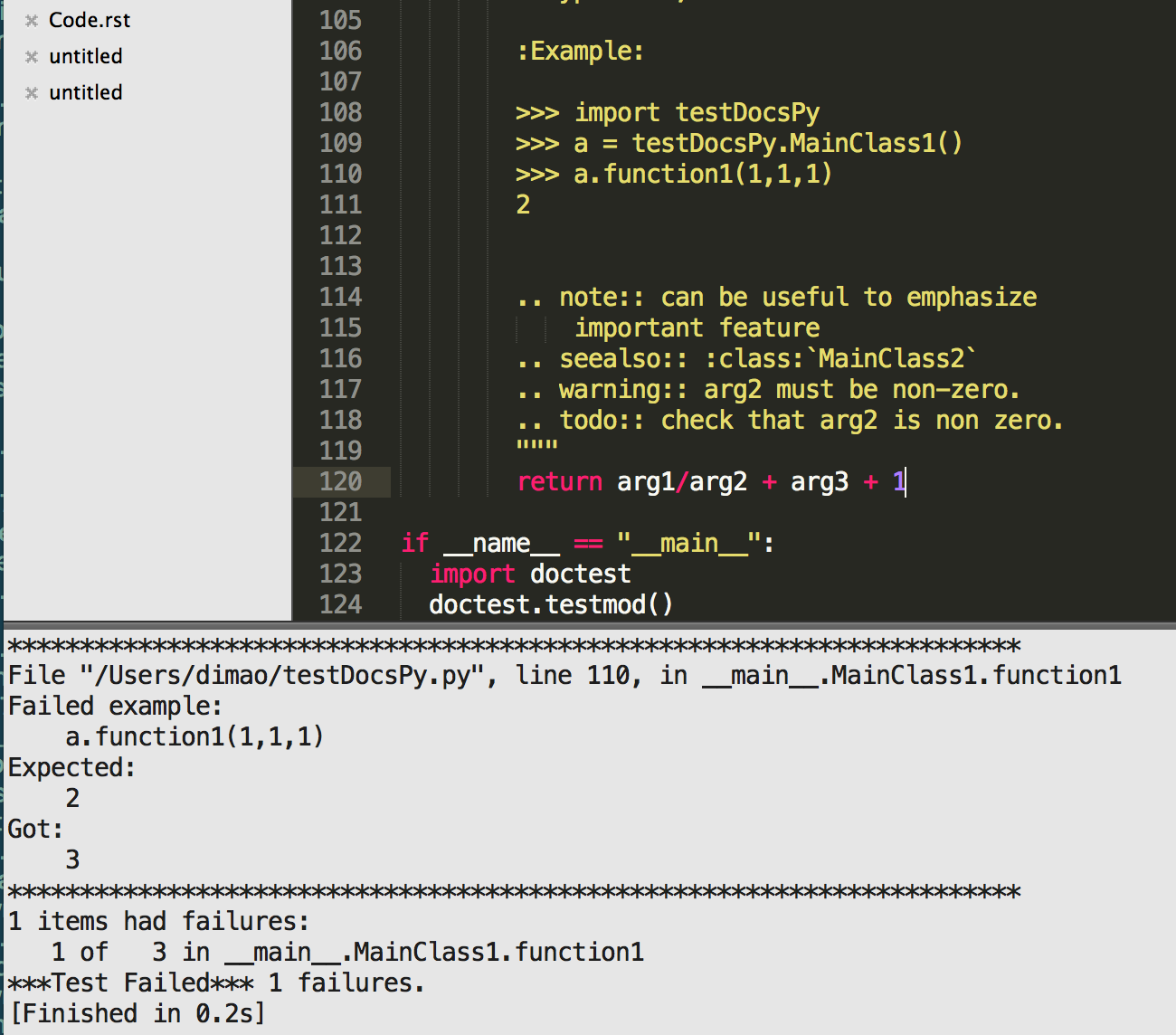
