



**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ**  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**САМАРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
Институт автоматизации и информационных технологий  
Кафедра «Информатика и вычислительная техника»

**ОТЧЕТ**

о выполнении лабораторной работы №2  
«Каркас мультиагентной системы»

по дисциплине «Архитектура программного обеспечения»

|               |               |           |        |                     |
|---------------|---------------|-----------|--------|---------------------|
| Преподаватель | Скобелев П.О. |           |        |                     |
|               | (должность)   | (подпись) | (дата) | (инициалы, фамилия) |
| Студент       | Абалымов А.А. |           |        |                     |
|               | 1-ИАИТ-114М   | (подпись) | (дата) | (инициалы, фамилия) |
|               | (группа)      |           |        |                     |
| Студент       | Желябин К.А.  |           |        |                     |
|               | 1-ИАИТ-114М   | (подпись) | (дата) | (инициалы, фамилия) |
|               | (группа)      |           |        |                     |

Самара 2025

## Самостоятельная работа:

В качестве самостоятельной работы необходимо расширить функции калькулятора, добавив туда возможности по выполнению других арифметических операций.

## Результаты работы:

Для расширения работы калькулятора (также как и в первой лабораторной работе) был добавлен актер – OperationAgent, который хранит математическую операцию (произведение, сумма чисел, среднее значение, очистка списка, вывод списка) и отправляет сообщение актору калькулятор для выполнения математической операции. Логiku агента математических операций реализовали в файле operation\_agent.py. Также был доработан агент-калькулятор.

## Код актора OperationAgent:

```
from thespian.actors import ActorAddress

from agent_base import AgentBase
from message import Message, MessageType

class OperationAgent(AgentBase):
    def __init__(self):
        super().__init__()
        # В это поле будем сохранять полученное значение.
        self.operation = ''
        self.subscribe(MessageType.INITIALIZATION,
self.handle_initialization)

    def handle_initialization(self, message: Message, sender: ActorAddress):
        print(f'Актер операции с адресом {self.myAddress} получил {message}
от {sender}')
        # Ожидаем, что в сообщении будет содержаться словарь со значением
математической операции и адресом актора-калькулятора.
        self.operation = message.msg_body.get('operation')
        calculator_address = message.msg_body.get('calculator_address')
        # Отправляем калькулятору свое значение.
        new_message = Message(MessageType.HELLO_WORLD, self.operation)
        self.send(calculator_address, new_message)
```

## Код доработанного агент-калькулятора:

```
from thespian.actors import ActorAddress

from agent_base import AgentBase
from message import Message

class CalculatorAgent(AgentBase):
    def __init__(self):
        super().__init__()
        # Все полученные числа будем сохранять в списке.
        self.values = []

    def handle_hello_world(self, message: Message, sender: ActorAddress):
        new_value = message.msg_body

        print(f'Калькулятор {self.myAddress} получил: {message}')

        if isinstance(new_value, (int, float)):
            self.values.append(new_value)
            print(f'Добавлено число: {new_value}. Всего чисел: {len(self.values)}')

        elif isinstance(new_value, str):
            if new_value == 'sum':
                result = sum(self.values)
                print(f'Сумма: {result}')
            elif new_value == 'multiply':
                result = 1
                for n in self.values: result *= n
                print(f'Произведение: {result}')
            elif new_value == 'avg':
                avg = sum(self.values)/len(self.values) if self.values else 0
                print(f'Среднее значение: {avg}')
            elif new_value == 'clear':
                self.values = []
                print('Список очищен')
            elif new_value == 'get values':
                print(f'Список чисел: {self.values}')
            else:
                print(f'Неизвестная операция: {new_value}')
```

