Министерство образования и науки Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования Петрозаводский государственный университет Институт математики и информационных технологий Кафедра информатики и математического обеспечения

# Отчет по дисциплине «Верификация ПО»

Выполнил: студент группы 22407, Клименко В.В.

Преподаватель: к.ф-м.н., доцент К. А. Кулаков

Петрозаводск, 2020

# Оглавление

Объект тестирования

# Описание приложения

Объектом тестирования является текстовый редактор на языке Python. Данный редактор работает с Json файлами, использует загрузку по URL, имеет графический интерфейс и проводит Json валидацию.

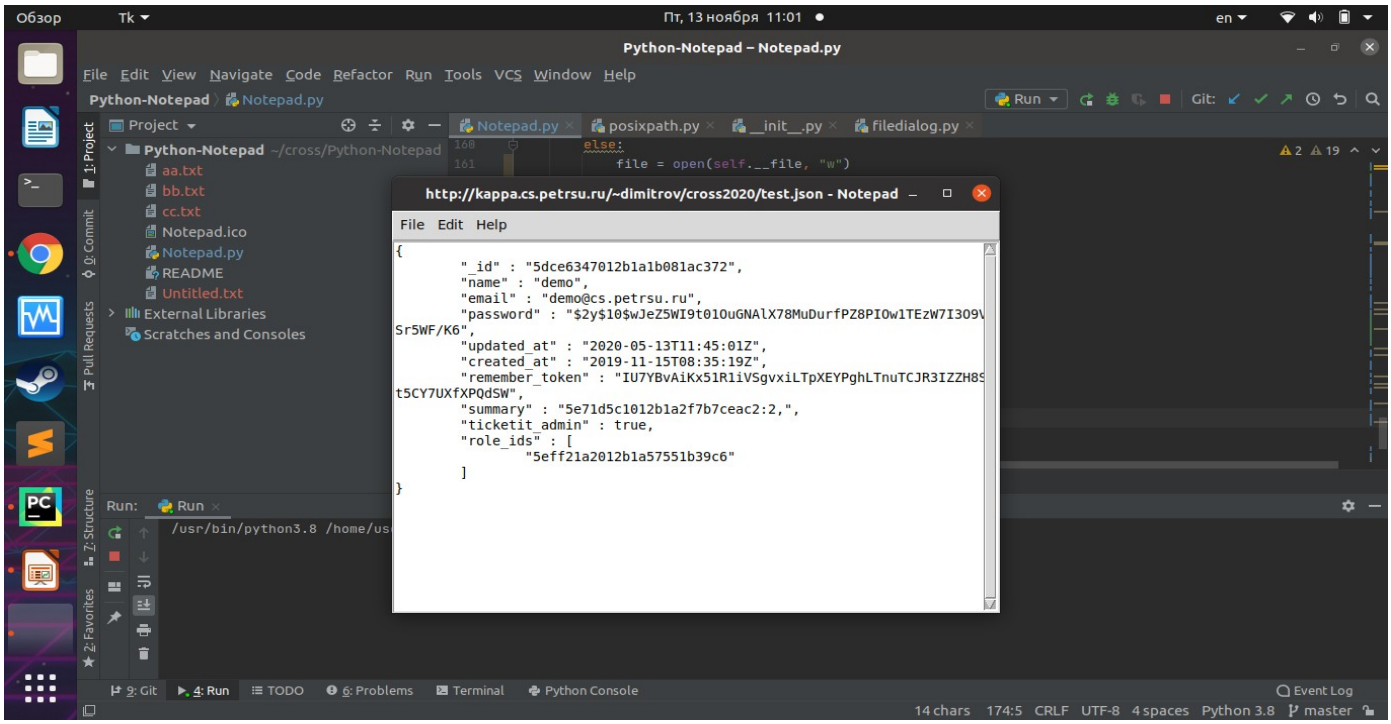
# Функциональные требования

1. Сохранение
2. Открытие по URL
3. Редактирование
4. Json валидация
5. Наличие графического интерфейса

