Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Факультет инновационных технологий

УТВЕРЖДЕНО:
Декан
С. В. Шидловский

Оценочные материалы по дисциплине

Тестирование программного обеспечения

по направлению подготовки / специальности

09.03.02 Информационные системы и технологии

Направленность (профиль) подготовки/ специализация:
Программное и аппаратное обеспечение беспилотных авиационных систем

Форма обучения
Очная

Квалификация
Инженер - программист
Инженер - разработчик

Год приема
2025

СОГЛАСОВАНО:
Руководитель ОП
С.В. Шидловский

Председатель УМК
О.В. Вусович

Томск – 2025

1. Компетенции и индикаторы их достижения, проверяемые данными оценочными материалами

Целью освоения дисциплины является формирование следующих компетенций:

БК-1 Способен применять общие и специализированные компьютерные программы при решении задач профессиональной деятельности.

ПК-1 Способен разрабатывать ПО для интеллектуального управления БАС.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

РОБК-1.1 Знает правила и принципы применения общих и специализированных компьютерных программ для решения задач профессиональной деятельности

РОБК-1.2 Умеет применять современные IT-технологии для сбора, анализа и представления информации; использовать в профессиональной деятельности общие и специализированные компьютерные программы

РОПК-1.1 Знает принципы разработки ПО для интеллектуального управления БАС

РОПК-1.7 Умеет осуществлять тестирование готового ПО

2. Оценочные материалы текущего контроля и критерии оценивания

Элементы текущего контроля:

– тесты;

– контрольная работа;

– домашнее задание.

Пример теста:

Тест (РОПК-1.7)

1. Дано: программа для расчета остаточного заряда аккумуляторной батареи, в зависимости от длительности полета БПЛА.

На вход программе подается планируемое время полета (*integer*) в секундах, программа должна рассчитать и вывести остаточный заряд аккумуляторной батареи (*integer*). Что из нижеперечисленного является негативным тестом:

а) На вход программе подается значение 1

б) На вход программе подается значение 9999999

в) На вход программе подается значение 50

г) На вход программе подается значение 600.5

2. Что из нижеперечисленного НЕ относится к классу эквивалентности:

а) тест пограничного значения

б) позитивные тесты

г) тесты по типу входных данных

д) тест значения из негативного диапазона

Ключи: 1 г), 2 а)

Критерии оценивания: тест считается пройденным, если обучающийся верно ответил, более чем на половину вопросов.

Пример контрольной работы:

Контрольная работа (РОБК-1.1, РОБК-1.2, РОПК-1.1., РОПК-1.7.) Контрольная работа состоит из 2 задач.

Задачи:

Задача 1

Программа БПЛА поддерживает очередь команд, которые БПЛА должен выполнить. Размер очереди - 5.

Как разработчик программы, какие требования к реализации вы бы уточнили? Составьте чек-лист проверок для тестирования этого функционала.

Задача 2

Дан следующий фрагмент программы, представляющий собой интерфейс запуска команд:

```
class Messages:
    no_cmd_msg = 'Command not found'
    fly_msg = "I'm flying!"
    ground_msg = "I'm grounding!"
    record_msg = "I'm recording!"


class CmdMaster:
    def fly(self):
        return Messages.fly_msg

    def ground(self):
        return Messages.ground_msg

    def record(self):
        return Messages.record_msg

    def no_cmd(self):
        return Messages.no_cmd_msg


class CmdInterface:
    def __init__(self, cmd_source: CmdMaster):
        self.cmd_source = cmd_source

    def get_cmd(self, cmd):
            if cmd == 1:
                return self.cmd_source.fly()
            elif cmd == 2:
                return self.cmd_source.ground()
            elif cmd == 3:
                return self.cmd_source.record()
            else:
                return self.cmd_source.no_cmd()
```

напишите по одному позитивному и негативному модульному тесту для метода *get_cmd*.