## Проверка работы программы:

```
C:\Users\zhelo\AppData\Local\Programs\Python\Python39\python.exe
Добавлено число: 3. Всего чисел: 2
Актор числа с адресом ActorAddr-/A~d получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'init_value': 5, 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body=5)
Добавлено число: 5. Всего чисел: 3
Актор операции с адресом ActorAddr-/A~e получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'sum', 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='sum')
Сумма: 9
Актор операции с адресом ActorAddr-/A~f получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'avg', 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='avg')
Среднее значение: 3.0
Актор операции с адресом ActorAddr-/A~g получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'multiply', 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='multiply')
Произведение: 15
Актор операции с адресом ActorAddr-/A~h получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'some_operation', 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='some_operation')
Неизвестная операция: some_operation
Актор операции с адресом ActorAddr-/A~i получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'get values', 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='get values')
Список чисел: [1, 3, 5]
Актор операции с адресом ActorAddr-/A~j получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'clear', 'calculator_address': <thespian.actors.ActorAddress object at 0x00000168CD424910>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='clear')
Список очищен
Press any key to continue . . .
```

```
C:\Users\zhelo\AppData\Local\Programs\Python\Python39\python.exe
Добавлено число: 3. Всего чисел: 2
Актор числа с адресом ActorAddr-/A~d получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'init_value': 5, 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body=5)
Добавлено число: 5. Всего чисел: 3
Актор операции с адресом ActorAddr-/A~e получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'sum', 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='sum')
Сумма: 9
Актор операции с адресом ActorAddr-/A~f получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'avg', 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='avg')
Среднее значение: 3.0
Актор операции с адресом ActorAddr-/A~g получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'multiply', 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='multiply')
Произведение: 15
Актор операции с адресом ActorAddr-/A~h получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'some_operation', 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='some_operation')
Неизвестная операция: some_operation
Актор операции с адресом ActorAddr-/A~i получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'get values', 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='get values')
Список чисел: [1, 3, 5]
Актор операции с адресом ActorAddr-/A~j получил Message(msg_type=<MessageType.INITIALIZATION: 'Инициализация', msg_body={'operation': 'clear', 'calculator_address': <thespian.actors.ActorAddress object at 0x0000019C36262940>}) от ActorAddr-System:ExternalRequester
Калькулятор ActorAddr-/A~a получил: Message(msg_type=<MessageType.HELLO_WORLD: 'Приветствие', msg_body='clear')
Список очищен
Press any key to continue . . .
```

## Код проверки работы акторов:

```
if __name__ == '__main__':
    # Создаем систему акторов, внутри которой они будут жить
    actorSystem = ActorSystem()
    # Создаем актор-калькулятор, сохраняем его адрес.
    calculator_address = actorSystem.createActor(CalculatorAgent)
    # Создаем актор-число
    number_agent_1 = actorSystem.createActor(NumberAgent)
    # Формируем сообщение со значением числа и адресом калькулятора.
    init_message_1_data = {'init_value': 1, 'calculator_address':
calculator_address}
    # Отправляем актору числа сообщение, после которого он должен отправить
другое сообщение калькулятору.
    init_message_1 = Message(MessageType.INITIALIZATION, init_message_1_data)
    actorSystem.tell(number_agent_1, init_message_1)

    number_agent_2 = actorSystem.createActor(NumberAgent)
    init_message_2_data = {'init_value': 3, 'calculator_address':
calculator_address}
    init_message_2 = Message(MessageType.INITIALIZATION, init_message_2_data)
    actorSystem.tell(number_agent_2, init_message_2)

    number_agent_3 = actorSystem.createActor(NumberAgent)
    init_message_3_data = {'init_value': 5, 'calculator_address':
calculator_address}
    init_message_3 = Message(MessageType.INITIALIZATION, init_message_3_data)
    actorSystem.tell(number_agent_3, init_message_3)

    #создание агентов математических операций

    operation_agent_1 = actorSystem.createActor(OperationAgent)
    init_message_1_data = {'operation': 'sum', 'calculator_address':
calculator_address}
    operation_message_1 = Message(MessageType.INITIALIZATION,
init_message_1_data)
    actorSystem.tell(operation_agent_1, operation_message_1)

    operation_agent_2 = actorSystem.createActor(OperationAgent)
    init_message_2_data = {'operation': 'avg', 'calculator_address':
calculator_address}
    operation_message_2 = Message(MessageType.INITIALIZATION,
init_message_2_data)
    actorSystem.tell(operation_agent_2, operation_message_2)

    operation_agent_3 = actorSystem.createActor(OperationAgent)
    init_message_3_data = {'operation': 'multiply', 'calculator_address':
calculator_address}
    operation_message_3 = Message(MessageType.INITIALIZATION,
init_message_3_data)
    actorSystem.tell(operation_agent_3, operation_message_3)

    operation_agent_4 = actorSystem.createActor(OperationAgent)
    init_message_4_data = {'operation': 'some_operation',
'calculator_address': calculator_address}
```

```

        operation_message_4 = Message(MessageType.INITIALIZATION,
init_message_4_data)
        actorSystem.tell(operation_agent_4, operation_message_4)

        operation_agent_5 = actorSystem.createActor(OperationAgent)
        init_message_5_data = {'operation': 'get values', 'calculator_address':
calculator_address}
        operation_message_5 = Message(MessageType.INITIALIZATION,
init_message_5_data)
        actorSystem.tell(operation_agent_5, operation_message_5)

        operation_agent_6 = actorSystem.createActor(OperationAgent)
        init_message_6_data = {'operation': 'clear', 'calculator_address':
calculator_address}
        operation_message_6 = Message(MessageType.INITIALIZATION,
init_message_6_data)
        actorSystem.tell(operation_agent_6, operation_message_6)

```