Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Кубанский государственный университет»

Кафедра вычислительных технологий

**ОТЧЕТ**

о выполнении лабораторной работы №3

по дисциплине «Основы компьютерной лингвистики»

Выполнил: ст. гр. 36/2

Прозоров М.С.

Проверил: Вишняков Ю.М

Краснодар

2022

**Краткое описание пакетов и функций**

**NLTK** - Пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python. Содержит графические представления и примеры данных. Сопровождается обширной документацией, включая книгу с объяснением основных концепций, стоящих за теми задачами обработки естественного языка, которые можно выполнять с помощью данного пакета.

**Natasha** - набор Python-библиотек для обработки текстов на естественном русском языке. Natasha решает базовые задачи обработки естественного русского языка: сегментация на токены и предложения, морфологический и синтаксический анализ, лемматизация, извлечение, нормализация именованных сущностей.

Программа использует одновременно две библиотке - NLTK и Natasha.

NLTK используется для сегментации текста из моего датасета на предложения. Natasha используется для построения и вывода в консоль дерева представления предложения, полученного после синтаксического разбора. Для всех строк, полученных из текста, разбиваем их содержимое на

предложения, содержащиеся в них. Затем каждое предложение в цикле

сохраняем в переменную sent. После этого, с помощью библиотеки Natasha, выполняем различные виды анализа текста и выводим необходимый нам синтаксический анализ в подробном и графическом в виде через консоль.

**Листинг к заданию 3**

from natasha import Segmenter, NewsEmbedding, NewsMorphTagger, NewsSyntaxParser, Doc

from nltk import sent\_tokenize

def norm(txt):

\_, x = map(int, txt.split('\_'))

return x

path = "dataset.txt"

file = open(path, "r", encoding="utf8")

segmenter = Segmenter()

emb = NewsEmbedding()

morph\_tagger = NewsMorphTagger(emb)

syntax\_parser = NewsSyntaxParser(emb)

words = dict()

tree = {}

lines = file.readlines()

sents = []

for line in lines:

sentences = sent\_tokenize(line)

for i in range(len(sentences)):

sents.append(sentences[i])

def compute(ind):

text = sents[ind]

doc = Doc(text)

doc.segment(segmenter)

doc.tag\_morph(morph\_tagger)

doc.parse\_syntax(syntax\_parser)

root\_id = 0

sent = doc.sents[0]

tokens = [token for token in sent.tokens if token.id]

for token in tokens:

norm\_id = norm(token.id)

words[norm\_id] = token.text

if token.rel == 'root':

root\_id = norm\_id

for i in words.keys():

if i != 0:

tree[i] = []

for token in sent.tokens:

norm\_id = norm(token.id)

norm\_head\_id = norm(token.head\_id)

if (norm\_head\_id != 0):

tree[norm\_head\_id].append(norm\_id)

return sent, root\_id

def print\_result(sent, a):

def lrep(a):

s = '('

s += words[a]

if len(tree[a]) > 0:

for t in tree[a]:

s += lrep(t)

s += ')'

return s

def rrep(a):

s = '('

if len(tree[a]) > 0:

for t in tree[a]:

s += lrep(t)

s += words[a]

s += ')'

return s

print('root:', words[a])

print('words:', words)

print('tree:', tree)

sent.syntax.print()

print('left:', lrep(a))

print('right:', rrep(a))

def main():

for i in range(5):

sent, a = compute(i)

print\_result(sent, a)

print()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Результат работы программы**

Стоит отметить, что на в примере выполнен синтаксический разбор одного предложения. При выполнении программы, разработанной в ходе выполнения лабораторной работы, в консоли выводятся все предложения, с их подробным синтаксическим разбором.

root: Говорим

words: {1: 'Говорим', 2: 'Flutter', 3: '—', 4: 'подразумеваем', 5: 'Dart'}

tree: {1: [2, 4], 2: [], 3: [], 4: [3, 5], 5: []}

Говорим

Flutter

┌► — punct

┌─└─ подразумеваем

└──► Dart obj

left: (Говорим(Flutter)(подразумеваем(—)(Dart)))

right: ((Flutter)(подразумеваем(—)(Dart))Говорим)

