

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«Дальневосточный федеральный университет»**

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ  
Департамент программной инженерии и искусственного интеллекта**

**РАЗРАБОТКА ТРАНСЛЯТОРА С С++ НА PYTHON**

КУРСОВАЯ РАБОТА

по дисциплине «Технологии коллективной промышленной разработки информационных систем» по образовательной программе подготовки бакалавров по направлению 09.03.04 «Программная инженерия»

Выполнили студенты гр. Б9119-09.03.04прогин  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Колпакова А. И.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Олейникова A. E.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Егорова К. С.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Мартын В.А

Руководитель:  
асисстент департамента ПИиИИ,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Иваненко Ю. С.

г. Владивосток  
2025

Оглавление

[1. Разработка плана проекта 3](#_Toc182842369)

[2. Разработка регламента проведения инспекции 4](#_Toc182842370)

[3. Разработка модели состояний задач 6](#_Toc182842371)

[4. Разработка презентации проекта 7](#_Toc182842372)

[5. Разработка требований к проекту 9](#_Toc182842373)

# **Разработка плана проекта**

Исполнители проекта:

* Team Leader – Колпакова Александра;
* Coder 1 – Колпакова Александра;
* Coder 2 – Олейникова Анна;
* Coder 3 – Егорова Ксения;
* Coder 4 – Мартын Владислав;
* Build Engineer – ;
* Technical Writer – .

Перечень задач и сроки их выполнения представлены на рисунке 1.

Рисунок 1 – Календарный план работы над проектом

# **Разработка регламента проведения инспекции**

* Критерии инспекции для различных типов рабочих продуктов

|  |  |
| --- | --- |
| **Рабочий продукт** | **Размер продукта или изменения в нем не менее** |
| Требования | 10% рабочего продукта |
| Документы по дизайну | 5% рабочего продукта |
| Не комментированный код | 20 NCLOC |
| Тесты | 25% рабочего продукта |

* Роли участников инспекции

*Обязанности автора* – оповестить председателя инспекции о готовности инспектируемого программного продукта. Провести обзорное собрание.

*Обязанности председател*я – убедиться в готовности продукта к формальной инспекции и определить её необходимость. Оповестить всех участников о необходимости проведения инспекции.

*Обязанности инспектора* – провести анализ работоспособности инспектируемого продукта.

* Участники инспекции

*Автор*: Егорова Ксения, Мартын Владислав

*Инспектор*: Олейникова Анна.

*Председатель*: Колпакова Александра.

* Этапы инспекции
  + Формальный этап
    - Оповестить команду и выбрать дату проведения
    - Провести инспекцию
    - Утвердить коллективное решение
    - Завершение инспекции
  + Неформальный этап
    - Оповестить команду
    - Провести инспекцию
* Порядок организации

Инспектируемый продукт выгружать в отдельную ветку на GitHub, информацию о командном сборе публиковать в telegram, видеосвязь по discord.

* Перечень статусов и степени важности замечаний
  + Очень важно
  + Важно
  + Выполнено
* Метрики, характеризующие эффективность инспекции

Inspection Fault Density (IFD) IFD = (Количество найденных ошибок / Размер рабочего продукта).

Стратегическая цель метрики – повысить качество разрабатываемого ПО. Изучаемый объект метрики – инспекция, измеряемый атрибут – плотность найденных в ходе инспекции ошибок. Единица измерения – ошибка / <страница, требование, LOC, тест>.