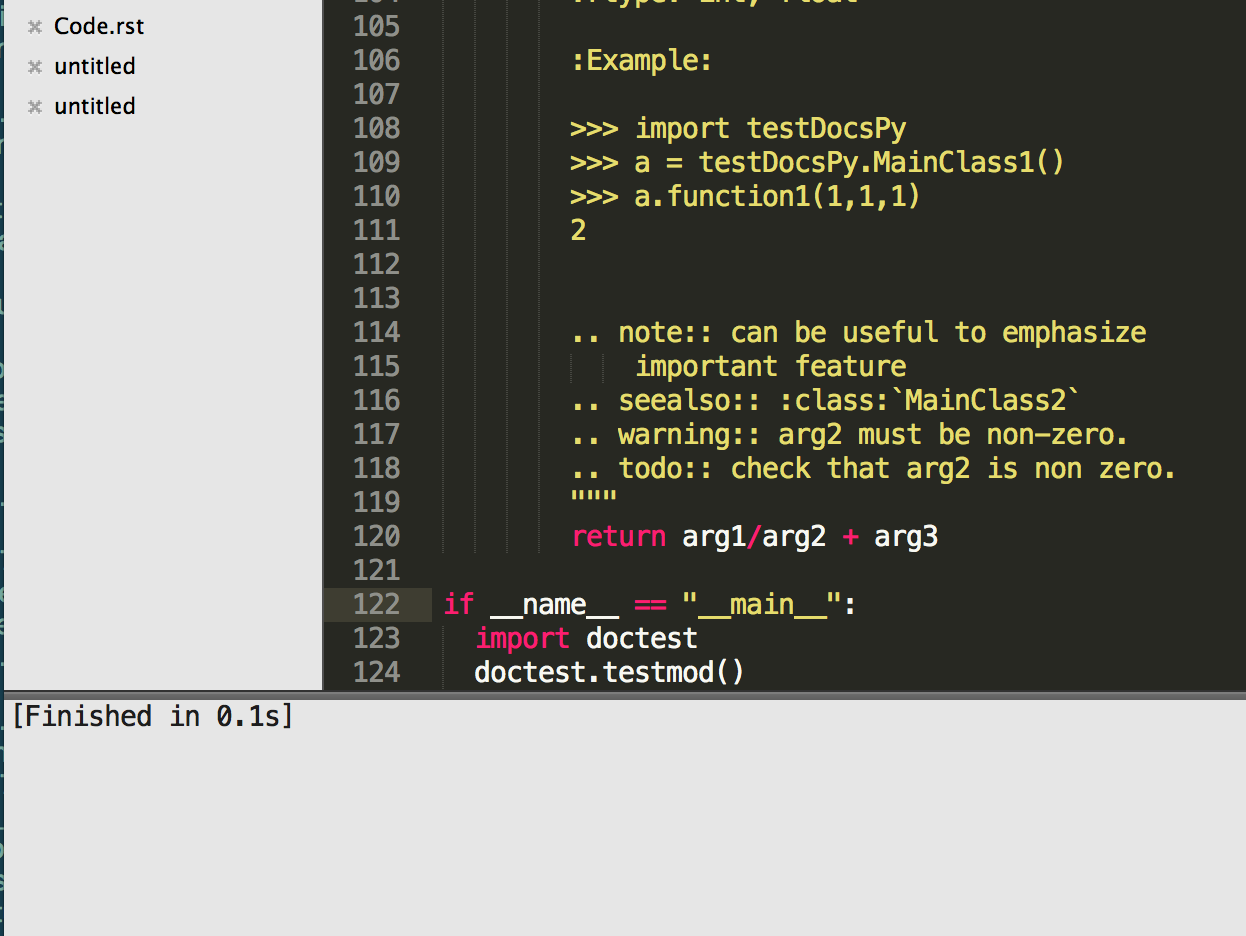
ЛАБОРАТОРНА РОБОТА No4