[DocToken(stop=7, text='Говорим', id='1\_1', head\_id='1\_0', rel='root', pos='VERB', feats=<Imp,Ind,Plur,1,Pres,Fin,Act>), DocToken(start=8, stop=15, text='Flutter', id='1\_2', head\_id='1\_1', rel='nsubj', pos='PROPN', feats=<Yes>), DocToken(start=16, stop=17, text='—', id='1\_3', head\_id='1\_4', rel='punct', pos='PUNCT'), DocToken(start=18, stop=31, text='подразумеваем', id='1\_4', head\_id='1\_1', rel='parataxis', pos='VERB', feats=<Imp,Ind,Plur,1,Pres,Fin,Act>), DocToken(start=32, stop=36, text='Dart', id='1\_5', head\_id='1\_4', rel='obj', pos='PROPN', feats=<Yes>)]

root: пользуются

words: {1: 'Многие', 2: 'люди', 3: 'уже', 4: 'пользуются', 5: 'приложениями', 6: ',', 7: 'написанными', 8: 'на', 9: 'Flutter', 10: ',', 11: 'но', 12: 'не', 13: 'замечают', 14: 'этого', 15: '.'}

tree: {1: [], 2: [1], 3: [], 4: [2, 3, 5, 13, 15], 5: [7], 6: [], 7: [6, 9], 8: [], 9: [8], 10: [], 11: [], 12: [], 13: [10, 11, 12, 14], 14: [], 15: []}

┌► Многие amod

┌►└─ люди nsubj

│ ┌► уже advmod

┌─┌─────└─└─ пользуются

│ │ ┌─└──► приложениями obj

│ │ │ ┌► , punct

│ │ ┌─└──►└─ написанными acl

│ │ │ ┌► на case

│ │ └────►└─ Flutter obl

│ │ ┌────► , punct

│ │ │ ┌──► но cc

│ │ │ │ ┌► не advmod

│ └──►└─└─└─ замечают conj

│ └──► этого obj

└──────────► . punct

left: (пользуются(люди(Многие))(уже)(приложениями(написанными(,)(Flutter(на))))(замечают(,)(но)(не)(этого))(.))

right: ((люди(Многие))(уже)(приложениями(написанными(,)(Flutter(на))))(замечают(,)(но)(не)(этого))(.)пользуются)

[DocToken(stop=6, text='Многие', id='1\_1', head\_id='1\_2', rel='amod', pos='ADJ', feats=<Nom,Pos,Plur>), DocToken(start=7, stop=11, text='люди', id='1\_2', head\_id='1\_4', rel='nsubj', pos='NOUN', feats=<Anim,Nom,Masc,Plur>), DocToken(start=12, stop=15, text='уже', id='1\_3', head\_id='1\_4', rel='advmod', pos='ADV', feats=<Pos>), DocToken(start=16, stop=26, text='пользуются', id='1\_4', head\_id='1\_0', rel='root', pos='VERB', feats=<Imp,Ind,Plur,3,Pres,Fin,Mid>), DocToken(start=27, stop=39, text='приложениями', id='1\_5', head\_id='1\_4', rel='obj', pos='NOUN', feats=<Inan,Ins,Neut,Plur>), DocToken(start=39, stop=40, text=',', id='1\_6', head\_id='1\_7', rel='punct', pos='PUNCT'), DocToken(start=41, stop=52, text='написанными', id='1\_7', head\_id='1\_5', rel='acl', pos='VERB', feats=<Perf,Ins,Plur,Past,Part,Pass>), DocToken(start=53, stop=55, text='на', id='1\_8', head\_id='1\_9', rel='case', pos='ADP'), DocToken(start=56, stop=63, text='Flutter', id='1\_9', head\_id='1\_7', rel='obl', pos='PROPN', feats=<Yes>), DocToken(start=63, stop=64, text=',', id='1\_10', head\_id='1\_13', rel='punct', pos='PUNCT'), DocToken(start=65, stop=67, text='но', id='1\_11', head\_id='1\_13', rel='cc', pos='CCONJ'), DocToken(start=68, stop=70, text='не', id='1\_12', head\_id='1\_13', rel='advmod', pos='PART', feats=<Neg>), DocToken(start=71, stop=79, text='замечают', id='1\_13', head\_id='1\_4', rel='conj', pos='VERB', feats=<Imp,Ind,Plur,3,Pres,Fin,Act>), DocToken(start=80, stop=85, text='этого', id='1\_14', head\_id='1\_13', rel='obj', pos='PRON', feats=<Inan,Gen,Neut,Sing>), DocToken(start=85, stop=86, text='.', id='1\_15', head\_id='1\_4', rel='punct', pos='PUNCT')]

