

# Список проектов 2020

## 1. Задачи на одно- и двумерную машину Тьюринга

Исходная идея была воплощена в мобильном приложении:

<https://github.com/OSLL/adfmp20-turing>

Задача: создать веб-приложение, которое будет интерпретатором одно- и двумерной машины Тьюринга, а также движком для решения простейших задач на ней. Функциональность:

- создание, редактирование, отображение, импорт и экспорт программ для Машин,
- визуальная симуляция работы Машин,
- выполнение программы по шагам,
- примитивные инструменты отладки,
- общие интерфейсы для создания задач,
- задачи-примеры из <https://github.com/OSLL/adfmp20-turing> и курса Информатика,
- сохранение результатов решения задач в БД,
- поддержка пользователей,
- поддержка протокола LTI\*.

Технологии: Python, Flask, MongoDB.

**deliverables:** веб-приложение, позволяющее решать задачи на машинах Тьюринга, работающее на Ubuntu 18.04.

## 2. Плагин для Moodle - отображение общего прогресса по курсу

Задача: создать плагин для Moodle, который будет отображать прогресс пользователя по текущему курсу. Функциональность:

- отображение прогресс-бара,
- отображение текущего процента прохождения,
- отображения цветом пройденных модулей,
- отображение «расстояния до сертификатов»,
- публикация в каталоге плагинов\*.

Технологии: PHP, MySQL, Moodle.

**deliverables:** работоспособный плагин, который можно установить в Moodle версии 3.8.2.

## 3. Плагин для Moodle - лайки и дизлайки

Задача: создать плагин для Moodle, который позволит ставить активностям лайки и дизлайки. Функциональность:

- отображение счетчиков,
- отображение кнопок,

- хранение данных в БД,
- не возможность поставить лайк / дизлайк дважды и поставить обе оценки сразу,
- отключение кнопок в настройках курса,
- отключение кнопок в настройках активностей,
- публикация в каталоге плагинов\*.

Технологии: PHP, MySQL, Moodle.

**deliverables:** работоспособный плагин, который можно установить в Moodle версии 3.8.2.

## [R&D] 4. Автоматическая проверка дипломных презентаций

Исследовательская составляющая: применение простых методов Natural Language Processing для анализа связности документа (насколько в нем раскрыты цели и задачи).

Задумка: дипломники нашей кафедры перед защитой проходят проверку презентаций по набору правил [http://se.moevm.info/doku.php/diplomants:start:slides\\_checklist\\_etu](http://se.moevm.info/doku.php/diplomants:start:slides_checklist_etu). Большая часть из этого набора может быть проверена автоматически, что сэкономит массу времени как самим учащимся, так и преподавателям.

Задача: создать веб-приложение, которое позволит проверять презентации (форматы ppt/pptx/odp/odpx) по наборам критериев ([http://se.moevm.info/doku.php/diplomants:start:slides\\_checklist\\_etu](http://se.moevm.info/doku.php/diplomants:start:slides_checklist_etu)), а также иметь потенциал для расширения этого списка / кастомизации шкалы оценивания. Функциональность:

- форма отправки презентации на проверку
  - загрузка файла
  - индикатор загрузки и индикатор процесса проверки
- страница результатов проверки
  - список проведенных проверок, результат (пройдена / не пройдена), указание на проблемные слайды,
  - уникальный идентификатор страницы проведенной проверки,
- проверка соблюдения простых (→ их можно проверить с помощью поиска подстрок / подсчета объектов) критериев:
  - Количество основных слайдов
  - Все слайды, кроме титульного должны иметь номера в правом нижнем углу.
  - Заголовки слайдов занимают не более двух строк.
  - На одном из слайдов в начале презентации описывается актуальность работы.
  - Есть слайд «Цель и задачи» (наличие формулировок Цели и Задач)
  - Есть слайд «Апробация работы»
  - Есть слайд «Заключение» (сам факт наличия слайда)
- проверка «сложных критериев» (потребуется проделать стемминг всех слов в презентации чтобы выполнить поиск):
  - На каждую задачу (и достигнутый по ней результат) есть отдельный слайд или слайды, которые его демонстрируют (формулировки задач пересекаются с заголовками слайдов).
  - Есть слайд «Заключение», в котором
    - кратко приводятся основные результаты решения каждой задачи со слайда «Цель и задачи», (формулировки задач пересекаются с результатами в Заключении, про каждую задачу что-то написано)

- приведены направления дальнейшей разработки или улучшения программы.  
(есть в каком-то виде фраза про дальнейшие ...)
- сохранение результатов проверки в БД,
- поддержка пользователей-администраторов,
- поддержка протокола LTI\*,
- настройка шкалы проверки и сохранение шкал - выбор критериев для проверки \*,

Технологии: Python, Flask, MongoDB.