**Модульне тестуваня (Unit-тести) та рефакторинг.**  
**Мета роботи:** оволодіти навичками створення програмного забезпечення за метолологією

TDD та ознайомитися з процедурами рефакторинга. 



| Table 1 | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| №Теста | Название Сценария | Основной путь | Альтернативный путь | Процент выполнения | Результат | Код теста | Комментарий |
| **1** | Добавление нового пользователя | Создать нового пользователя -> ввести имя и ввести возраст -> вывод имя и возраста |  | 100% | Name: Gogo  Age: 12 | >>> import testDocsPy  >>> a = testDocsPy.User("Gogo", 12)  >>> print a | Тестирование прошло успешно, ошибок при выполнении не возникало |
| **2** |  |  | Создать нового пользователя -> оставить пустым имя -> вывод имя и возраста | 100% | Name: Mark  Age: 12 | >>> import testDocsPy  >>> a = testDocsPy.User(age=12)  >>> print a | Тестирование прошло успешно, ошибок при выполнении не возникало |
| **3** |  |  | Создать нового пользователя -> оставить пустым возраст -> вывод имя и возраста | 100% | Name: Gogo  Age: 21 | >>> import testDocsPy  >>> a = testDocsPy.User(“Gogo”)  >>> print a | Тестирование прошло успешно, ошибок при выполнении не возникало |
| **4** |  |  | Создать нового пользователя -> оставить пустым и имя и возраст -> вывод имя и возраста | 100% | Name: Mark  Age: 21 | >>> import testDocsPy  >>> a = testDocsPy.User()  >>> print a | Тестирование прошло успешно, ошибок при выполнении не возникало |
| **5** | Суммирование чисел | Пользователь вводит два числа—> ввывод результата |  | 100% | 2 | >>> import testDocsPy  >>> a = testDocsPy.MainClass1()  >>> a.sum(1,1) | Тестирование прошло успешно, ошибок при выполнении не возникало |
| **6** |  |  | Пользователь вводит одно число | 100% | 1 | >>> import testDocsPy  >>> a = testDocsPy.MainClass1()  >>> a.sum(1) | Тестирование прошло успешно, так как по умолчанию числа 0, то ошибок не возникало |
| **7** |  |  | Пользователь не вводит числа | 100% | 0 | >>> import testDocsPy  >>> a = testDocsPy.MainClass1()  >>> a.sum() | Тестирование прошло успешно, так как по умолчанию числа 0, то ошибок не возникало |

| Table 1-1 | |
| --- | --- |
| 1)  Test Plan Identifier | RS1.HUB1.0 |
|  |
| 2)  References | None Identified |
|  |
| 3)  Introduction | This is the plan for the laboratory work №4 where we should create test for our project, in my case it StockChart |
|  |
| 4)  Test Items | 1. Adding new user to system 2. Sum two numbers |
|  |
| 5)  Software Risk Issues | There are some features that out of control:   1. Sum operator in python can change definition with a probability 0.0008% 2. Software update python2.7.1 |
|  |
| 6)  Features to be Tested | 1. Adding user with name and age 2. Adding user with name and without age 3. Adding user with without name and age 4. Adding user with without name and without age 5. ….Sum 2 numbers |
|  |
| 7)  Features not to be Tested | Sum 3 numbers |
|  |
| 8)  Approach | The testing for the project will consist of Unit and System/Integration (combined). |
|  |
| 8.1) Test Tools | We are going to use library doctest in Python |
| 9)  Item Pass/Fail Criteria | The test will be completed when all features and items pass the test. |
|  |
| 10)  Suspension Criteria and Resumption Requirements | No code are ready for testing at pilot initiation.  The pilot project will be delayed until at one of the classes are ready to initiate the pilot process. |
|  |
| 11)  Test Deliverables | 1. Acceptance test plan 2. System test plan 3. Unit test plans/turnover documentation 4. Screen prototypes |
|  |
| 12)  Remaining Test Tasks | None |
|  |
| 13)  Environmental Needs | PC, Air |
|  |
| 14)  Staffing and Training Needs | Brain |
|  |
| 15)  Responsibilities | Class Sum, User - assign to John Galt  Testing - miss Jenny |
|  |
| 16)  Schedule | none |
|  |
| 17)  Planning Risks and Contingencies | The project staff has two position unfilled - Designer and DataBase Architecture |
|  |
| 18)  Approvals | John Galt - wizard  miss Jenny - manikin  Dima Osadchy - CEO |
|  |

**Conclusion: As you can see Unit Testing is a core of Test Driven Development, in order to decrease bugs amount it is a good solution to write tests for each function.**