**Theory**

1. **What are the main problems of modern NLP and NLU?**

Challenges that modern NLP and NLU face include:

* Building the appropriate vocabulary – given a text, algorithm should be able to generate a sophisticated set of vocabulary in order to gain some insights of the text
* Setting the context – similar words or even sentences may have different meaning based on the context, and algorithm should be able to derive context and proper domain choice
* Extracting named entities – an algorithm should distinguish between simple vocabulary and named entities
* Disambiguation – this may include resolving lexical ambiguity, syntactic ambiguity and semantic ambiguity

1. **Which libraries would you pick to use for the following cases and why (all problems should be solved for the Russian)**

* Sentiment analysis – for sentiment analysis I would use Gensim library, which provides great technologies called Word2Vec and Doc2Vec for extracting features, because it is scalable, robust and efficient in feature extraction, and also have some pretrained models. And then I would choose keras library for building neural networks with obtained features.
* Multi-label classification – here I also would use Gensim for feature extraction, and then scikit-multilearn, which is built on top of scikit-learn, which is very simple to use, or some more complicated technologies, such as keras or tensorflow for building neural networks.
* Dependency parsing – for dependency parsing I would us spaCy, because it is fast because of Cython support and has some outstanding and useful language models pre-installed
* POS-tagging – I would use the most well-known library nltk, because it is one of the most maintained and has a plenty plenty of approaches
* NER – for NER I would also use nltk, because it also provides many options, one of which is a wrapper for the Stanford NER

1. **How would you evaluate a classification model, which metrics would you use?**

There are several metrics for evaluating classification model, and they should be chosen according to the problem. Some of them are:

* Accuracy - proportion of true results among the total number of cases examined, should be used when there is no class imbalance
* Precision – proportion of truly positives to all positive results, should be used when we want to be very sure of our prediction.
* Recall – what proportion of actual positives is correctly classified, should be used when we want to capture as many positives as possible.
* F1 score – harmonic mean of precision and recall, should be used when we want to have a model with both good precision and recall.
* AUC - how well the probabilities from the positive classes are separated from the negative classes, should be used when how well predictions are ranked is more important than their absolute values.

1. **Main pipeline for the text-preprocessing**

* Cleaning – removing special characters, keeping only what can be useful for the context
* Tokenization – splitting phrases into tokens
* POS tagging – part-of-speech tagging, so that token carries more information than just a string of characters
* Lemmatizing (or stemming, normalization) – reducing a word to its lemma (core representative word)
* Stop words removal – remove words that will not help in building meaningful features

1. **Microservices or monoliths? Why.**

I think it is better to split applications into microservices, because such architecture is more flexible (all the services can be deployed and updated independently), a bug in one service only influences that particular service but not the whole system, it is easier to add new features, understand the structure of the system and manage it, and each service can be scaled independently. So, if building long-term project, for future reliability it is useful to use microservices.

1. **Describe the hardest programming task you’ve been facing with. It’s not necessarily ML task, could be just a programming. Why this task was hard to accomplish? What was your solution for the task? Can you share a github project?**

The hardest task I have been recently facing with was developing my first production level Android application. The task was hard to accomplish, because I had no previous experience in that, I even didn’t know the language I was writing on (Kotlin) and also because the application had to use a lot of third-party libraries, such as Yandex Map Kit with which I was not previously familiar with. I were able to accomplish the task with the help of Senior developer, who managed me during the whole project, helped me to choose appropriate architecture, and also showed me some ready-made projects which I could refer to while facing some problems.

1. **Did you work with VCS? Which one?**

No

1. **Did you work with Github Actions?**

No

1. **How familiar are you with Docker and other orchestration tools?**

No

1. **What is ed25519 and why is it concerning to be better than ecdsa?**

Ed25519 is one of the strongest and fastest encryption algorithms. It is concerning to be better than ecdsa, because the latter require a parameter, which has to be completely random, secret, and unique. But if your machine has a poor random number generator, an observer can figure out your private key. So ed25519 is mathematically more powerful.

1. **Do you have any experience in data mining?**

No