|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
| Институт кибербезопасности и цифровых технологий |
| Кафедра КБ-2 «Прикладные информационные технологии» |

**ПРАКТИЧЕСКАЯ РАБОТА №1 по дисциплине**

**«Технологии и методы программирования»**

***(наименование дисциплины)***

**Тема практической работы   
«Модельно-ориентированный подход к проектированию»**

Студент группы: БИСО-01-20 Давыдова Д.А.

*(шифр учебной группы) (Фамилия И.О)*

Преподаватель: Лесько С.А.

*(Фамилия И.О)*

Москва, 2023

**Оглавление**

[**Порядок выполнения практической работы** 3](#_Toc133748227)

[Постановка задачи 3](#_Toc133748228)

[Задание 3](#_Toc133748229)

[**Диаграмма вариантов использования информационной системы** 4](#_Toc133748230)

[**Диаграмма классов** 5](#_Toc133748231)

[**Контрольные вопросы** 7](#_Toc133748232)

# **Порядок выполнения практической работы**

## Постановка задачи

1. При помощи программы PlantUML либо любого редактора построить UML-диаграмму вариантов использования, диаграмму классов проектируемой информационной системы в соответствии с вариантом задания, а также диаграмму последовательности для наиболее часто используемых прецедентов. При построении диаграммы классов нужно добиться достаточной детализации информационной системы. Убедитесь в том, что использовали отношения dependency, aggregation/c¬omposition, generalization, описали размещение классов по пакетам проекта.
2. Подготовить отчет с включением диаграмм. Загрузить в GIT. Прикрепить ссылку

## Задание

Вариант 7

Система учёта рабочего времени

Функциональные требования руководителя:

* Выдача задания
* Отслеживание хода выполнения задания

Функциональные требования исполнителя:

* Учёт рабочего времени
* Выполнение задания

# **Диаграмма вариантов использования информационной системы**

@startuml "Практическая работа 1"

left to right direction

title Система учета рабочего времени

actor Руководитель

actor Исполнитель

rectangle Система {

Руководитель -- (Выдавать задание)

(Выдавать задание) ..>(Выполнять задание):<<include>>

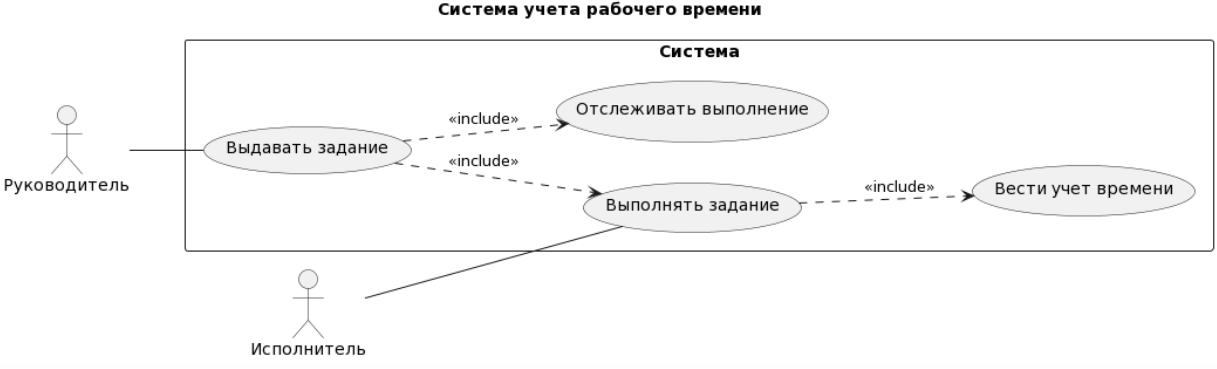
(Выдавать задание) ..> (Отслеживать выполнение):<<include>>

Исполнитель -- (Выполнять задание)

(Выполнять задание) ..> (Вести учет времени):<<include>>

}

@enduml



Описание

Руководитель выдаёт задания и отслеживает ход их выполнения в системе. Исполнитель ведёт учёт рабочего времени, затраченного на выполнение задания.

# **Диаграмма классов**

@startuml

class Руководитель{

+Паспортные данные

+ФИО

+Должность

Выдача задания()

Проверка Статуса()

}

class Задание{

+Номер задания

+Кто выдал задание

+Исполнитель задания

+Время исполнения задания

}

class Исполнитель{

+Паспортные данные

+ФИО

+Должность

Выполнение задания()

Изменение статуса()

Проверка времени()

}

class СтатусЗадания{

+Номер задания

+Статус задания

}

class ВремяЗадания{

+Номер задания

+Время

}

Руководитель --> Задание:Выдает

Исполнитель --> Задание:Выполняет

Исполнитель --> СтатусЗадания:Изменяет статус

Исполнитель -- ВремяЗадания:Проверять

Руководитель -- СтатусЗадания:Проверяет статус

Задание..>СтатусЗадания

Задание..>ВремяЗадания

@enduml

****

**Контрольные вопросы**

1. Что такое UML? Какие вы знаете основные диаграммы UML?

UML - унифицированный язык моделирования. Основные диаграммы:

•диаграмма прецедентов;

•диаграмма классов;

•диаграмма объектов;

•диаграмма последовательностей;

•диаграмма взаимодействия;

•диаграмма состояний;

•диаграмма активности;

•диаграмма развертывания.

1. Какие элементы входят в состав диаграммы классов?

Основными элементами являются классы и связи между ними. Классы характеризуются при помощи атрибутов и операций. Атрибуты описывают свойства объектов класса. Большинство объектов в классе получают свою индивидуальность из-за различий в их атрибутах и взаимосвязи с другими объектами, однако возможны объекты с идентичными значениями атрибутов и взаимосвязей, т.е. индивидуальность объектов определяется самим фактом их существования, а не различиями в их свойствах. Имя атрибута должно быть уникально в пределах класса. За именем атрибута может следовать его тип и значение по умолчанию. Операция есть функция или преобразование. Операция может иметь параметры и возвращать значения.