# **Разработка модели состояний задач**

* Перечень возможных состояний задач и их интерпретация состояний
  + TODO – задача в очереди на выполнение
  + IN ASSIGN – задача доведена до исполняющего
  + IN PROGRESS – в данный момент исполнитель занят решением задачи
  + CHECKOUT – исполнитель закончил задачу и нуждается в инспекции
  + DONE – задача полностью выполнена
* Правила создания новой задачи
  + Задачи (TODO, IN ASSIGN, DONE) назначает лидер команды
  + Задачи (IN PROGRESS, CHECKOUT) назначает исполнитель
* Правила перехода из состояния в состояние
  + TODO переходит в IN ASSIGN когда лидер доносит информацию до исполнителя, что ему назначена задача
  + IN ASSIGN переходит в IN PROGRESS когда исполнитель берется за выполнение задачи
  + IN PROGRESS переходит в CHECKOUT когда исполнитель закончил задачу и нуждается в проведении инспекции
  + CHECOUT переходит в DONE в случае когда была проведена инспекция с результатом положительно
  + CHECKOUT переходит в IN PROGRESS в случае когда была проведена инспекция с результатом отрицательно

# **Разработка презентации проекта**

Первый слайд презентации содержит состав команды (рисунок 2)



Рисунок 2 - Первый слайд

Второй слайд содержит краткое описание проекта (рисунок 3)

