МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РЕАЛІЗАЦІЯ МЕТОДУ ВИЯВЛЕННЯ ДАТ У ДОВГИХ ТЕКСТАХ»**

Виконав: студент 4 курсу, групи 6.1229
спеціальності 122 комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми комп'ютерні науки
(назва освітньої програми)
В.О. Матвєєв
(ініціали та прізвище)

Керівник старший викладач кафедри комп'ютерних наук, к.т.н. Добровольський Г.А.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти бакалавр

Спеціальність 122 комп'ютерні науки
(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук,
д.т.н., професор

_____ Чопоров С.В.
(підпис)

" ____ " _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Матвєєву Володимиру Олександровичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Реалізація методу виявлення дат у довгих текстах.
керівник роботи Добровольський Геннадій Анатолійович, к.т.н.
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 26 » січня 2023 року № 102-с

2. Строк подання студентом роботи 05.06.2023

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Реалізація методу виявлення дат у довгих текстах

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 27.01.2023

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	27.01.2023	
2.	Збір вихідних даних.	27.02.2023	
3.	Обробка методичних та теоретичних	27.03.2023	
	джерел.		
4.	Розробка першого та другого розділу.	27.04.2023	
5.	Розробка третього розділу.	27.05.2023	
6.	Оформлення та нормоконтроль		
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.		

Студент _____ В.О. Матвєєв
(підпис) (ініціали та прізвище)

Керівник роботи _____ Г.А. Добровольський
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г.Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Реалізація методу виявлення дат у довгих текстах»: 60 с., 20 рис., 4 табл., 11 джерел, 12 додатків.

ВИЯВЛЕННЯ ДАТИ, ДАНІ, ДАТА, ЗАТРАЧЕНО, ЗНАЙДЕНО, НАЙЕКЗОТИЧНІША ДАТА, ПОШУК ДАТИ, РЕАЛІЗАЦІЯ, ПРОГРАМА, ТЕКСТ, ФОРМАТ, ФРАГМЕНТ ТЕКСТУ.

Об’єкт дослідження – дати.

Мета роботи: реалізація методу виявлення у довгих текстах дат різних форматів одночасно із вибором формату.

Метод дослідження – аналітичний.

Кваліфікаційна робота містить дослідницький компонент з пошуку готових рішень поставленого завдання, тестування та аналіз відповідно до вимог до програми. Уточнення та розробка архітектури програми на основі обраного стеку технологій. Практична частина містить етапи проектування та реалізації програми та подальше її тестування. Внаслідок аналізу типової статистики зроблено висновок, що розроблена програма справляється з завданням без критичних помилок.

SUMMARY

Bachelor's qualifying theses « Implementing a method for detecting dates in long texts»: 60 pages, 20 figures, 4 tables, 11 references, 12 supplements.

DATE DETECTION, DATA, DATE, SPENT, FOUND, MOST EXOTIC DATE, DATE SEARCH, IMPLEMENTATION, PROGRAM, TEXT, FORMAT, TEXT FRAGMENT.

Object of the study – dates.

Aim of the study: implementation of a method for detecting dates of different formats in long texts simultaneously with format selection.

Method of research – analytical.

The thesis contains a research component to find ready-made solutions to the task, testing and analysis in accordance with the requirements for the program. Refinement and development of the program architecture based on the selected technology stack. The practical part includes the stages of designing and implementing the program and its subsequent testing. Based on the analysis of typical statistics, it is concluded that the developed program copes with the task without critical errors.

ЗМІСТ

ВСТУП

Аналіз даних з часовою компонентою - це важлива складова будь-якого дослідження, оскільки час може мати значний вплив на дані та їх інтерпретацію. Цей аналіз дозволяє виявити різноманітні залежності між даними та часом, а також зробити відповідні висновки та прогнози.

Наприклад, аналіз даних з часовою компонентою дозволяє прогнозувати майбутні тенденції, що може бути важливим для прийняття важливих рішень в бізнесі або при плануванні дій на майбутнє. На основі аналізу даних можна прогнозувати зміни в попиті на різні товари та послуги, що дозволить підготуватися до змін та прийняти відповідні заходи. Аналіз даних з часовою компонентою може допомогти виявити аномалії та незвичайні зміни у даних. Виявлення несподіваних змін у поведінці користувачів на сайті може допомогти виявити проблеми з функціональністю сайту або з рекламними кампаніями.

Першим кроком у дослідженні даних із часовою компонентою є виявлення дат та приведення їх до стандартного типу datetime, але формат дат може бути різним і залежати від походження даних.

Наприклад, 2 квітня 2023 року може записуватись як "2 April 2023" “02/04/23”, “02/04/2023”, “04/02/2023”, “02-04-2023”, “02.04.2023”, “2023年04月02日”. Найскладнішим у наведеному прикладі є виявлення точного значення дат типу “04/02/2023”, які можуть означати як “2 квітня 2023 року” так і “4 лютого 2023 року”, в залежності від походження даних та налаштувань програмного забезпечення, задіяного у зборі даних.

Ще більшої складності додає необхідність виявляти дати невідомого наперед формату у текстах, написаних природною мовою. Такими текстами можуть бути новини із різних джерел, наприклад, у тексті новини із сайта ЗНУ (рис. 1) згадується кілька дат:

а) дата публікації новини 07.04.2023 09:20;

б) дата наказу 27.03.2023;

в) період із 12 по 14 квітня 2023 року;

г) дата зустрічі “13 квітня” (рік не вказано).

Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі

07.04.2023 09:20

Відповідно до наказу Нацагентства від 27.03.2023 року № 623-Е, у період із 12 по 14 квітня 2023 року, в Запорізькому національному університеті (ZNU, Zaporizhzhia National University) експертна група проводитиметься у віддаленому (дистанційному) режимі акредитаційну експертизу за спеціальністю 033 «Філософія» освітньої програми «Філософія» за третім (освітньо-науковим) рівнем вищої освіти.

