Suggested Moves" button from web page using 'find_element_by_id()' method provided by selenium web driver. After we capture the button we can make some operation on it like button.click().**

```
generate_moves_btn = driver.find_element_by_id('js-get-moves')
generate_moves_btn.click()
```

3. **After we press the button “Get Suggested Moves” we need to find a way to get the result of the chess algorithm generator. For that we can also use the ‘find_element_by_id()’ method.**

```
alphabeta_prunning = driver.find_element_by_id('js-move-alphabeta_prunning')
hermann = driver.find_element_by_id('js-move-hermann')
ids_field = driver.find_element_by_id('js-move-ids')
minmax = driver.find_element_by_id('js-move-minmax')
negamax = driver.find_element_by_id('js-move-negamax')
ruffian = driver.find_element_by_id('js-move-ruffian')
rybka = driver.find_element_by_id('js-move-rybka')
sos = driver.find_element_by_id('js-move-sos')
spike = driver.find_element_by_id('js-move-spike')
stockfish = driver.find_element_by_id('js-move-stockfish')
```

4. **After all of these steps we can make assertions for the results**

```
try:
    assert alphabeta_prunning.text == 'e2e4'
    print(colored('Alphabeta_prunning field has been generated correctly', 'green'))
except AssertionError:
    print(colored('Alphabeta_prunning field was wrongly generated', 'red'))
try:
    assert hermann.text == 'e2e4'
    print(colored('Herman field has been generated correctly', 'green'))
except AssertionError:
    print(colored('Herman field was wrongly generated', 'red'))
try:
    assert ids_field.text == 'd2d4'
    print(colored('Ids field has been generated correctly', 'green'))
except AssertionError:
    print(colored('Ids field was wrongly generated', 'red'))
```