Генератор условий задач

Команда:

9303 Гугунов Сергей

1381 Тарасов Константин

1381 Возмитель Влас

1381 Манучарова Ангелина

1381 Харитонов Никита

1384 Степаненко Денис

План на текущую итерацию

- Собрать и обработать требования заказчика
- Изучить теоретический материал
- Создать макет UI
- Создать сценарии использования приложения

Процесс реализации задач

Были проведены три встречи с заказчиком.
Были составлены протоколы встречи и, как результат, определены цели проекта, возможные типы задач, которые необходимо сгенерировать.

Цели и типы задач оформлены на wiki-странице и загружены в репозиторий.

Протоколы встреч

Выдержки из протоколов:

Протокол от 18.02

- Определена цель проекта
- Определена суть программы
- Определены input и ouput

Протокол от 22.02

- Даны комментарии о макете
- Даны комментарии о предложенных типах задач

Протокол от 26.02

- Даны комментарии по дополненным типам и исправленному макету

С полным протоколом каждой встречи можно ознакомиться в репозитории

Цели и типы задач

Цели проекта:

- Оптимизировать время работы преподавателя, это включает в себя создание задач быстрее и более точные
- Составить алгоритм, который помогает в создании n-количества типов задач
- Понять, как можно измерить мощность множества задач, которое может быть сформировано в рамках этого количества задач
- Составить классификацию типов задач, которые возможно генерировать в рамках проекта

Типы задач:

- Синтаксис
- Логика
- Алгоритмы
- Оптимизация
- Затирание кода
- Теор. Вопросы

Полное описание находится на wiki-странице проекта

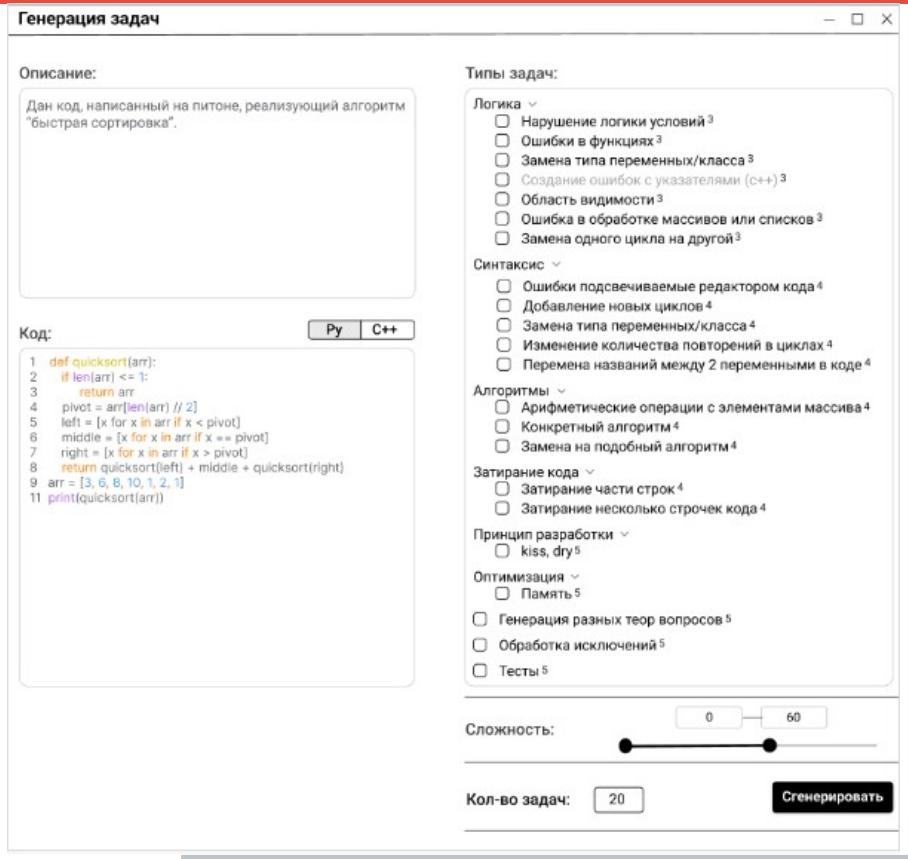
Изучение теоретического материала

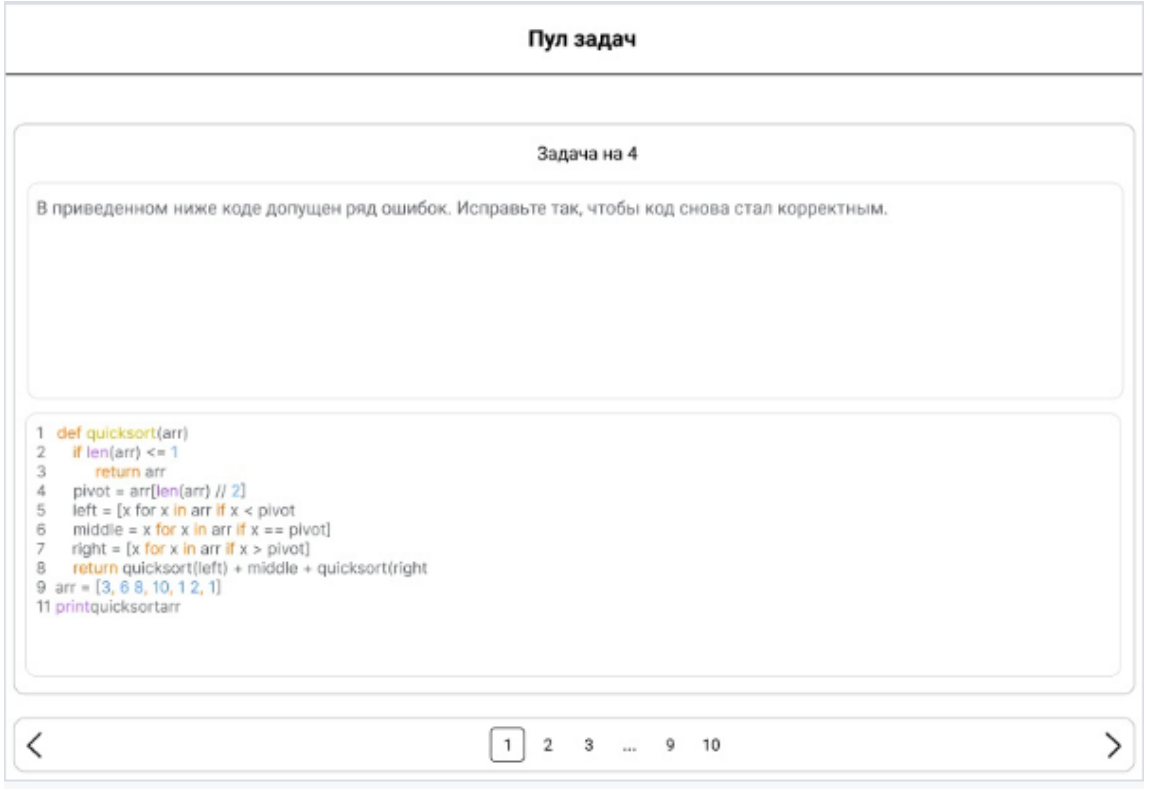
Был изучен теоретический материал, который понадобится для выполнения задачи. Результатом изучения стал перечень библиотек и статей, которые помогут в написании кода.

- AST
- Codegen
- SpaCy
- CodeBERT
- NLTK

Полный перечень находится в репозитории

Макет UI


Генерация задач
Описание:
Дан код, написанный на питоне, реализующий алгоритм "быстрая сортировка".
Код:
Ру
C++
1 def quicksort(arr):
2 if len(arr) <= 1:
3 return arr
4 pivot = arr[len(arr) // 2]
5 left = [x for x in arr if x < pivot]
6 middle = [x for x in arr if x == pivot]
7 right = [x for x in arr if x > pivot]
8 return quicksort(left) + middle + quicksort(right)
9 arr = [3, 6, 8, 10, 1, 2, 1]
11 print(quicksort(arr))
Типы задач:
Логика
Нарушение логики условий 3
Ошибки в функциях 3
Замена типа переменных/класса 3
Создание ошибок с указателями (c++) 3
Область видимости 3
Ошибка в обработке массивов или списков 3
Замена одного цикла на другой 3
Синтаксис
Ошибки подсвечиваемые редактором кода 4
Добавление новых циклов 4
Замена типа переменных/класса 4
Изменение количества повторений в циклах 4
Перемена названий между 2 переменными в коде 4
Алгоритмы
Арифметические операции с элементами массива 4
Конкретный алгоритм 4
Замена на подобный алгоритм 4
Затирание кода
Затирание части строк 4
Затирание несколько строчек кода 4
Принцип разработки
kiss, dry 5
Оптимизация
Память 5
Генерация разных теор вопросов 5
Обработка исключений 5
Тесты 5
Сложность:
0
60
Кол-во задач:
20
Сгенерировать


Пул задач
Задача на 4
В приведенном ниже коде допущен ряд ошибок. Исправьте так, чтобы код снова стал корректным.
1 def quicksort(arr)
2 if len(arr) <= 1
3 return arr
4 pivot = arr[len(arr) // 2]
5 left = [x for x in arr if x < pivot
6 middle = x for x in arr if x == pivot]
7 right = [x for x in arr if x > pivot]
8 return quicksort(left) + middle + quicksort(right
9 arr = [3, 6 8, 10, 1 2, 1]
11 printquicksortarr
1
2
3
...
9
10

Сценарии использования

Были добавлены сценарии использования приложения, с которыми также можно ознакомиться в репозитории проекта

План на следующую итерацию

Определение опорной технологии для реализации проекта и её освоение

Создание первого прототипа программы с работающим одним или более типом задач