**deliverables:** веб-приложение, работающее на Ubuntu 18.04.

## [R&D] 5. Инструмент поиска повторов на аудиозаписях

Исследовательская составляющая: применение простых методов обработки аудио.

Задумка: для одной практической задачи необходимо работать с аудиозаписями докладов (например защиты курсовой), проведенных вживую. Однако, достоверно известно, что некоторые аудиозаписи сфабрикованы и содержат повторы .

Задача: создать приложение командной строки, которое будет принимать на вход аудиофайлы (mp3), со следующей функциональностью:

- фильтрация шумов (по аудиообразцу шума), оценка их уровня
- сохранение отфильтрованной версии входных файлов
- вычисление повторяющегося фрагмента в рамках одного mp3 файла,
- вычисление повторяющегося фрагмента в рамках нескольких mp3 файлов,
- Docker-контейнер для работы приложения

Технологии:

- Python
- (может и не пригодится) <https://github.com/unmade/audiomatch>
- <https://github.com/aubio/aubio>
- [https://github.com/dodiku/noise\\_reduction](https://github.com/dodiku/noise_reduction)

**deliverables:** приложение для командной строки, работающее на Ubuntu 18.04.

## [R&D] 6. Инструмент анализа голосов на аудиозаписи

Исследовательская составляющая: применение методов обработки аудио.

Задача: создать приложение командной строки, которое будет принимать на вход аудиофайлы (mp3), со следующей функциональностью:

- идентификация диктора по набору контрольных образцов,
- идентификация наличия нескольких дикторов на аудио,
- вычисление и вывод фрагментов аудиозаписи для каждого из дикторов в файлы
- распознавание речи для каждого диктора (с указанием меток времени) с выводом в файл
- Docker-контейнер для работы приложения

Технологии:

- Python
- <https://github.com/resemble-ai/Resemblyzer>
- <https://realpython.com/python-speech-recognition/>

**deliverables:** приложение для командной строки, работающее на Ubuntu 18.04.

## 7. Инструмент поиска плагиата в исходном коде

Задача: создать веб-приложение для поиска плагиата в исходном коде, со следующей функциональностью:

- загрузка базы исходных кодов для поиска плагиата
- загрузка проверяемого исходника по ссылке и через форму
- вывод входного исходника с пометками заимствований
- вывод процента оригинальности
- поддержка языков C, python, java, js для входных файлов
- обработка репозиторий в качестве входных данных
- поиск по материалам Github / Stackoverflow
- Docker-контейнер для работы приложения

Технологии:

- Python
- <https://github.com/TylerJaacks/ItsJustACoincidenceProfessor>
- <https://github.com/codeclassroom/PlagCheck>
- <https://github.com/abhilampard/Simple-Plagiarism-Checker>
- <https://github.com/JonathanReeve/text-matcher>

**deliverables:** веб-приложение, работающее на Ubuntu 18.04.

## 8. Инструмент для генерации датасетов для фотограмметрии

Что такое фотограмметрия: <https://habr.com/ru/post/319464/>

Чем вдохновляется тема [https://github.com/moevm/bsc\\_kovinev](https://github.com/moevm/bsc_kovinev)

Задумка: создать приложение, где можно будет настроить параметры виртуальной сцены (что показывать), освещения (какой светит свет), погоды (дождь, снег, без осадков), а также параметры создания набора снимков (траектория, тип камеры).

Задача: создать GUI-приложение для создания фотограмметрических датасетов, со следующей функциональностью:

- Отображение пользовательской модели сцены
- Настройка освещения
- Настройка погоды
- Настройка траектории и частоты съемок
- Экспорт снимков в файлы

- Docker-контейнер для работы приложения

Технологии:

- Unreal Engine

**deliverables:** GUI-приложение, работающее на Ubuntu 18.04.

## 9. Генератор заданий для к/р по темам «Интерполяция функций» и «Численное интегрирование»

Задача: реализовать генератор заданий для к/р по темам «Интерполяция функций» и «Численное интегрирование». Приложение должно по некоторому номеру генерировать заданное число вариантов по данным темам, а также ответы для преподавателя. Примеры заданий можно найти [здесь](#).