# Функции приложения

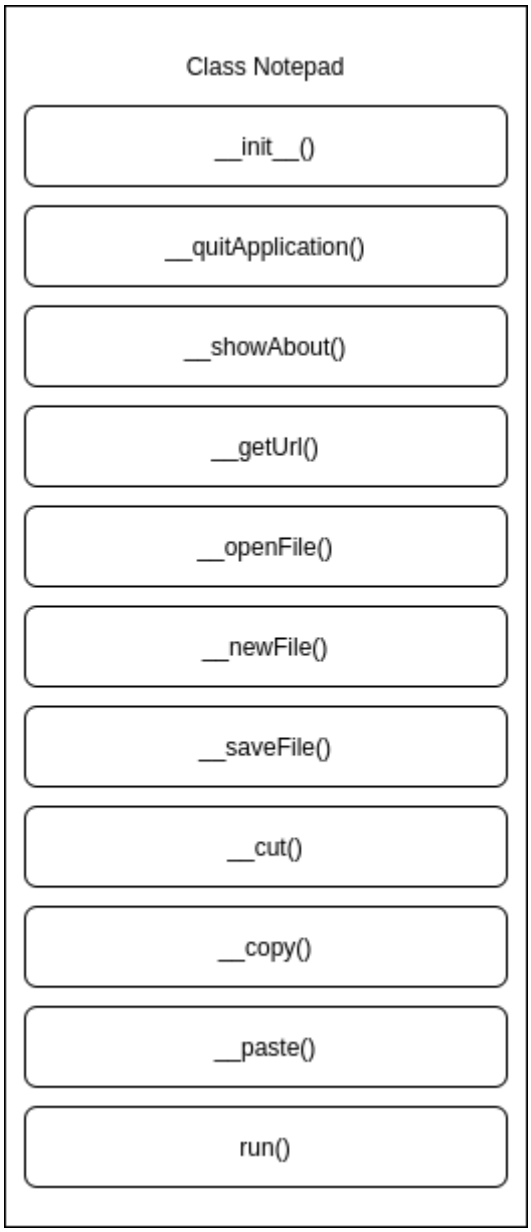
- def __init__(self, **kwargs) — инициализация
- def __quitApplication(self) — выход
- def __showAbout(self) — показать справку
- def __getUrl(self) — получить url
- def __openFile(self) — открыть файл
- def __newFile(self) — создать новый файл
- def __saveFile(self) — сохранить файл
- def __cut(self) — вырезать
- def __copy(self) — копировать
- def __paste(self) — вставить
- def run(self) — запустить

# Реализация


Обзор
Пт, 13 ноября 11:01
Python-Notepad – Notepad.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Python-Notepad › Notepad.py
Run
Git:
Project
Python-Notepad ~/cross/Python-Notepad
aa.txt
bb.txt
cc.txt
Notepad.ico
Notepad.py
README
Untitled.txt
External Libraries
Scratches and Consoles
Notepad.py
posixpath.py
__init__.py
filedialog.py
else:
file = open(self.__file, "w")
http://kappa.cs.petrsu.ru/~dimitrov/cross2020/test.json - Notepad
File Edit Help
{
"_id" : "5dce6347012b1a1b081ac372",
"name" : "demo",
"email" : "demo@cs.petrsu.ru",
"password" : "$2y$10$wJeZ5WI9t01OuGNAlX78MuDurfPZ8PIOw1TEzW7I3O9V
Sr5WF/K6",
"updated_at" : "2020-05-13T11:45:01Z",
"created_at" : "2019-11-15T08:35:19Z",
"remember_token" : "IU7YBvAiKx51R1iVSgvxiLTpXEYPghLTnuTCJR3IZZH8S
t5CY7UXfXPQdSW",
"summary" : "5e71d5c1012b1a2f7b7ceac2:2,",
"ticketit_admin" : true,
"role_ids" : [
"5eff21a2012b1a57551b39c6"
]
}
Run:
Run
/usr/bin/python3.8 /home/us
9: Git
4: Run
TODO
6: Problems
Terminal
Python Console
Event Log
14 chars 174:5 CRLF UTF-8 4 spaces Python 3.8 master

Стратегия тестирования

# Архитектура




Программа состоит из одного модуля, который отвечает за всю работу текстового редактора. Взаимодействие функций осуществляется с помощью следующих полей:

- root — главный trinket объект(окно)
- thisWidth — ширина окна
- thisHeight — высота окна
- thisTextArea — текстовая зона
- thisMenuBar — меню
- thisFileMenu — пункт меню «файл»
- thisEditMenu — пункт меню «правка»
- thisHelpMenu — пункт меню «помощь»
- thisScrollBar — элемент прокрутки
- file — файловый дескриптор

Все функции не получают параметров при вызове и не возвращают значений.

# Блочное тестирование

Блочные тесты проверяют работоспособность отдельных функций программы. Их поведение не зависит от результатов работы других тестируемых функций. Для их проверки в качестве аргументов при вызове функции передаются различные значения, после чего возвращаемый результат функции проверяется на наличие ошибок.

# Интеграционное тестирование

При интеграционном тестировании проверяется взаимодействие существующих модулей. Описанные выше, в блочном тестирование, функции, принимают ввод и

отдают
вывод другим. Таким образом проверяется отсутствие ошибок во взаимодействие их между друг другом.

## Аттестационное тестирование

Тестирование работы программы в целом. Производится путем запуска тестировщиком скомпилированной программы и созданием необходимых конфигурационных файлов. Таким образом проверяется работоспособность программы в виде приближенном к реальным условиям эксплуатации.

## Нагрузочное тестирование

Нагрузочное тестирование проверяет работу приложения в экстремальных условиях: при большом количестве операций копирования, при копировании больших файлов.

## Критерии прохождения тестирования

Тест считается пройденным если полученный и ожидаемый результат совпадают. Тестирование считается пройденным если во время его прохождения не выявлено критических ошибок и процент пройденных тестов не меньше 80%.

## Критерии остановки тестирования

Тестирование должно быть остановлено если количество не пройденных тестов больше 30% от общего количества, а так же при обнаружении критических ошибок сильно влияющих на функциональность приложения.

## Критерии возобновления тестирования

Тестирование возобновляется при исправлении ошибок на предыдущем этапе тестирования.

Тестирование

# Блочное тестирование

| | |
|---|---|
| № | Б1 |
| Цель теста: | Проверка работоспособности функции «__newFile ». |
| Тип теста: | Общий. |
| Объект теста: | Функция __newFile. |
| Входные параметры: | - |
| Ожидаемый результат: | Заголовок окна меняется на заданный, файл устанавливается пустым |

| | |
|---|---|
| № | Б2 |
| Цель теста: | Проверка работоспособности функции «__getUrl». |
| Тип теста: | Общий. |
| Объект теста: | Функция __getUrl. |
| Входные параметры: | http://kappa.cs.petrsu.ru/~dimitrov/cross2020/test.json |
| Ожидаемый результат: | Поле url совпадает с введенной ссылкой |

# Интеграционное тестирование

| | |
|---|---|
| № | И1 |

| | |
|---|---|
| Цель теста: | Проверка работоспособности взаимодействий __getUrl и __openFile |
| Тип теста: | Позитивный. |
| Объект теста: | Функции getUrl и openFile |
| Входные данные: | Валидная url (http://kappa.cs.petrsu.ru/~dimitrov/cross2020/test.json) |
| Ожидаемый результат: | Содержимое файла выводится в окно редактирования |

| | |
|---|---|
| № | И2 |
| Цель теста: | Проверка работоспособности взаимодействий __saveFile и __openFile |
| Тип теста: | Позитивный. |
| Объект теста: | Функции saveUrl и openFile |
| Входные данные: | Валидная url, по которой хранится валидный json файл |
| Ожидаемый результат: | Файл сохраняется |

# Аттестационное тестирование

| | |
|---|---|
| № | A1 |
| Описание: | Проверяются функциональные требования: Сохранение, Открытие по URL, Редактирование, Json валидация, Наличие графического интерфейса |
| Тип теста: | Позитивный |
| Сценарий аттестационного тестирования: | 1 Запустить приложение<br>2 Открыть URL с JSON<br>3 Внести изменения, оставив содержимое валидным JSON файлом<br>4 Сохранить файл |
| Ожидаемый результат | Файл должен сохраниться |

| | |
|---|---|
| № | A2 |
| Описание: | Проверяются функциональные требования: Сохранение, Открытие по URL, Редактирование, Json валидация, Наличие графического интерфейса |
| Тип теста: | Негативный |
| Сценарий аттестационного тестирования: | 1. Запустить приложение<br>2. Открыть URL с JSON<br>3. Внести изменения, оставив содержимое невалидным JSON файлом<br>4. Сохранить файл |
| Ожидаемый результат | Файл не должен сохраниться |