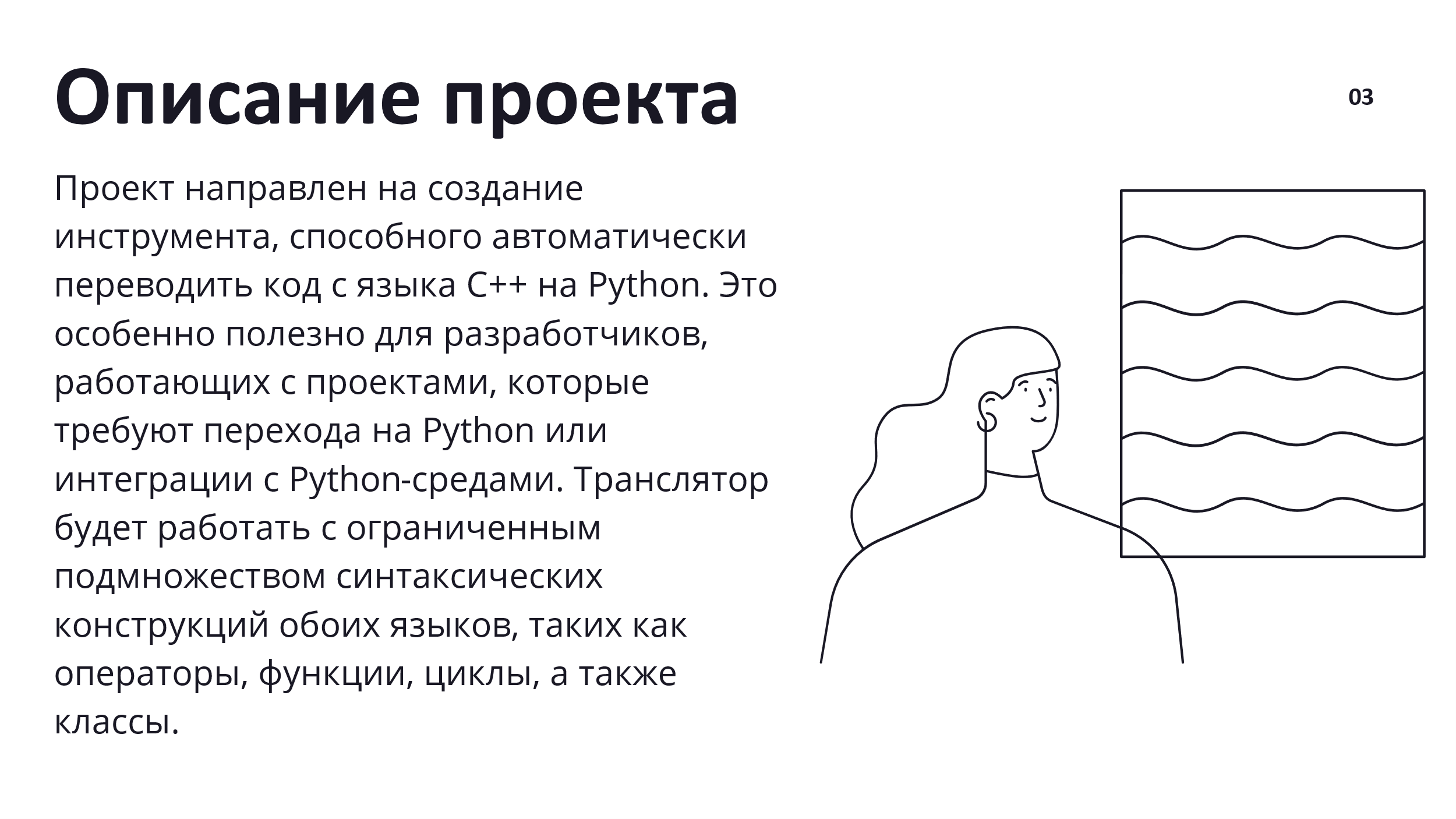


Рисунок 3 - второй слайд

На третьем слайде представлены преимущества проекта (рисунок 4)



Рисунок 4 - третий слайд

На четвертом слайде представлен прототип пользовательского интерфейса проекта (рисунок 5)

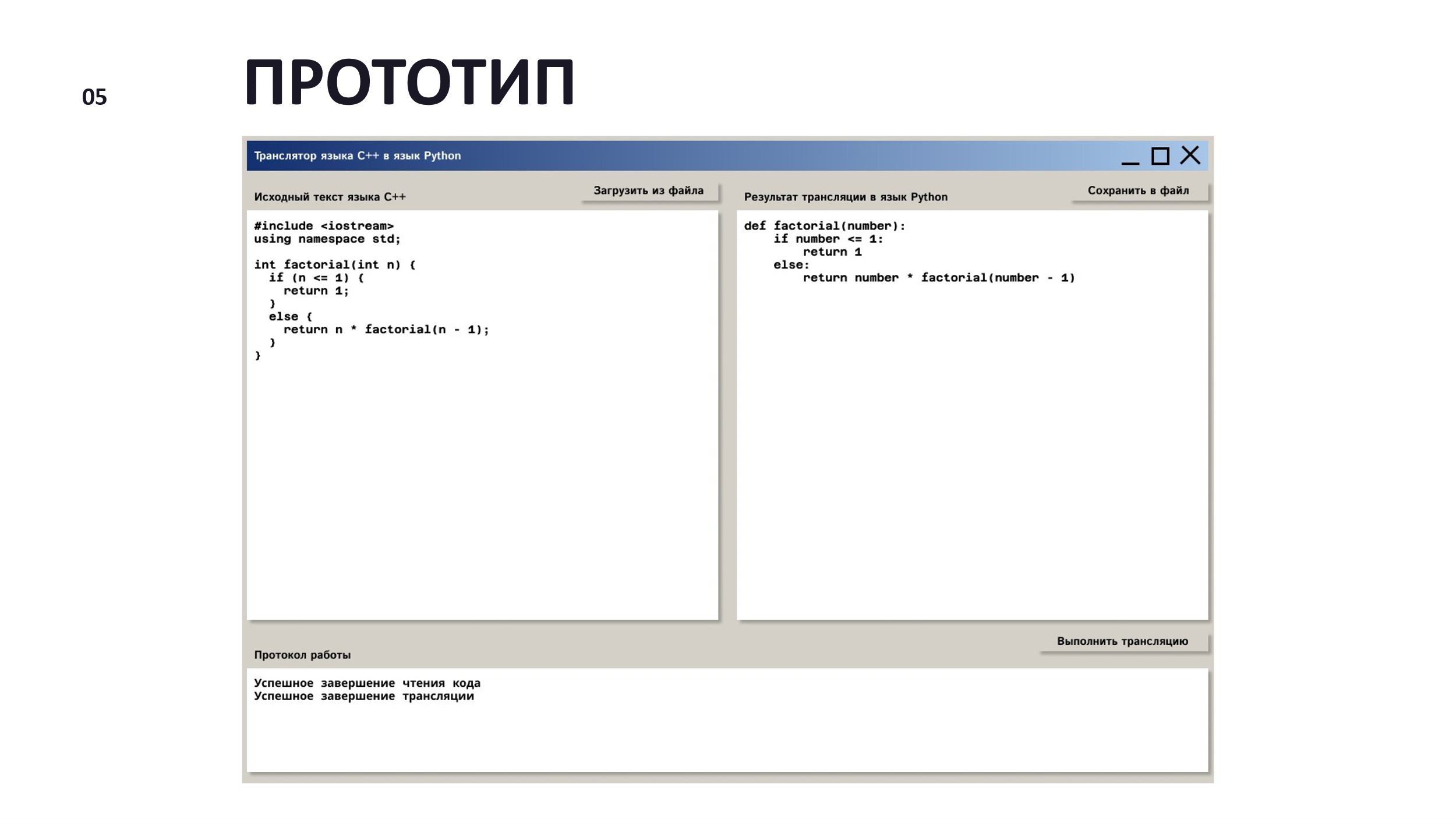


Рисунок 5 - четвертый слайд

На пятом слайде представлен основной функционал проекта (рисунок 6)

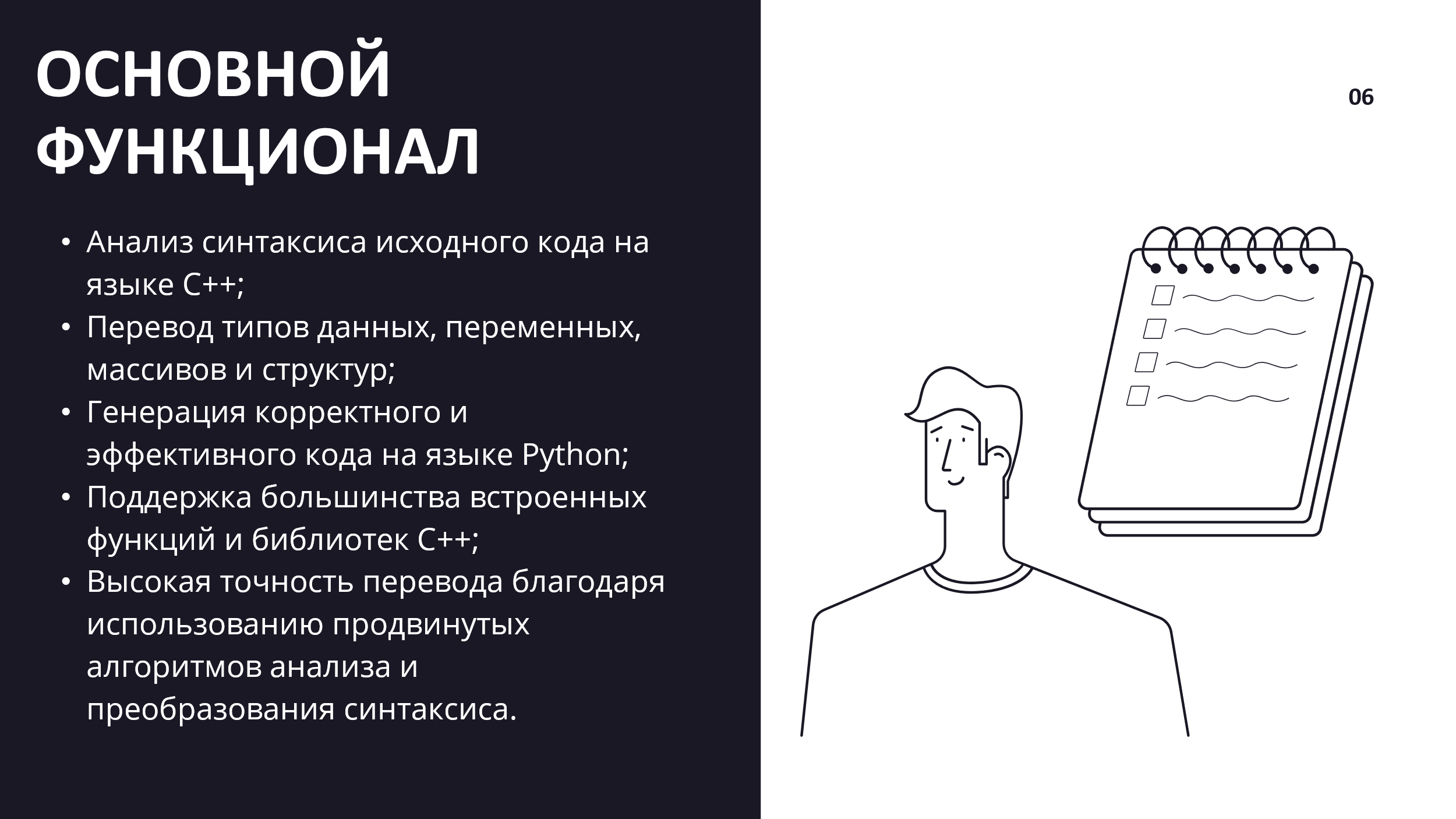


Рисунок 6 - пятый слайд

# **Разработка требований к проекту**

Программный продукт *Транслятор С++ -> Python* предназначен для трансляции (перевода) с языка программирования С++ на язык Python.

Программный продукт *Транслятор С++ -> Python* состоит из следующих подсистем:

1. Пользовательский интерфейс (UI)
2. Лексический анализатор (LA)
3. Синтаксический анализатор (SA)
4. Генератор кода (CG)

**Требования к подсистеме «Пользовательский интерфейс»**

Требование REQ\_UI\_001

Пользовательский интерфейс продукта должен содержать:

* Поле ввода исходного кода на языке C++
* Поле вывода транслированного кода на языке Python
* Кнопка трансляции языка
* Кнопка загрузки файла
* Кнопка выгрузки файла
* Окно протокола работы

Требование REQ\_UI\_002

При допущении пользователем синтаксической ошибки, при написании в поле ввода исходного кода и нажатии на кнопку трансляции, в отдельном окне должно выводиться сообщение об ошибке.

Требование REQ\_UI\_003

Кнопка трансляции языка должна иметь вид прямоугольника с подписью «Выполнить трансляцию» (См. рисунок 7).



Рисунок 7 – Кнопка трансляции

Требование REQ\_UI\_004

Поля ввода и вывода исходного и транслированного языка соответственно, должны располагаться на одном уровне, причем поле ввода должно находиться левее.

Требование REQ\_UI\_005

Кнопка выгрузки файла должна иметь вид прямоугольника с подписью «Загрузить из файла» (См. рисунок 8).



Рисунок 8 – Кнопка выгрузки

Требование REQ\_UI\_006

Кнопка загрузки в файл должна иметь вид прямоугольника с подписью «Сохранить в файл» (См. рисунок 9).



Рисунок 9 – Кнопка загрузки

Требование REQ\_UI\_007

Поле вывода протокола работы должно располагаться под полями ввода и вывода кода.

**Требования к подсистеме «Лексический анализатор»**

Требование REQ\_LA\_001

На вход подсистеме поступает исходный код на языке С++. На выход подсистемы поступает множество токенов, либо ошибка.

Требование REQ\_LA\_002

Каждый токен представляет собой четверку <тип, значение, номер строки, номер столбца>.

**Требования к подсистеме «Синтаксический анализатор»**

Требование REQ\_SA\_001

Синтаксический анализатор получает на вход массив токенов от лексического анализатора. На его основе строится абстрактное синтаксическое дерево разбора, если синтаксических ошибок нет, иначе – выдаётся ошибка.

**Требования к подсистеме «Генератор кода»**

Требование REQ\_CG\_001

Генератор кода получает на вход дерево разбора от синтаксического анализатора. На выходе получаем код программы на языке Python.

//Добавить описание от треттьего лица