

Рабочий лист №1

Дата "01" февраля 2015 г.
(заполняется оргкомитетом)

Шифр ЛН-3
(заполняется оргкомитетом)

Оценка работы

(таблица заполняется по итогам проверки работы членами жюри олимпиады)

№ задания	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Итого (итоговый балл, подпись председателя жюри)
Балл																80
№ задания	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
Балл																113

Математика

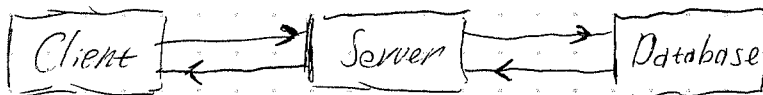
(название олимпиады, заполняется участником)

Тринадцатая информатика

(профиль олимпиады, заполняется участником)

15-10

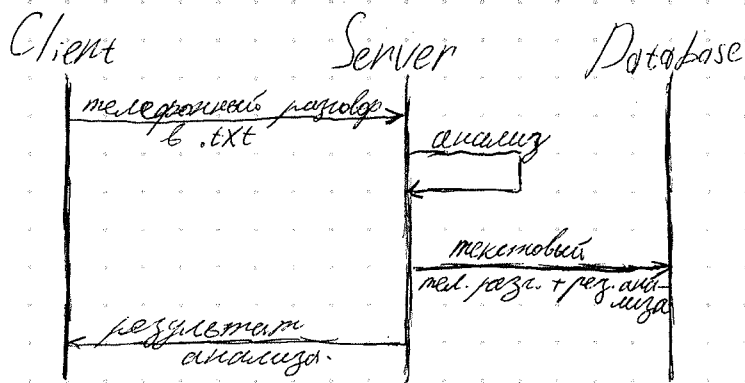
Задача №1.



Client: программа, работающая на стороне клиента. Слушает и обрабатывает телефонный звонок. Переводит на сервер его текстовый вид. В своём интерфейсе имеет поле для получения результата анализа.

Server: программа, работающая на стороне сервера. Принимает текстовый файл телефонного разговора. Анализ разговора может происходить с использованием машинного обучения, то есть проверка содержания по заранее подготовленным шаблонам. Также будет происходить дополнительное обучение искусственного интеллекта, в случае выявления новых шаблонов. После анализа разговора отправляет на клиенту ответ о наличии мошенничества. Также отправляет в базу данных текстовый вид разговора и результат проверки, который можно будет использовать в роли примера.

Database: хранилище данных, содержит в себе телефонные разговоры и результаты анализа.



Также при разработке можно воспользоваться, например, MVP моделью, что позволяет легко масштабировать программу. (Model, View, Presenter)

Использование "циклического" шаблона проектирования позволит вносить улучшения и доработки.

Задача N2.

Входными данными будет текстовый расшифрованный телеграфный разговор. Текстовое сообщение можно делить по словам или по предложениям. Анализ можно проводить с помощью регулярных выражений и функции искусственного интеллекта. Для вычисления вероятности, находим отношение найденных "опасных" слов или предложений к общему количеству слов/предложений. Используем хранилище для запоминания наиболее встречающихся шаблонов при массовых числах, для моментального получения результата. *Записываем данные в виде "слово: количество", где "слово" - это слова которые часто встречаются при массовых числах, а "количество" - сколько раз слово встречалось в случае массовых чисел за время анализа различных разговоров, и если это количество больше определенного значения, то сразу определяем случай массовых чисел.

1. Получаем текстовое сообщение
2. Делим файл по словам/предложениям.
3. Проверяем, есть ли слово/предложение в хранилище, если да и количество встречаемых таких слов/предложений больше, сообщаем о массовых числах, если нет, то пункт 4.
4. Анализируем сообщение, находим отношение найденных "опасных" слов/предложений к общему количеству.
5. Сообщаем результат (вероятность).

Дополнительный рабочий лист
(без рабочего листа №1 недействителен)

Дата "01" февраля 2025 г.
(заполняется участником)

Шифр ММ-З
(заполняется участником)

Задача №3

/ * резюмированное выражение для поиска * /

```
let passportReg = new RegExp("/(0-9)*4(5|10-9)*6/", "g");
```

```
let sniSReg = new RegExp("/(0-9)*31S(0-9)*8/", "g");
```

15-крател
"д" - филм филм
поиска совпаде-
ний по всей
строке.

```
let bankReg = new RegExp("/(0-9)*4(0-9)*4(0-9)*4(0-9)*4/", "g");
```

```
Const parse = (message: string) => {
```

let map: (String, String[]) = {}.

```
let passportResult = passport.match(passportReg, match(message),
```

```
let snitsResult = snitsReg.match(message);
```

```
let bankResult = bankReg.match(message);
```

passportResult for Each item \Rightarrow {
map[passport].push(item);

3/

```
snisResult. forEach (item => {
    map[snis]. push(item);
```

3)

```
bankResult.forEach(item => {
    map[bank].push(item);
});
```

子)

return ~~the~~ map;

3)

На вход в назначение сообщения может прийти текст
всех файлов телеграфного разбора или собранные из "мессенджера".