| | |
|---|---|
| № | A3 |
| Описание: | Проверяются функциональные требования: Сохранение, Открытие по URL, Редактирование, Json валидация, Наличие графического интерфейса |

| | |
|---|---|
| Тип теста: | Негативный |
| Сценарий аттестационного тестирования: | 1. Запустить приложение<br>2. Открыть URL с невалидным JSON<br>3. Внести изменения, оставив содержимое невалидным JSON файлом<br>4. Сохранить файл |
| Ожидаемый результат | Файл не должен сохраниться |

| | |
|---|---|
| № | A4 |
| Описание: | Проверяются функциональные требования: Сохранение, Открытие по URL, Редактирование, Json валидация, Наличие графического интерфейса |
| Тип теста: | Позитивный |
| Сценарий аттестационного тестирования: | 1. Запустить приложение<br>2. Открыть URL с JSON<br>3. Открыть URL с JSON еще раз<br>4. Внести изменения, оставив содержимое валидным JSON файлом<br>5. Сохранить файл |
| Ожидаемый результат | Файл должен сохраниться |

| | |
|---|---|
| № | A5 |
| Описание: | Проверяются функциональные требования: Сохранение, Открытие по URL, Редактирование, Json валидация, Наличие графического интерфейса |
| Тип теста: | Позитивный |
| Сценарий аттестационного тестирования: | 1. Запустить приложение<br>2. Открыть URL с невалидным JSON<br>3. Внести изменения, оставив содержимое валидным JSON файлом |

| | |
|---|---|
| | 4. Сохранить файл |
| Ожидаемый результат | Файл должен сохраниться |

## Специальное тестирование

| | |
|---|---|
| № | C1 |
| Описание: | Проверяются способность к чтению больших файлов |
| Тип теста: | Позитивный |
| Описание специального тестирования: | Читается (Вызов openFile) валидный json файл, который имеет размер 95-105 Мегабайт |
| Ожидаемый результат | Функция должна успешно обновить поле textArea, отработав не более 5 минут |

Результаты тестирования

# Таблица результатов

Журнал блочного и интеграционного тестирования

Каждый тест запускался 3 раза

| Номера тестов | Тип теста | Кол-во тестов | Кол-во ошибок | Дата | Тестировщик |
|---|---|---|---|---|---|
| Б1, Б2 | Общий | 2 | 0 | 29.11.20 | Клименко В.В. |
| И1,И2 | Общий | 2 | 1 в тесте И1 (Отчет об ошибке №1) | 30.11.20 | Клименко В.В. |

# Журнал аттестационного тестирования.

| Номера тестов | Кол-во тестов | Кол-во ошибок | Дата | Тестировщик |
|---|---|---|---|---|
| A1, A2 | 2 | 1(Отчет об ошибке №2) | 30.11.20 | Клименко В.В. |

## Журнал найденных ошибок

## Отчет об ошибке №1

Тест: И1

Объект тестирования: функции __getUrl и __openFile

Условия: выбрана русская раскладка для ввода

Приоритет: Низкий (ошибка теста)

Алгоритм: ввести Url

Ожидаемый результат: Файл открылся.

результат: Файл не открылся.

Воспроизводимость: 100%

Теоретическая причина: из-за ошибки в фреймворке тестирования вводились клавиши, а не символы, которые соответствуют набору из url.

Дата проведения: 30.11.20

### Отчет об ошибке №2

Тест: A1

Условия: отсутствует соединение с интернетом

Приоритет: Низкий (аппаратная ошибка)

Алгоритм:

1. Запустить приложение
2. Открыть URL с JSON
3. Внести изменения, оставив содержимое невалидным JSON файлом
4. Сохранить файл

Ожидаемый результат: Файл открылся и сохранился.

результат: Файл не открылся и не сохранился.

Воспроизводимость: 100%

Теоретическая причина: из-за отсутствия невозможно получить json файл по url без подключения к интернету.

Дата проведения: 30.11.20

# Результат

Проведенное тестирование показало, что программа нуждается в небольших доработках относительно обработки ошибок. Выявленные ошибки, были получены из за неправильной первичной настройки в коде программы и не критичны при её использовании. 90 строки из 127 значимых покрыты, таким образом 71% строк кода покрывается тестами.