Институт приборостроения, автоматизации и информационных технологий

Кафедра информационных систем и цифровых технологий

Дисциплина «Качество и тестирование программного обеспечения»

Отчет к лабораторной работе № 6

«Разработка собственного тестового драйвера»

Выполнил:

Василения Иван Валерьевич

Принял:

Олькина Елена Викторовна

Орёл, 2025г

Листинг программы:

import importlib.util

import json

import os

from pathlib import Path

class TestDriver:

def \_\_init\_\_(self, module\_path, test\_file):

self.module\_path = module\_path

self.test\_file = test\_file

def load\_module(self, module\_name):

spec = importlib.util.spec\_from\_file\_location(module\_name, self.module\_path / f"{module\_name}.py")

module = importlib.util.module\_from\_spec(spec)

spec.loader.exec\_module(module)

return module

def run\_tests(self):

with open(self.test\_file, 'r') as file:

tests = json.load(file)

results = []

for i, test in enumerate(tests, start=1):

try:

module = self.load\_module(test['module'])

func = getattr(module, test['name'], None)

if not callable(func):

raise AttributeError(f"Function {test['name']} is not found or not callable.")

result = func(\*test['params'])

expected\_result = test['expected\_result']

status = 'success' if result == expected\_result else 'faile'

message = '' if status == 'success' else f"Expected {expected\_result}, got {result}"

results.append({

'test\_number': i,

'received\_result': result,

'expected\_result': expected\_result,

'status': status,

'message': message

})

except Exception as e:

results.append({

'test\_number': i,

'received\_result': '',

'expected\_result': test.get('expected\_result', ''),

'status': 'faile',

'message': str(e)

})

return results

def generate\_report(self, results):

if os.name == 'nt': # Windows

os.system('color') # Включаем поддержку ANSI цветов в Windows 10+

RED = '\033[31m'

GREEN = '\033[32m'

RESET = '\033[0m'

for result in results:

report = "\n"

report += f"\nTest #{result['test\_number']}: {f'{GREEN}Success{RESET}' if result['status'] == 'success' else f'{RED}Failed{RESET}'}"

report += f"\n\tReceived Result: {result['received\_result']}"

report += f"\n\tExpected Result: {result['expected\_result']}"

report += f"\n\tMessage: {result['message']}" if result['message'] else ""

print(report)

if \_\_name\_\_ == "\_\_main\_\_":

module\_path = Path("test\_modules")

test\_file = "tests.json"

driver = TestDriver(module\_path, test\_file)

results = driver.run\_tests()

driver.generate\_report(results)

Листинг тестов:

[

{

"name": "insertion\_sort",

"params": [[]],

"expected\_result": [],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[1]],

"expected\_result": [1],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[3, 1, 2]],

"expected\_result": [1, 2, 3],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]],

"expected\_result": [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[-5, -10, 0, 5, 10]],

"expected\_result": [-10, -5, 0, 5, 10],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[1, 1, 1, 1, 1]],

"expected\_result": [1, 1, 1, 1, 1],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[1, 2, 3, 4, 5]],

"expected\_result": [1, 2, 3, 4, 5],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[5, 4, 3, 2, 1]],

"expected\_result": [1, 2, 3, 4, 5],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[10, 3, 17, 99, 44, 54, 67, 82, 1, 33, 55, 96]],

"expected\_result": [1, 3, 10, 17, 33, 44, 54, 55, 67, 82, 96, 99],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[-1, -2, -3, -4, -5]],

"expected\_result": [-5, -4, -3, -2, -1],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[100, 200, 300, 400, 500]],

"expected\_result": [100, 200, 300, 400, 500],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[500, 400, 300, 200, 100]],

"expected\_result": [100, 200, 300, 400, 500],

"module": "insertion\_sort"

},

{

"name": "insertion\_sort",

"params": [[3, 14, 1, 5, 9, 2, 6, 3]],

"expected\_result": [1, 2, 3, 3, 5, 6, 9, 14],

"module": "insertion\_sort"

}

]