root: мини-приложение

words: {1: 'Например', 2: ',', 3: 'мини-приложение', 4: 'Gmail', 5: ',', 6: 'приложения', 7: 'BMW', 8: ',', 9: 'eBay', 10: ',', 11: 'Tencent', 12: '.', 13: 'замечают', 14: 'этого', 15: '.'}

tree: {1: [2], 2: [], 3: [1, 4, 6, 12], 4: [], 5: [], 6: [5, 7], 7: [9, 11], 8: [], 9: [8], 10: [], 11: [10], 12: [], 13: [], 14: [], 15: []}

┌► Например parataxis

│ ,

┌─────┌─└─ мини-приложение

│ │ └► Gmail appos

│ │ ┌► , punct

│ └►└─ приложения conj

│ ┌─┌─└──► BMW appos

│ │ │ ┌► , punct

│ │ └──►└─ eBay conj

│ │ ┌► , punct

│ └────►└─ Tencent conj

└────────► . punct

left: (мини-приложение(Например(,))(Gmail)(приложения(,)(BMW(eBay(,))(Tencent(,))))(.))

right: ((Например(,))(Gmail)(приложения(,)(BMW(eBay(,))(Tencent(,))))(.)мини-приложение)

[DocToken(stop=8, text='Например', id='1\_1', head\_id='1\_3', rel='parataxis', pos='ADV', feats=<Pos>), DocToken(start=8, stop=9, text=',', id='1\_2', head\_id='1\_1', rel='punct', pos='PUNCT'), DocToken(start=10, stop=25, text='мини-приложение', id='1\_3', head\_id='1\_0', rel='root', pos='NOUN', feats=<Inan,Nom,Masc,Sing>), DocToken(start=26, stop=31, text='Gmail', id='1\_4', head\_id='1\_3', rel='appos', pos='PROPN', feats=<Yes>), DocToken(start=31, stop=32, text=',', id='1\_5', head\_id='1\_6', rel='punct', pos='PUNCT'), DocToken(start=33, stop=43, text='приложения', id='1\_6', head\_id='1\_3', rel='conj', pos='NOUN', feats=<Inan,Nom,Neut,Plur>), DocToken(start=44, stop=47, text='BMW', id='1\_7', head\_id='1\_6', rel='appos', pos='PROPN', feats=<Yes>), DocToken(start=47, stop=48, text=',', id='1\_8', head\_id='1\_9', rel='punct', pos='PUNCT'), DocToken(start=49, stop=53, text='eBay', id='1\_9', head\_id='1\_7', rel='conj', pos='PROPN', feats=<Yes>), DocToken(start=53, stop=54, text=',', id='1\_10', head\_id='1\_11', rel='punct', pos='PUNCT'), DocToken(start=55, stop=62, text='Tencent', id='1\_11', head\_id='1\_7', rel='conj', pos='PROPN', feats=<Yes>), DocToken(start=62, stop=63, text='.', id='1\_12', head\_id='1\_3', rel='punct', pos='PUNCT')]

root: используется

words: {1: 'В', 2: 'Яндексе', 3: ',', 4: 'в', 5: 'частности', 6: 'в', 7: 'Яндекс', 8: 'Про', 9: ',', 10: 'фреймворк', 11: 'Flutter', 12: 'используется', 13: 'для', 14: 'написания', 15: 'мобильных', 16: 'приложений', 17: 'на', 18: 'Dart', 19: '.'}

tree: {1: [], 2: [1], 3: [], 4: [5], 5: [], 6: [], 7: [3, 4, 6, 8, 9], 8: [], 9: [], 10: [11], 11: [], 12: [2, 7, 10, 14, 19], 13: [], 14: [13, 16], 15: [], 16: [15, 18], 17: [], 18: [17], 19: []}

┌► В case

┌──────►└─ Яндексе obl

│ ┌────► , punct

│ │ ┌►┌─ в discourse

│ │ │ └► частности fixed

│ │ │ ┌► в case

│ ┌►└─└─└─ Яндекс obl

│ │ │ └──► Про nmod

│ │ └────► , punct

│ │ ┌►┌─ фреймворк nsubj:pass

│ │ │ └► Flutter appos

┌─└─└─┌─└─── используется

│ │ ┌► для case

│ ┌─└──►└─ написания obl

│ │ ┌► мобильных amod

│ ┌─└────►└─ приложений nmod

│ │ ┌► на case

│ └──────►└─ Dart nmod

└──────────► . punct

left: (используется(Яндексе(В))(Яндекс(,)(в(частности))(в)(Про)(,))(фреймворк(Flutter))(написания(для)(приложений(мобильных)(Dart(на))))(.))

right: ((Яндексе(В))(Яндекс(,)(в(частности))(в)(Про)(,))(фреймворк(Flutter))(написания(для)(приложений(мобильных)(Dart(на))))(.)используется)

[DocToken(stop=1, text='В', id='1\_1', head\_id='1\_2', rel='case', pos='ADP'), DocToken(start=2, stop=9, text='Яндексе', id='1\_2', head\_id='1\_12', rel='obl', pos='PROPN', feats=<Inan,Loc,Masc,Sing>), DocToken(start=9, stop=10, text=',', id='1\_3', head\_id='1\_7', rel='punct', pos='PUNCT'), DocToken(start=11, stop=12, text='в', id='1\_4', head\_id='1\_7', rel='discourse', pos='ADP'), DocToken(start=13, stop=22, text='частности', id='1\_5', head\_id='1\_4', rel='fixed', pos='NOUN', feats=<Inan,Loc,Fem,Sing>), DocToken(start=23, stop=24, text='в', id='1\_6', head\_id='1\_7', rel='case', pos='ADP'), DocToken(start=25, stop=31, text='Яндекс', id='1\_7', head\_id='1\_12', rel='obl', pos='PROPN', feats=<Inan,Acc,Masc,Sing>), DocToken(start=32, stop=35, text='Про', id='1\_8', head\_id='1\_7', rel='nmod', pos='PROPN', feats=<Yes>), DocToken(start=35, stop=36, text=',', id='1\_9', head\_id='1\_7', rel='punct', pos='PUNCT'), DocToken(start=37, stop=46, text='фреймворк', id='1\_10', head\_id='1\_12', rel='nsubj:pass', pos='NOUN', feats=<Inan,Nom,Masc,Sing>), DocToken(start=47, stop=54, text='Flutter', id='1\_11', head\_id='1\_10', rel='appos', pos='PROPN', feats=<Yes>), DocToken(start=55, stop=67, text='используется', id='1\_12', head\_id='1\_0', rel='root', pos='VERB', feats=<Imp,Ind,Sing,3,Pres,Fin,Pass>), DocToken(start=68, stop=71, text='для', id='1\_13', head\_id='1\_14', rel='case', pos='ADP'), DocToken(start=72, stop=81, text='написания', id='1\_14', head\_id='1\_12', rel='obl', pos='NOUN', feats=<Inan,Gen,Neut,Sing>), DocToken(start=82, stop=91, text='мобильных', id='1\_15', head\_id='1\_16', rel='amod', pos='ADJ', feats=<Gen,Pos,Plur>), DocToken(start=92, stop=102, text='приложений', id='1\_16', head\_id='1\_14', rel='nmod', pos='NOUN', feats=<Inan,Gen,Neut,Plur>), DocToken(start=103, stop=105, text='на', id='1\_17', head\_id='1\_18', rel='case', pos='ADP'), DocToken(start=106, stop=110, text='Dart', id='1\_18', head\_id='1\_16', rel='nmod', pos='PROPN', feats=<Yes>), DocToken(start=110, stop=111, text='.', id='1\_19', head\_id='1\_12', rel='punct', pos='PUNCT')]

root: предполагает

words: {1: 'Поэтому', 2: 'внутри', 3: 'мы', 4: 'их', 5: 'не', 6: 'разделяем', 7: '—', 8: 'работа', 9: 'с', 10: 'Flutter', 11: 'предполагает', 12: 'знание', 13: 'Dart', 14: '.', 15: 'мобильных', 16: 'приложений', 17: 'на', 18: 'Dart', 19: '.'}

tree: {1: [], 2: [], 3: [], 4: [], 5: [], 6: [1, 2, 4, 5, 7, 8], 7: [], 8: [10], 9: [], 10: [9], 11: [3, 12, 14], 12: [13], 13: [], 14: [], 15: [], 16: [], 17: [], 18: [], 19: []}

┌──────► Поэтому advmod

│ ┌────► внутри advmod

┌►│ │ мы nsubj

│ │ │ ┌──► их obj

│ │ │ │ ┌► не advmod

│ └─└─└─└─ разделяем

│ │ └──► — punct

│ ┌─└────► работа nsubj

│ │ ┌► с case

│ └────►└─ Flutter nmod

└───┌───┌─ предполагает

│ ┌─└► знание obj

│ └──► Dart nmod

└────► . punct

left: (предполагает(мы)(знание(Dart))(.))

right: ((мы)(знание(Dart))(.)предполагает)

[DocToken(stop=7, text='Поэтому', id='1\_1', head\_id='1\_6', rel='advmod', pos='ADV', feats=<Pos>), DocToken(start=8, stop=14, text='внутри', id='1\_2', head\_id='1\_6', rel='advmod', pos='ADV', feats=<Pos>), DocToken(start=15, stop=17, text='мы', id='1\_3', head\_id='1\_11', rel='nsubj', pos='PRON', feats=<Nom,Plur,1>), DocToken(start=18, stop=20, text='их', id='1\_4', head\_id='1\_6', rel='obj', pos='PRON', feats=<Inan,Acc,Plur,3>), DocToken(start=21, stop=23, text='не', id='1\_5', head\_id='1\_6', rel='advmod', pos='PART', feats=<Neg>), DocToken(start=24, stop=33, text='разделяем', id='1\_6', head\_id='1\_0', rel='root', pos='VERB', feats=<Imp,Ind,Plur,1,Pres,Fin,Act>), DocToken(start=34, stop=35, text='—', id='1\_7', head\_id='1\_6', rel='punct', pos='PUNCT'), DocToken(start=36, stop=42, text='работа', id='1\_8', head\_id='1\_6', rel='nsubj', pos='NOUN', feats=<Inan,Nom,Fem,Sing>), DocToken(start=43, stop=44, text='с', id='1\_9', head\_id='1\_10', rel='case', pos='ADP'), DocToken(start=45, stop=52, text='Flutter', id='1\_10', head\_id='1\_8', rel='nmod', pos='PROPN', feats=<Yes>), DocToken(start=53, stop=65, text='предполагает', id='1\_11', head\_id='1\_0', rel='root', pos='VERB', feats=<Imp,Ind,Sing,3,Pres,Fin,Act>), DocToken(start=66, stop=72, text='знание', id='1\_12', head\_id='1\_11', rel='obj', pos='NOUN', feats=<Inan,Acc,Neut,Sing>), DocToken(start=73, stop=77, text='Dart', id='1\_13', head\_id='1\_12', rel='nmod', pos='PROPN', feats=<Yes>), DocToken(start=77, stop=78, text='.', id='1\_14', head\_id='1\_11', rel='punct', pos='PUNCT')]

**Приложение**

**Датасет, используемый в ЛР №3**

Говорим Flutter — подразумеваем Dart

Многие люди уже пользуются приложениями, написанными на Flutter, но не замечают этого. Например, мини-приложение Gmail, приложения BMW, eBay, Tencent.

В Яндексе, в частности в Яндекс Про, фреймворк Flutter используется для написания мобильных приложений на Dart. Поэтому внутри мы их не разделяем — работа с Flutter предполагает знание Dart.

Фреймворк — это набор готовых абстракций, участков кода, функций и методов, которые можно подключить в Dart-проект всего одной строчкой. А ещё Flutter — это целая идеология и набор правил: как писать код и какие приёмы использовать. Это целый мир, существующий по своим законам и традициям.

Преимущества Flutter для разработчиков

— Кроссплатформенность. На Flutter легко переходить тем, кто уже пробовал создавать мобильные приложения на iOS или Android. При этом код получается качественным, потому что у команды появляется больше времени на доработку и тестирование: не нужно распылять внимание на несколько приложений, чтобы проверить, как в них реализованы одни и те же функции.

— Работать на Flutter может одна команда вместо двух или даже трёх. У нас в команде есть люди, которые пришли с хорошим знанием Flutter, есть бывшие Android- и iOS-разработчики. Все они ценны тем, что знают специфику разных приложений и при этом решают совместные задачи.

— Для того чтобы сделать хорошее коммерческое приложение, достаточно выучить всего один язык — Dart. Для сравнения: в нативной разработке нужно разобраться в специфике Android и выучить минимум два языка — Java и XML. К тому же придётся повозиться с Android-специфичными штуками. В случае с Flutter достаточно задать в консоли проект и писать его на Dart, чтобы создать приложение, которое работает «из коробки». Да, в дальнейшем для некоторых фич, связанных с платформенной спецификой, понадобится и нативность, но базово будет достаточно Dart.

— У Dart низкий порог вхождения, поэтому освоить Flutter можно довольно быстро. Особенно если вы уже знаете несколько языков программирования. Так было со мной: когда я начинал учить Flutter, я просто открыл документацию Dart, прочёл ее часа за три, и начал на нём писать. Какие-то сверхнавыки, вроде функционального программирования, здесь не нужны. Освоите базу — и это уже даст вам возможность начать программировать на Flutter. Легко переходить на Flutter и тем, кто пробовал писать приложения на React. JS и Dart похожи реактивным UI, подходами к управлению состоянием и вёрсткой в коде.

— Flutter позволяет больше времени уделять тому, что важно для пользователя: приятному UI, стабильной работе, так как реализация многих фич на Flutter занимает ощутимо меньше времени, чем их нативные аналоги.

— Этот фреймворк идеален для прототипирования, особенно в том случае, когда хочется просто проверить какую-то гипотезу: можно за пару дней сделать приложение и протестировать его. Flutter обеспечивает очень высокую скорость разработки, если вы умеете с ним обращаться и хорошо знакомы с API Flutter.

Кроссплатформенность является не только главным преимуществом Flutter, но и причиной, по которой он появился. Часто стоит задача разработать одинаковые приложения для iOS, Android и браузера. Большая компания может позволить себе несколько команд, а игрокам поменьше важно оптимизировать процесс работы. Так получился Flutter. Он похож на бензин, который подойдёт любому автомобилю: с ним заведётся и Android, и iPhone, и десктоп, и Fuchsia, и даже в вебе всё будет работать. Правда, понадобятся дополнительные усилия для того, чтобы приложение было нативным и удобным для пользователей.

Заменяет ли Flutter нативную разработку

Приложение, написанное на Flutter, везде выглядит и работает одинаково — и на Xiaomi, и на iPhone. Это удобно, но в то же время теряется платформенно-специфичный опыт и нативность.

В мобильной разработке нативным считается приложение, написанное под конкретную платформу и до мелочей учитывающее её специфику. Например, для iOS есть набор дизайн-гайдлайнов, описывающих, как должно выглядеть и работать хорошее приложение. В такие гайды входит анимация, способы прокручивания списков, индикаторы загрузки и другие стандарты.

Существенная разница между операционными системами состоит в концептуальных различиях: по-разному называются и работают одни и те же сущности, отличаются ограничения. С Flutter легко создавать универсальный базовый интерфейс, но для разработки более серьёзных функций — отправки уведомлений, получения геолокации и работы виджетов — нужно знать платформенную специфику.

Для пользователя нативность приложения проявляется в том, что оно выглядит узнаваемым, в нём интуитивно понятный интерфейс, к которому не нужно долго привыкать. Конечно, во Flutter можно написать код так, чтобы приложение выглядело максимально органично, но всё равно останется ощущение, что «что-то не то». Особенно это характерно для Flutter-приложений в iOS. Так что Flutter не является полной заменой нативной разработке.