Пример ответов:

Задача 1:

Можно уточнить:

- требуемое поведение при превышении размера очереди,
- количество одновременно выполняемых команд,
- поддержка асинхронности

Чек-лист проверок:

1. Команда попадает в очередь
2. Первой выполняется первая команда в очереди
3. В очереди не более 5 команд

Задача 2:

Пример решения без использования модуля unittest:

```
class UnitTestCmdInterface:
    def __init__(self, test_unit: CmdInterface):
        self.test_unit = test_unit

    def test_positive_1(self):
        assert self.test_unit.get_cmd(cmd=1) == Messages.fly_msg

    def test_negative_1(self):
        assert self.test_unit.get_cmd(cmd='Fake') == Messages.no_cmd_msg
```

Критерии оценивания:

Результаты контрольной работы определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «отлично» выставляется, если задачи решены без ошибок.

Оценка «хорошо» выставляется, если задачи решены с незначительными замечаниями.

Оценка «удовлетворительно» выставляется, если задачи решены с значительными замечаниями.

Оценка «неудовлетворительно» выставляется, если не решена хотя бы одна задача.

3. Оценочные материалы итогового контроля (промежуточной аттестации) и критерии оценивания

Экзамен в четвертом семестре проводится в письменной форме по билетам. Экзаменационный билет состоит из двух частей.

Первая часть представлена одним вопросом, проверяющими РОПК-1.7. Ответ дается в развернутой форме.

Вторая часть представлена двумя задачами, проверяющими РОБК-1.1, РОБК-1.2 и РОПК-1.7. Ответы предполагают решение задач и краткую интерпретацию полученных результатов.

Продолжительность экзамена 1,5 часа.

Перечень теоретических вопросов:

1. Артефакты тестирования. Что это? Какие артефакты вам известны? Приведите примеры.
2. Жизненный цикл программного обеспечения. Тестирование в контексте жизненного цикла ПО.
3. Требования к тестированию на проекте. Оценка тестового покрытия.
4. Техники тестирования. Какие знаете? В чем разница между техниками “диаграмма состояний” и “таблица принятия решений”?
5. Преимущества и недостатки нисходящего и восходящего тестирования интеграций
6. Тестирование белого, серого и черного ящиков. Где применяются? Преимущества и недостатки?
7. Жизненный цикл дефекта (бага). Методика оценки и приоритизации дефектов (багов).
8. В чем различие между тест-планом и чек-листом? Преимущества и недостатки?
9. Тест-кейс. Критерии качества тест-кейса.
10. В чем суть инкрементного интеграционного тестирования? В чем его преимущества перед модульным? Недостатки?
11. Оценка эффективности внедрения автоматизации тестирования.

Примеры задач:

Задача 1.

Дано:

Эндпоинт:

https://eg.eg/api/contract?sort=startDateOfExecution,desc&sort=id,asc&size=20&page=0

Формат ответа:

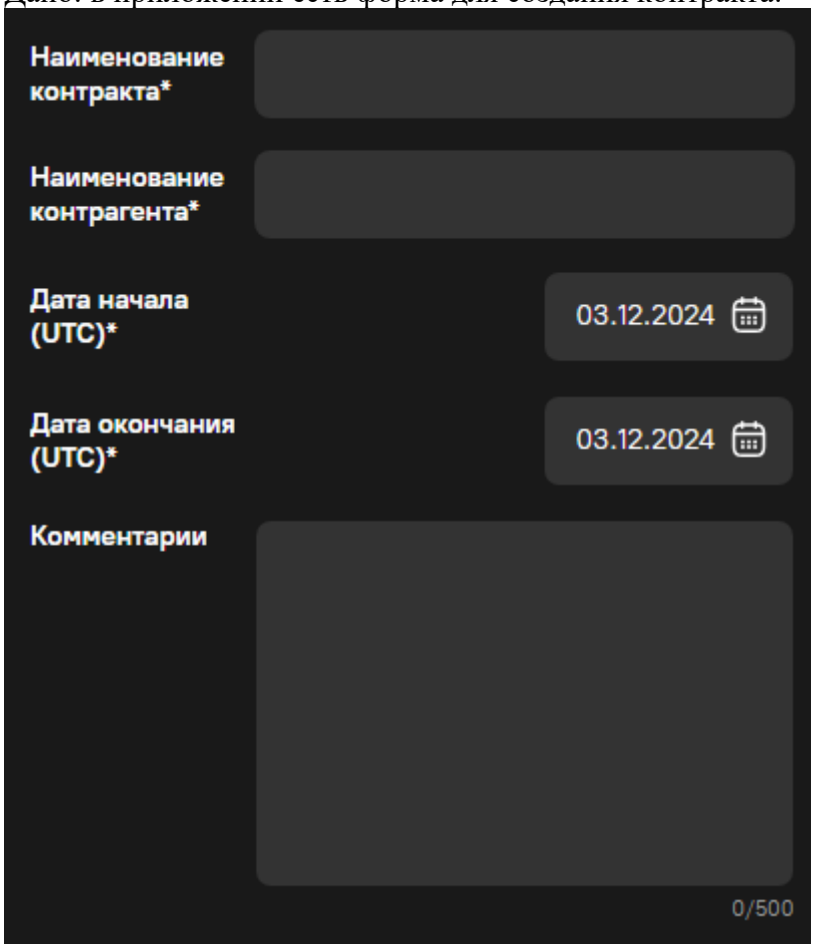

```
{
   "content": [
      {
         "id": 7,
         "contractName": "Контракт 1",
         "contrAgentName": "КОНТРАГЕНТ",
         "organizationId": 1,
         "startDateOfExecution": 1733184000000,
         "endDateOfExecution": 1734047999000,
         "expired": false,
         "comment": "",
         "status": "DRAFT"
      },
      {
         "id": 8,
         "contractName": "Контракт 2",
         "contrAgentName": "КОНТРАГЕНТ",
         "organizationId": 1,
         "startDateOfExecution": 1733184000000,
         "endDateOfExecution": 1733270399000,
         "expired": false,
         "comment": "",
         "status": "ACTIVE"
      }
         ],
   "totalPages": 1,
   "totalElements": 2,
   "last": true,
   "numberOfElements": 2,
   "size": 20,
   "number": 0,
   "sort": {
      "unsorted": false,
      "sorted": true,
      "empty": false
   },
   "first": true,
   "empty": false
}
```


Требуется: Написать авто-тесты, которые будут обращаться к приведенному эндпоинту и проверять следующие кейсы:

1. Значение в поле "numberOfElements" соответствует фактическому числу объектов в массиве "content".
2. Значение в поле "sorted" зависит от передаваемого query-параметра sort, т.е. если этот параметр задан, то "sorted": true, иначе - false.

Задача 2.

Дано: в приложении есть форма для создания контракта:



обязательные поля отмечены *.

Из требований известно, что:

- наименование контракта - это строка, состоящая из любых символов, кроме латиницы, максимальная длина 50.
- наименования контрагента - это строка, состоящая из любых символов, максимальная длина - 40.
- дата начала контракта может быть выставлена задним сроком.
- дата окончания контракта НЕ может быть выставлена задним сроком и НЕ может быть раньше даты начала контракта.

Требуется: Составить чек-лист проверок для приведенной формы.

Критерии оценивания:

Результаты экзамена определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «отлично» выставляется, если на вопрос дан верный развернутый ответ, а задачи решены без ошибок.

Оценка «хорошо» выставляется, если на вопрос в целом дан верный ответ, а задачи решены с незначительными замечаниями.

Оценка «удовлетворительно» выставляется, если в ответе на вопрос есть ошибки, или ответ неполный, или задачи решены с значительными замечаниями или ошибками.

Оценка «неудовлетворительно» выставляется, если не решена хотя бы одна задача или не дан ответ на вопрос, или приведенный ответ и решения задач не верны.

4. Оценочные материалы для проверки остаточных знаний (сформированности компетенций)

Тест

1. Какой из нижеприведенных терминов НЕ относится к тестовым артефактам:

а) Тест-кейс
б) Баг-репорт
в) Мок-тест
г) Чек-лист

2. Преимуществом монолитного тестирования является:

а) Независимость от драйверов и заглушек.
б) Легкость выявления источника возможных ошибок.
в) Уменьшение трудозатрат на регрессионное тестирование.
г) Упрощение делегирования ответственности в случае выявления дефекта.