Функциональность:

- приложение должно быть кроссплатформенным с пользовательским интерфейсом (gui/web)
- параметризация длины исходных данных (например, чтобы таблица могла быть произвольного размера)
- параметризация компоновки билетов на странице
- использование списка ФИО вместо номеров вариантов
- приложение должно генерировать файлы формата TeX и/или PDF по выбору пользователя

Технологии: Python

*Примечание: обратите внимание на то, как сформированы значения в заданиях. В заданиях по интерполяции полиномы получаются с целочисленными значениями, а в заданиях по численному интегрированию суммы должны делиться без калькулятора. Это делается для того, чтобы студенты «в полевых условиях» смогли решить данные задачи за отведённое время.*

**deliverables:** приложение, работающее на Ubuntu 18.04.

## 10. Разработка системы удаленного проведения экзаменов (Proctoring)

Кафедра сейчас занимается созданием собственной системы для прокторинга - процедуры наблюдения и контроля за дистанционным экзаменом (от англ. «proctor» - человек, который следит за ходом экзамена в университете). Приложение уже находится в процессе разработки.

Необходимо выполнить следующие задачи:

- Создание шкалы с отметками о нарушениях.

Во время экзамена студенты иногда нарушают правила проведения, поэтому в системе есть возможность просмотра нарушений. Нужно сделать кликабельную временную шкалу для видео с экзамена. Там выводить нарушения. Пример показан на фото:



- Реализовать чат с преподавателем.
- Добавление в базу данных MongoDB сведений о студенте. Сделать фильтрацию по любым параметрам (даты/ФИО студентов/LMS/оценки).

Технологии: JavaScript, React, NodeJS, MongoDB, Docker.

## 11. Настройка взаимодействия системы прокторинга с Moodle

Кафедра сейчас занимается созданием собственной системы для прокторинга - процедуры наблюдения и контроля за дистанционным экзаменом (от англ. «proctor» - человек, который следит за ходом экзамена в университете). Прокторинг реализован с использованием микросервисной архитектуры. Необходимо настроить взаимодействие системы прокторинга с образовательной платформой Moodle. Для этого нужно разобраться, как работает плагин Event Trigger, подключить к Moodle и настроить взаимодействие с системой прокторинга.

Необходимо протестировать и внедрить плагин.

Технологии: Moodle, JavaScript.

В системе прокторинга используются: React, NodeJS, MongoDB, Python, WebRTC, Kurento Media Server, Docker. При тестировании придется запускать весь стек технологий.

## 12. Инструмент для генерации отчетов

На данный момент разработан инструмент, который позволяет генерировать отчет из markdown в отчет, соответствующий регламенту «ЛЭТИ». Требуется:

1. Добавить систему ошибок(когда возникает какая либо ошибка, пользователь должен узнать об этом)
2. Изменить парсер markdown текста для случаев вложенного списка(больше двух) и др.
3. Автоматизировать сборку docker-контейнера
4. Добавить недостающие поля, такие как: аннотация на английском и др.
5. Вставка diff'a в отчет для нужных коммитов, используя github api
6. Поменять работу с гит репозиториями, используя github api
7. Тестирование\*

Технологии: Python3, Flask, Docker, Js, Mongodb, html, css

## 13. Мобильное приложение для кафедральной платформы moodle

Требуется реализовать возможности:

1. кастомизации стилистику приложения
2. учиться (смотреть видео/проходить тесты) и видеть свой прогресс
3. общаться с преподавателем через чат
4. преподаватель курса может смотреть статистику по курсу/студенту

Технологии: Kotlin

## 14. Плагин для создания сертификатов

Требуется реализовать плагин для платформы moodle, который позволяет создавать, настраивать, отображать сертификат как элемент курса, а также проверять сертификат на валидность.

В настройку сертификата входит:

- добавление элементов, таких как: Имя студента, фон, картинки (логотип платформы/курса), текст, название курса, ссылка на курс, ссылка на проверку сертификата, процент прохождения, дата, подписи преподавателей (в том числе в формате png), и т.д.
- настройка размера сертификата
- настройка положения элементов: нужно иметь возможность задать (или вычислить) координаты положения каждого элемента

Технологии: php, js, css, mysql

From:

<https://se.moevm.info/> - **МОЭВМ Вики** [se.moevm.info]

Permanent link:

[https://se.moevm.info/doku.php/courses:mse:projects\\_2020](https://se.moevm.info/doku.php/courses:mse:projects_2020)

Last update:

