

Рабочий лист №1

Дата "1" февраля 2025 г.  
(заполняется оргкомитетом)

Шифр ПИ-1  
(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																82
№ задания	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
Балл																93

Мониторинг

(название олимпиады, заполняется участником)

Прикладная информатика

(профиль олимпиады, заполняется участником)

GPT!

1) Архитектура системы была минимальна

Основные компоненты:

1) модуль сбора данных:

- источники данных (операторы чата, базы данных на компьютере, старинный API, файлы файлов)

- технологии: скрининг данных, Rest API, Web Sockets

2) Механизм хранения данных

- база данных для хранения файлов, текстовых расшифровок

- технологии: PostgreSQL, SQL для анализа, S3 для аудио

3) модуль обработки и преобразования

- расш. файлов (выделение ключевых слов, анализ тональности)

- технологии: Deep Speech / Vost

4) модуль ИИ

- обучение и развертывание модели генерации минимальности

- мех: TensorFlow, PyTorch

Сдано 2 листа



Дополнительный рабочий лист  
(без рабочего листа №1 недействителен)

Дата " 1 " февраля 20 25 г.  
(заполняется участником)

Шифр ПУ-1  
(заполняется участником)

3) Извлечение критической пере. информации

30

~~import re~~ 1) предложения о структуре

1) паспорт (паспорт ~~№~~ (данные паспорта, серия) {4 цифра}  
номер {6 цифра})

2) СНИЛС ({3 цифра}-{3 цифра}-{3 цифра} {2 цифра})

3) кредитка ({4 цифра} {4 цифра} {4 цифра} {4 цифра})

2) код

import re

def extract\_sensitive\_data(text):

" text - расшифровка или разделение  
return - словарь с найденными пере. данными "

→ patterns = {

"passport": r"(?:паспорт|данные моего паспорта|серия\s?(1\d{4})/\s?номер\s?)?/\s?(1\d{4})\s?(1\d{6})"

"SNILS": r"(1\d{3})[-\s]?(\d{3})[-\s]?(\d{3})?(\d{2})"

"CreditCard": r"(1\d{4})?(1\d{4})?(1\d{4})?(1\d{4})"

extracted\_data = {}

for key, pattern in patterns.items():

matches = re.findall(pattern, text)

if matches:

extracted\_data[key] = ["".join(match.strip() for match in matches)]

return extracted\_data