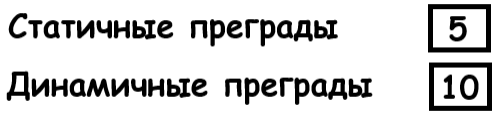
Ключи: 1 в), 2 а).

Задачи

Задача 1

Приведены следующие требования к функционалу:

Для эмулятора полета БПЛА нужно написать программу, которая будет создавать заданное количество преград для двух типов поведения. Пример интерфейса:



Максимальное общее количество преград - 20.

Составьте чек-лист проверки приведенного функционала.

Задача 2

Приведен фрагмент кода, представляющий собой некоторые вспомогательные объекты, а также тестируемый класс и написанные для него юнит-тесты:

```
class Messages:
    no_cmd_msg = 'Command not found'
    fly_msg = "I'm flying!"
    ground_msg = "I'm grounding!"
    record_msg = "I'm recording!"


class CmdMaster:
    def fly(self):
        return Messages.fly_msg
    def ground(self):
        return Messages.ground_msg
    def record(self):
        return Messages.record_msg
    def no_cmd(self):
        return Messages.no_cmd_msg


class CmdInterface:
    def __init__(self, cmd_source: CmdMaster):
        self.cmd_source = cmd_source
```

```
    def get_cmd(self, cmd):
            if cmd == 1:
                return self.cmd_source.fly()
            elif cmd == 2:
                return self.cmd_source.ground()
            elif cmd == 3:
                return self.cmd_source.record()
            else:
                return self.cmd_source.no_cmd()


class UnitTestCmdInterface:
    def __init__(self, test_unit: CmdInterface):
        self.test_unit = test_unit

    def test_positive_1(self):
        assert self.test_unit.get_cmd(cmd=1) == Messages.fly_msg

    def test_positive_2(self):
        assert self.test_unit.get_cmd(cmd=2) == Messages.fly_msg

    def test_negative_1(self):
        assert self.test_unit.get_cmd(cmd='Fake') == Messages.no_cmd_msg

    def test_negative_2(self):
        assert self.test_unit.get_cmd(cmd=None) == Messages.no_cmd_msg
```

Оцените уровень покрытия функции тестами и дайте оценку приведенным тестам.

Ответы:
Задача 1.
Чек-лист проверки:

- В первое поле введено 20, во второе - 0.
- В первое поле введено 0, во второе - 20.
- В первое поле введено 19, во второе - 1
- В первое поле введено 1, во второе - 19.
- В первое поле введено 1, во второе - 0.
- В первое поле введено 0, во второе - 1.
- В оба поля введены 10.
- В оба поля введены 0.
- В оба поля введены 20.
- В первое поле введено 21, во второе - 0
- В первое поле введено 0, во второе - 21
- В первое поле введено 11, во второе - 10
- В первое поле введено 10, во второе - 11.
- В оба поля введены дробные числа
- В оба поля введены не числа

Задача 2.
Примерный ответ:

Приведенная в примере функция поддерживает три основных команды и одну выделенную под обработку ошибок. В приведенных тестах только два позитивных кейса, при этом второй кейс - test_positive_2 - записан неверно, т.к. выполняется проверка на соответствие первой команде, а не второй. Негативные кейсы также покрывают не все возможные случаи, например ввод логического типа, ввод числа с плавающей точкой и др.

Случай с числом с плавающей точкой требует отдельного рассмотрения, т.к. интерпретатор python считает 1 == 1.0.

Теоретические вопросы:

1. Проблемы внедрения автоматизированного тестирования на проекте.

Ответ должен содержать аргументированные пункты проблематики авто-тестов, т.е. отвечающий должен понимать процесс внедрения автоматизированного тестирования и связанные с ним риски.

2. Классификация тестов по степени важности тестируемых функций.

Ответ должен содержать определения дымового, критического и расширенного тестирований.

5. Информация о разработчиках

Гимазов Руслан Уралович, кандидат технических наук, доцент кафедры интеллектуальных технических систем Факультета инновационных технологий.