Відповідно до програми візиту експертної групи, у четвер, 13 квітня, відбудеться відкрита зустріч із усіма охочими учасниками/цями освітнього процесу (крім гаранта ОНП та адміністрації ЗВО).

Зустріч відбудеться 13 квітня з 11:20 до 11:50.

Рисунок 1 – Текст новини із сайта ЗНУ

Згадані дати мають різні формати, але всі дати потрібно виявити та правильно перетворити на стандартний тип даних datetime. Складним випадком є дата “07.04.2023”, яка може означати “7 квітня 2023 року” або “4 липня 2023 року”.

Метою роботи є реалізація методу виявлення у довгих текстах дат різних форматів одночасно із вибором формату. Задачами роботи є

– огляд наявних методів виявлення дат;
– реалізація пошуку підрядків з датами у текстах, написаних природною мовою;
– перетворення знайдених підрядків з датами на стандартний формат datetime мови програмування python ;

– статистичний аналіз усіх спроб перетворення підрядків та визначення формату дат, використаного у вхідних даних;
– перевірка якості реалізованого методу виявлення дат.

1 АНАЛІЗ ЛІТЕРАТУРИ, ІСНУЮЧИХ БІБЛІОТЕК: ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ПРОБЛЕМИ, АНАЛІЗ ТЕХНОЛОГІЙ ТА ІНСТРУМЕНТІВ, ВИБІР МЕТОДІВ ТА АЛГОРИТМІВ

1.1 Аналіз задачі

Задача полягає в реалізації методу виявлення у довгих текстах дат різних форматів одночасно із вибором формату. Підхід до аналізу дат із текстів включає виділення слів та вилучення числових значень за допомогою шаблонів, регулярних виразів та методів пошуку/парсингу. Після отримання числових значень їх можна порівняти з попередньо відомими вибраними форматами. Наступний крок – форматування дати до єдиного виду та можливість повернути пару (номер_формату, дата). Серед неоднозначних форматів виділити той, що зустрічається найчастіше.

Тому важливо визначитися із форматами дат (див. додаток А). Серед них неоднозначні формати під номерами:

2. dd mm yyyy, де mm – рядок: Сомалійський формат(16 Februrari 2023), Італійський формат (1 luglio 2023), Сербский формат(16 februara 2023), Естонський формат(16 Februarii 2023), Польський формат(1 października 2023), Азербайджанский формат(16 Fevral 2023), Румунський формат(16 Februarie 2023), Шведський формат(16 Feber 2023), Словенський формат(16 februar 2023), Український формат(16 лютого 2023), Вірменський формат(16 Փետրվարի, 2023), Хорватський формат(16 veljače 2023), Албанский формат(16 Fevruari 2023), Фінський формат(16 helmikuuta 2023), Люксембурзький формат(16 février 2023), Мальтійський формат(16 Feberwari 2023), Грецький формат(16 Φεβρουαρίου 2023), Чешский формат(16 únor 2023)

9. dd mm yyyy, де dd – цифра(и) дві букви, mm – рядок: Французский формат(1er mars 2023)

27. dd “de” mm “de” yyyy, де mm – рядок: Іспанський формат(13 de enero de 2023)

28. dd mm yyyy, де mm – цифра та символ китайською: Китайский формат(22 8月 2021)

29. mm dd, yyyy, де mm – рядок китайською: Китайський формат(五月 26, 2022)

30. yyyymmdd, де yyyy – чотиризначне число та символ китайською, mm - цифра(и) та символ китайською, dd – цифра(и) та символ китайською: Китайский формат(2023年01月13日), Тайваньский формат(2023年2月16日)

33. mm yyyy, де mm – рядок Німецький формат(September 2023)

1.2 Огляд існуючих рішень проблеми

З огляду на існуючі рішення проблеми виявлення у довгих текстах дат різних форматів, одночасно з вибором формату, знайдено такі Python бібліотеки: dateparser, qddate, DateUtilParser.

Dateparser – модуль із стандартної бібліотеки Python, що надає функцію для розбору дат у рядковому введенні, дозволяючи витягувати інформацію, таку як день/місяць або рік, із заданого рядка часу дати. Використовує регулярні вирази, що дозволяє вказати необхідний формат(и) для пошуку/парсингу. Корисно в випадку необхідності витягти конкретну інформацію з даних, що містять різні формати і/або місцезнаходження дат або часу. Можна використовувати як просунутий, але повільніший за конкурентів модуль.

qddate – призначена для швидкого пошуку будь-яких дат на HTML сторінці. Розроблена спеціально для виявлення, маніпулювання та порівнювання дат. Можна використовувати, надавши вхідний рядок, що містить дату.

DateUtilParser – розширений парсер Python для дат з функціями аналізу та обробки різних форматів. Призначений для розбіру рядків, що містять дати. Надає функції, які дозволяють легко вилучати рік, місяць, день тижня, годину, хвилину та секунду з будь-якого заданого рядка дати.

1.3 Аналіз технологій та інструментів

Виконуючи аналіз технологій та інструментів, серед сучасних бібліотек, існуючих рішень з пошуку та виявлення дат важливо враховувати різні аспекти, в особливості – формування єдиного середовища тестування та вхідних даних, швидкість виконання програми та гнучкість розробки рішення проблеми.

1.3.1 Технічне завдання тестування бібліотек

Для кожної з конкурентних бібліотек проведені 2 тести.

Тест 1

Виявлення найекзотичніших дат за допомогою бібліотеки зі строки тексту (для перевірки обраних бібліотек вхідні дані використані однакові, див. рис. 2).

```
dates = [
    "21 June 2018",
    "16 Februrari 2023",
    "16 Fevral 2023",
    "16 février 2023",
    "16 Φεβρουαρίου 2023",
    "1er mars 2023",
    "13 de enero de 2023",
    "五月 26, 2022",
    "2023年01月13日",
    "September 2023",
]
```

Рисунок 2 – Вхідні дані для тесту 1

Тест 2

Пошук та виявлення дат за допомогою бібліотеки з великого фрагменту тексту (для перевірки обраних бібліотек вхідні дані використані однакові, див. рис. 3).

```
date_text = """Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі
Відповідно до наказу Нацагентства від 27.03.2023 року № 623-Е, у період із 12 по 14 квітня 2023 року, в Запорізькому національному університеті (ZNU, Zaporizhzhia National University) експертна група проводитиметься у віддаленому (дистанційному) режимі акредитаційну експертизу за спеціальністю 033 «Філософія» освітньої програми «Філософія» за третім (освітньо-науковим) рівнем вищої освіти.
ZNU Zaporizhzhia National University ЗНУ Запорізький національний університет факультет соціології (управління філософія Національне агентство забезпечення якості вищої освіти освітній процес акредитаційна експертиза Соціології та управління (https://www.znu.edu.ua/ukr/489/490/504)
Головні новини (https://www.znu.edu.ua/ukr/489/znu-front-news)
29 переглядів
07.04.2023 09:20"""
```

Рисунок 3 – Вхідні дані для тесту 2

1.3.2 Тестування та аналіз бібліотек

Dateparser, тест 1 Повний код програми (див. додаток Б).

Внаслідок виконання програми пошуку та визначення найекзотичніших дат за допомогою Dateparser знайдено 9 дат з 10 та затрачено 1.6960279941558838 с (рис. 4).

2018-06-21 00:00:00 21 June 2018

None 16 Februrari 2023

2023-02-16 00:00:00 16 Fevral 2023

2023-02-16 00:00:00 16 février 2023

2023-02-16 00:00:00 16 Φεβρουαρίου 2023

2023-03-01 00:00:00 1er mars 2023

2023-01-13 00:00:00 13 de enero de 2023

2022-05-26 00:00:00 五月 26, 2022

2023-01-13 00:00:00 2023年01月13日

2023-09-10 00:00:00 September 2023

Час виконання програми: 1.6960279941558838 с

Рисунок 4 – Результат пошуку та визначення дат за допомогою Dateparser

Dateparser, тест 2 Повний код програми (див. додаток В).

Внаслідок виконання програми пошуку дати з великого фрагменту тексту за допомогою Dateparser знайдено 3 дати з 3 та затрачено 0.3367950916290283 с (рис. 5).

2023-03-27 00:00:00 27.03.2023

2023-07-04 00:00:00 07.04.2023

2023-04-14 00:00:00 14 квітня 2023

Час виконання програми: 0.3367950916290283 с

Рисунок 5 – Результат пошуку дати з великого фрагменту тексту за допомогою Dateparser

qdDate, тест 1 Повний код програми (див. додаток Г).

Внаслідок виконання програми пошуку найекзотичніших дат задопомогою qddate знайдено 3 дати з 10 та затрачено 0.05828237533569336 с (рис. 6).

```
2018-06-21 00:00:00
None
None
2023-02-16 00:00:00
None
None
2023-01-13 00:00:00
None
None
None
Час виконання програми:  0.05828237533569336 с
```

Рисунок 6 – Результат пошуку дат за допомогою qddate

qdDate, тест 2 Повний код програми (див. додаток Д).

Внаслідок виконання програми пошуку дати з великого фрагменту тексту за допомогою qddate знайдено 2 дати з 3 та затрачено 0.03363537788391113 с (рис. 7).

```
2023-03-27 00:00:00      27.03.2023
2023-04-07 00:00:00      07.04.2023
None      14 квітня 2023
Час виконання програми:  0.03363537788391113 с
```

Рисунок 7 – Результат пошуку та визначення дати з великого фрагменту тексту за допомогою qddate

DateUtilParser, тест 1 Повний код програми (див. додаток Е).

Внаслідок виконання програми пошуку найекзотичніших дат за допомогою dateutil.parser знайдено 2 дати з 10 та затрачено 0.0029649734497070312 с (рис. 8).

```
2018-06-21 00:00:00
Unknown string format: 16 Februrari 2023
Unknown string format: 16 Fevral 2023
Unknown string format: 16 février 2023
Unknown string format: 16 Φεβρουαρίου 2023
Unknown string format: 1er mars 2023
Unknown string format: 13 de enero de 2023
Unknown string format: 五月 26, 2022
Unknown string format: 2023年01月13日
2023-09-10 00:00:00
Час виконання програми:  0.0029649734497070312 с
```

Рисунок 8 – Результат пошуку дат за допомогою dateutil.parser

DateUtilParser, тест 2

Повний код програми (див. додаток Ж).

Внаслідок виконання програми пошуку дати з великого фрагменту тексту за допомогою dateutil знайдено 2 дати з 3 та затрачено 0.0019943714141845703 с (рис. 9).

```
2023-03-27 00:00:00      27.03.2023
2023-07-04 00:00:00      07.04.2023
Unknown string format: 14 квітня 2023
Час виконання програми:  0.0019943714141845703 с
```

Рисунок 9 – Результат пошуку та визначення дати з великого фрагменту тексту за допомогою dateutil

Статистика наведена в таблицях 1 і 2.

Таблиця 1 – Статистика тесту 1

Бібліотека	Кількість дат	Дат знайдено	Затрачено часу(с)
Dateparser	10	9	1.6960279941558838
qddate	10	3	0.05828237533569336
dateutil.parser	10	2	0.0029649734497070312

Таблиця 2 – Статистика тесту 2

Бібліотека	Кількість дат	Дат знайдено	Затрачено часу(с)
Dateparser	3	3	0.3367950916290283
qddate	3	2	0.03363537788391113
dateutil.parser	3	2	0.0019943714141845703

1.4 Вибір методів та алгоритмів

Серед досліджених бібліотек Dateparser показала найкращій результат по кількості знайдених дат та найгірший по затраченому часу.

Qddate серед переваг має швидку роботу виконання програми. Але меншу кількість знайдених дат, порівняно з Dateparser.

Dateutil.parser має найкращій результат по швидкості роботи виконання програми, але найгірший по кількості знайдених дат.

Обрано бібліотеку dateparser, методи якого найкраще відповідають вимогам проекту, як більш продвинуті, пластичні(та більш повільні) та відповідно до статистики за найкращім результатом.

В стек технологій додається модуль re, функції та методи якого дозволяють маніпулювати регулярними виразами. Використаний для складання шаблонів різних форматів дат.

2 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1 Технічне завдання

Задача полягає в реалізації методу виявлення у довгих текстах дат різних форматів одночасно із вибором формату з використанням стандартної Python бібліотеки dateparser в комбінації з re.

Потрібно:

– скласти перелік форматів дати та перелік шаблонів регулярних виразів;
– реалізувати метод пошуку дати зі строки;
– скласти тести для пошуку всіх форматів дати у довгому тексти з файлу;
– додати можливість повернути пари (номер_формату, дата);
– результат записати до файлу;
– серед неоднозначних форматів вибрати формат, який зустрічається найчастіше.

2.2 Розробка архітектури програмного продукту: опис архітектурних компонентів, залежностей між ними та опис взаємодії

Архітектура відноситься до структури, організації та взаємодії між компонентами в програмному продукті, до яких входять різні функції та методи, що взаємодіють один з одним.

За необхідності пошуку дат різних форматів у довгих текстах слід визначитися з шаблонами, за допомогою яких внаслідок аналізу будуть виділені відповідні слова та числові значення. Тому слід сформувати список з шаблонами дат що містять: регулярні вирази, функцію попередньої обробки дати до єдиного виду та функцію перевірки формату в неоднозначних датах. Список має вигляд як на рис. 10.

```
Шаблони дат = [(регулярний вираз, функція предобробки дати, функція перевірки формату)]
```

Рисунок 10 – Псевдокод списку шаблонів дат

Лямбда вирази можуть бути відсутні.

Наступний компонент зосереджує основну логіку обробки тексту – функція пошуку та виявлення дати зі строки; повертає пари (формат, дата).

Складність виникає, коли в тексті серед однозначно визначених форматів дат міститься неоднозначні.

Можливі такі неоднозначні результати, де даними на вході буде "14 квітня 2023", та після обробки рядка і виявлення дати, що міститься в ній, може виникнути некоректний вихід пар ("14 квітня 2023", 14.04.2023) та ("квітня 2023", 01.04.2023). Подібний результат можна розцінювати як помилку, через що виникає необхідність її виключити. Вирішенням цієї проблеми буде додаткова перевірка на зміст шаблону в іншому шаблоні та його пропускання.

Можливі такі неоднозначні результати, де даними на вході буде "07.04.2023", на виході – ("07.04.2023", 07.04.2023) та ("07.04.2023", 04.07.2023). Тобто два шаблони зчитують одну дату, результат дублюється зі зміною місць дня та місяця. Тому слід врахувати це та привести відповідно до єдиного формату datetime.

Варто врахувати помилки в результаті серед неоднозначних форматів, тому функція має шукати дати в тексті, визначати їх відповідно до шаблону у списку шаблонів, виконувати обробку дати для наступних маніпуляцій, виконувати перевірку на вміст однієї дати в іншій та повертати пари (формат, дата в форматі datetime), одночасно лічити статистику форматів дат, задля подальшого пошуку серед неоднозначних форматів дат, той, що зустрічається найчастіше. Архітектура функції показана на блок схемі (рис. 11).

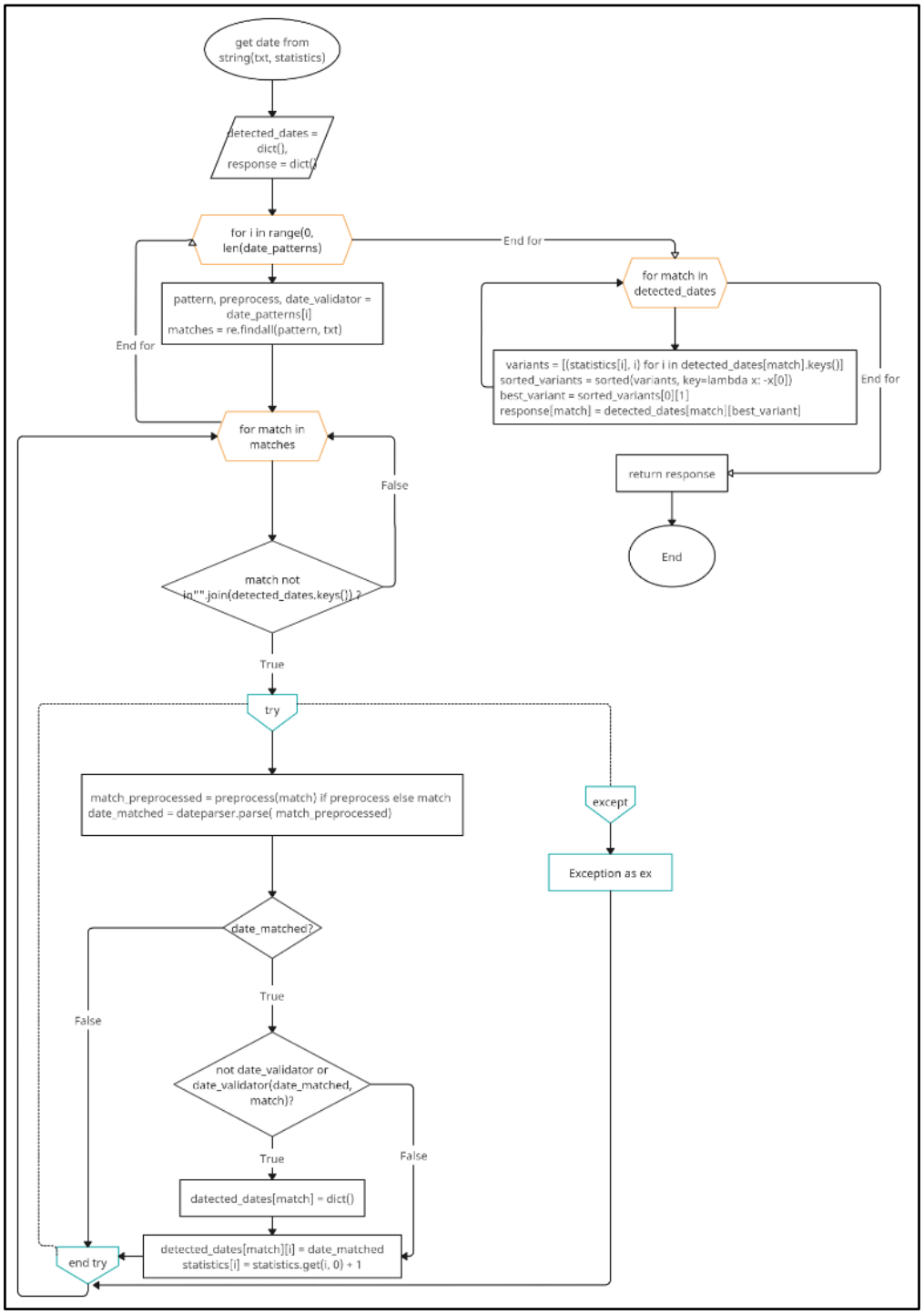

Рисунок 11 – Блок схема функції пошуку та виявлення дати зі строки

Наступний компонент програми – тестування функції пошуку та виявлення дати з рядка. Функція повинна реалізувати зчитування довгого тексту з файлу, тобто відкриття файлу у режимі читання, передавати вхідним параметром текст до функції пошуку та визначення дати з рядка, повертати словник з популярністю форматів дат. Архітектура функції показана на блок схемі (рис. 12).

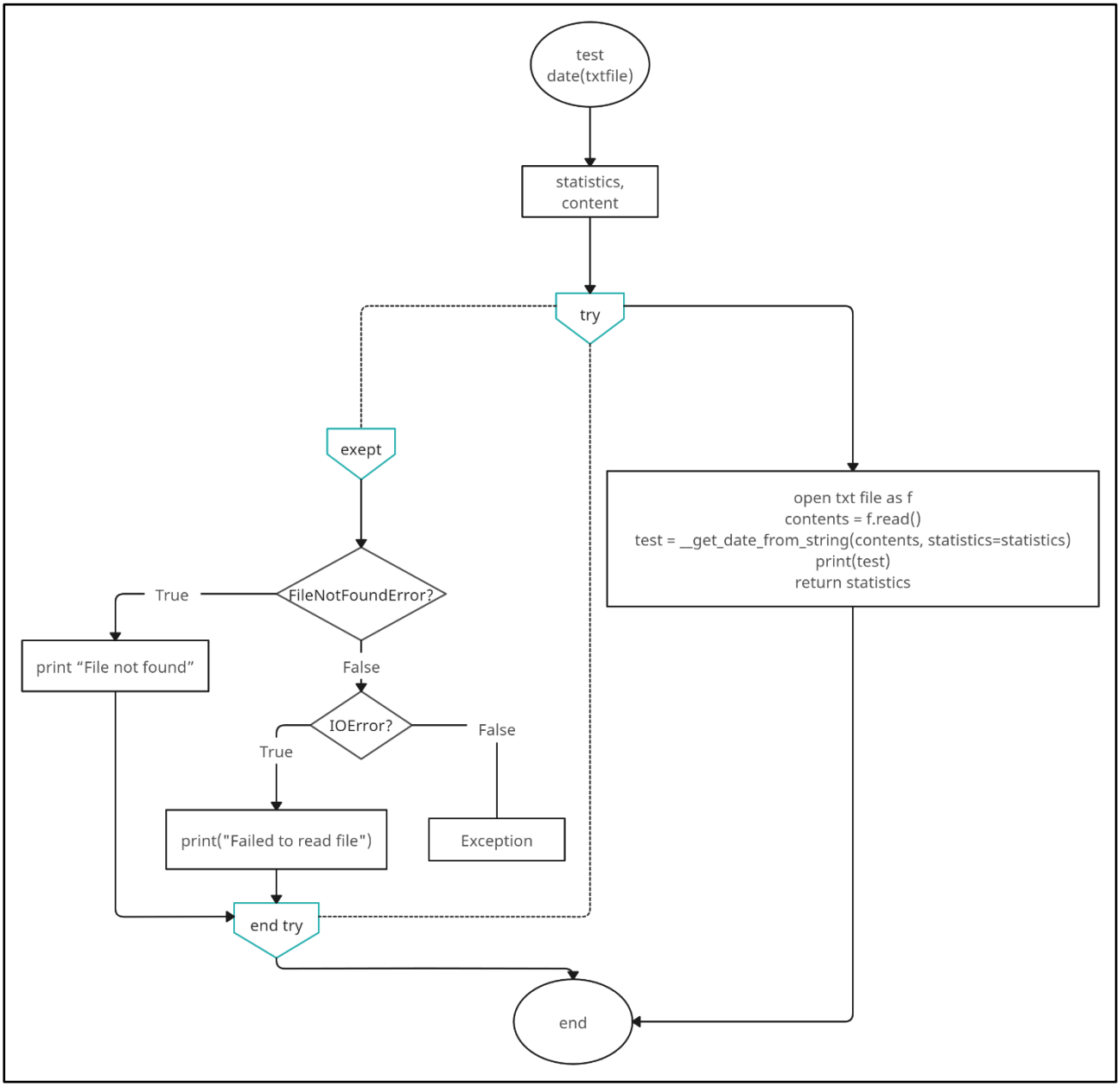

Рисунок 12 – Блок схема функції тестування

Серед неоднозначних форматів необхідно вибрати формат, який зустрічається найчастіше. Реалізовано в функції, що оброблятиме результат роботи тестування функції пошуку та виявлення дати з рядка – словник. Серед компонентів словника, буде виявлений той номер формату, що є неоднозначним та зустрічається найчастіше. Архітектура функції показана на блок схемі (рис. 12).

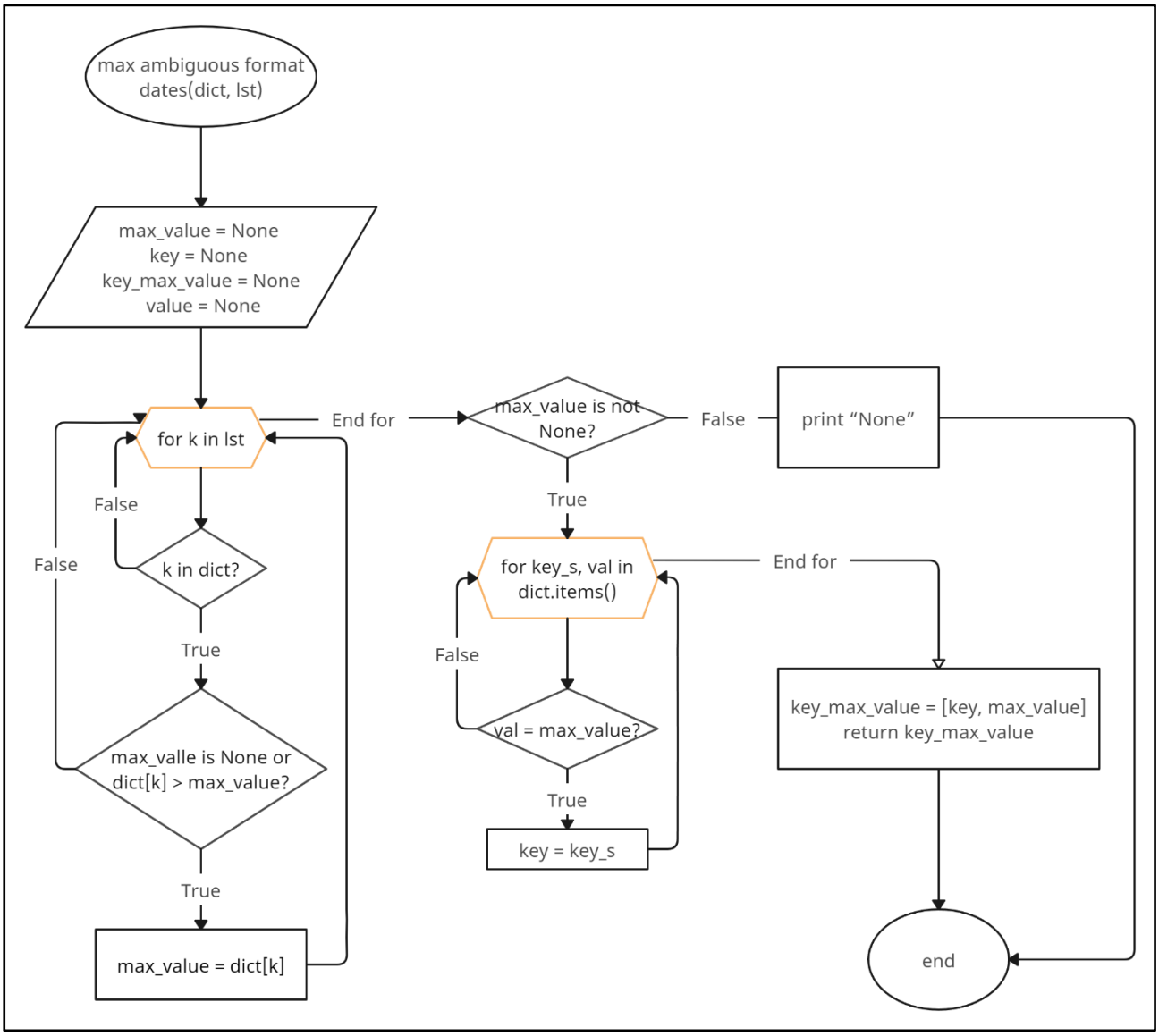

Рисунок 13 – Блок схема функції виявлення найпоширенішого неоднозначного формату

2.3 Проектування та реалізація програми

2.3.1 Проектування програми

Проектування програми базується на розробленій архітектурі програмного продукту і включатиме:

– імпорт бібліотек re, dateparser, time і logging, необхідних для реалізації;

– список date_patterns шаблонів дат регулярних виразів;
– основна функція, __get_date_from_string, пошук і отримання дати з рядка з можливістю повернення пар (формат, дата);
– функція тестування, __test_date, з вихідними даними з файлу;
– функція __max_ambiguous_format_date для вибору найбільш поширеного неоднозначного формату дати;
– функція, __write_date_to_file, запис результату роботи програми в файл.

2.3.2 Реалізація програми

Перша стадія реалізації програми у вигляді скороченого псевдокоду з описом для виділення логіки та наочності має вигляд як на рис. 14 – 16.

```
Функція отримання дати зі строки: #приймає параметри текст
    Ініціалізація змінних
    Для кожної позиції від 0 до довжини масиву патернів дат виконується наступне:
        Пошук всіх дат в тексті відповідно до патернів та занесення до загального списку
        Для кожної дати в загальному списку виконується наступне:
            Виконати препроцесінг дат до єдиного вигляду
            Перетворити на стандартний тип даних datetime
            Виконати перевірку валідності дати
            Якщо дати ще немає в результуючому масиві, то додати
            Додати 1 відповідно індексу знайденої дати до статистики
    Сортування дат
    Повернення результату роботи функції
```

Рисунок 14 – Псевдокод функції отримання дати зі строки

Функція тестування роботи головної функції: #приймає параметр файл
    Ініціалізація змінних
    Відкриття файлу для читання
    Запис до змінної вмісту файлу
    Передача змінної до головної функції
    Повернення результату роботи функції

Рисунок 15 – Псевдокод функції тестування

Функція виявлення найпоширенішого неоднозначного формату дати:
    Ініціалізація змінних
    Пошук відповідно до завчасно обраних індексів найпоширенішої дати в словнику статистики
    Повернення ключу та значення найпоширенішої дати

Рисунок 16 – Псевдокод функції виявлення найпоширенішого неоднозначного формату

Повний код програми (див. додаток И)

3 АНАЛІЗ РЕЗУЛЬТАТІВ

Для оцінки якості роботи програми та подальшого аналізу виконано тестування та за результатами тестування складено статистичну таблицю.

3.1 Тестування

Тест 1

Пошук та виявлення дат з великого фрагменту тексту, що написаний природною мовою. Вхідні дані загалом мають 3 дати(«27.03.2023», «14 квітня 2023», «07.04.2023», див. рис. 17).

```
date_text = """ Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі

Відповідно до наказу Нацагентства від 27.03.2023 року № 623-Е, у період із 12 по 14 квітня 2023 року, в Запорізькому національному університеті (ZNU, Zaporizhzhia National University) експертна група проводитиметься у віддаленому (дистанційному) режимі акредитаційну експертизу за спеціальністю 033 «Філософія» освітньої програми «Філософія» за третім (освітньо-науковим) рівнем вищої освіти.

ZNU Zaporizhzhia National University ЗНУ Запорізький національний університет факультет соціології (управління філософія Національне агентство забезпечення якості вищої освіти освітній процес акредитаційна експертиза Соціології та управління (https://www.znu.edu.ua/ukr/489/490/504)

Головні новини (https://www.znu.edu.ua/ukr/489/znu-front-news)

29 переглядів

07.04.2023 09:20"""
```

Рисунок 17 – Вхідні дані для тесту 1

Внаслідок пошуку та виявлення дат отримано результат, знайдено та виявлено 3 дати з 3-х та затрачено часу 1.6797008514404297 с (рис. 18).

```
{'14 квітня 2023': datetime.datetime(2023, 4, 14, 0, 0), '27.03.2023': datetime.datetime(2023, 3, 27, 0, 0), '07.04.2023': datetime.datetime(2023, 4, 7, 0, 0)}
statistics:  {2: 1, 12: 2}
[2, 1]
Час виконання програми:  1.6797008514404297 с
```

Рисунок 18 – Результат пошуку та виявлення дат

Тест 2

Пошук та виявлення дат з великого фрагменту тексту, що є списком приблизно 400 форматів дат. Вхідні дані мають 567 строк, кожна дата з нової строки. (див. додаток Ж).

Результат пошуку та виявлення дат(див. додаток З). Знайдено та виявлено 376 дати з 576 та затрачено часу 2.850278615951538 с.

Тест 3

Для читання дат з файлу формату json імпортована бібліотека jsonlines, програма дороблена додатковою функцією, що зчитує значення, яке містить дату, обраного поля. Скорочений опис-псевдокод (рис. 19).

```
Функція виявлення доти з json-файлу: #вхідні дані: файл та поле
    Відкриття файлу для читання
    Додання кожної знайденої дати як значення відповідного поля до загальної строки. Кожна дата з нового рядка
    Повернення результату роботи функції
```

Рисунок 19 – Псевдокод функції виявлення дати з json

Повний фрагмент коду функції у програмі(див. додаток К)

Пошук та виявлення дат з великого фрагменту тексту, що є вилученим фрагментом бази даних – 41430 рядків. Вхідні дані мають вигляд:(див. додаток И)

Результат пошуку та виявлення дат(див. додаток І). Знайдено та виявлено 41430 дати з 41430 та затрачено часу 2.850278615951538 с.

Статистика представлена в табл. 3.

Таблиця 3 – Результуюча статистика тестування програми

№ тесту	Кількість дат	Знайдено дат	Витрачено часу(с)
1	3	3	1.6797008514404297
2	576	376	2.850278615951538
3	41430	41430	24.454955339431763

3.2 Аналіз результатів

Після виконання трьох тестів на пошук і виявлення дат в довгому тексті, програма коректно відпрацювала в трьох випадках.

Виявлено помилки ERROR:root:KeyError('d') у другому тестуванні. З метою налагодження знайдені дати виведено у попередньому рядку рядка про помилку (рис. 20).

З цього випливає, що знайдені дати виду “dd mm yyyy”, де mm – рядок, не є датою, оскільки dd належить одному рядку, а mm yyyy – інший.

За результатом другого тесту виявлено на 200 дат менше від загальної кількості, через неповний збіг прогнозованих форматів дат в шаблоні з форматами дат вхідного тексту. Помилка вважається некритичною, адже список патернів форматів дат здатний до масштабування.

```
01
Februar 2023
ERROR:root:KeyError('d')

23
September 2023
ERROR:root:KeyError('d')

15
Juli 2023
ERROR:root:KeyError('d')

01
Dezember 2023
ERROR:root:KeyError('d')

01
September 2023
ERROR:root:KeyError('d')

03
April 2023
ERROR:root:KeyError('d')

23
Juli 2023
ERROR:root:KeyError('d')

04
Juni 2023
ERROR:root:KeyError('d')

04
November 2023
ERROR:root:KeyError('d')
```

Рисунок 20 – Помилки виявлення дат

ВИСНОВКИ

Мета роботи, реалізація метода виявлення у довгих текстах дат різних форматів одночасно із вибором формату, була досягнута.

Загалом було розглянуто проблему пошуку дат різних форматів, серед яких є неоднозначні, у довгому тексті. Було виявлено, що існуючих на даний момент актуальних рішень недостатньо через їхню неповноцінність. Тому необхідність пошуку та обробки тексту написаного природною мовою не виконується. Для вирішення цієї проблеми було розроблено програму, яка справляється з поставленим завданням без критичних помилок.

Для цього було використано стек технологій бібліотек Python, а саме: Dateparser, для виявлення дат та зведення до єдиного формату datetime; re, для визначення необхідного детального списку форматів дат для пошуку.

У рамках роботи були зроблені обмеження щодо кількості форматів дат внаслідок припущення їх популярності в потенціальному тексті написаним природною мовою.

Основні результати були зібрані до формату єдиної типової статистики.

Серед переваг можна виділити можна виділити точність пошуку дат, здатність до масштабування, зокрема розширення кількості доступних для пошуку форматів дат, та здатність до обробки великих масивів даних.

До недоліків можна віднести відносно тривалий час виконання програми.

Можливий розвиток програми з додаванням нових шаблонів дат та методів читання з великого масиву даних іншої структури.

Робота з реалізації метода виявлення дат має велику практичну значимість у контексті аналізу даних з часовою компонентною, що поширить ймовірність успіху в прогнозуванні майбутніх тенденцій через безпосередню залежність від часу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Leskovec J., Rajaraman A., Ullman J. Mining of Massive Datasets. 3rd Edition. Cambridge: Cambridge University Press, 2020. 603 p. URL : http://www.mmds.org/ (дата звернення: 27.02.2023).

2. Hyndman R. J., Athanasopoulos G. Forecasting: Principles and Practice. 3rd ed. Clayton : OTexts and Monash University, 2021. 432 p. 27.02.2023

3. Leskovec J., Rajaraman A., Ullman J. Mining of Massive Datasets. 3rd Edition. Cambridge: Cambridge University Press, 2020. 603 p. URL : http://www.mmds.org/ (дата звернення: 27.02.2023).

4. Hyndman R. J., Athanasopoulos G. Forecasting: Principles and Practice. 3rd ed. Clayton : OTexts and Monash University, 2021. 432 p.

5. David Mezzetti. How to Work With Dates in Python. Better Programming. URL : https://betterprogramming.pub/ (дата звернення: 27.03.2023).

6. Date format by country. Wikipedia. URL : https://en.wikipedia.org (дата звернення: 27.03.2023).

7. Dateparser – python parser for human readable dates. Dateparser Documentation. URL : https://dateparser.readthedocs.io/ (дата звернення: 27.03.2023).

8. Qddate – quick and dirty python parser dates what could be found during HTML scraping. Github. URL : https://github.com/ (дата звернення: 27.03.2023).

9. Dateutil - powerful extensions to datetime. Dateutil Documentation. URL : https://dateutil.readthedocs.io (дата звернення: 27.03.2023).

10. Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі. Запорізький національний університет. URL : https://www.znu.edu.ua/ (дата звернення: 27.02.2023).

11. Re – Regular expression operations. Python Documentation. URL : https://docs.python.org/ (дата звернення: 27.04.2023).

ДОДАТОК А

Формати дат

№	Формат	Приклад
0	yyyy-mm-dd	2023-02-01
1	mm dd, yyyy, де mm – рядок	July 1, 2023
2	dd mm yyyy, де mm – рядок	3 Jan 2023
3	yyyy-mm-dd, де mm – рядок	2023-May-01
4	yyyy/mm/dd	2023/11/03
5	yyyy/mm/dd, де mm – рядок	2023/Nov/01
6	yyyy.mm.dd	2023.02.17
7	dd, mm, yyyy, де mm – рядок	13, OCT, 2021
8	yyyy.mm.dd, де mm – рядок	01.Jan.2023
9	dd mm yyyy, де dd – цифра(и) дві букви, mm – рядок	1er juin 2023
10	mm dd yyyy, де mm – рядок, dd – цифри(и) та дві букви	May 1st, 2023
11	dd.mm.yyyy	13.01.2023
12	mm.dd.yyyy	06.14.2023
13	dd.mm.yyyy, де mm – рядок	13.Jan.2023

14	mm dd, yyyy, де mm – рядок	Februar 15, 2023
15	dd-mm-yyyy	13-04-2012
16	mm-dd-yyyy	05-18-2013
17	dd-mm-yyyy, де mm – рядок	15-Oct-2018
18	mm-dd-yyyy, де mm – рядок	Sep-01-2022
19	dd_mm_yyyy	22_10_2010
20	mm_dd_yyyy	10_24_2013
21	dd_mm_yyyy, де mm – рядок	24_Dec_2023
22	mm_dd_yyyy, де mm – рядок	Dec_23_2023
23	dd/mm/yyyy	08/04/2018
24	mm/dd/yyyy	04/28/2016
25	dd/mm/yyyy, де mm – рядок	5/May/2022
26	mm/dd/yyyy, де mm – рядок	May/5/2022
27	dd “de” mm “de” yyyy, де mm – рядок	21 de Octubre de 2014
28	dd mm yyyy, де mm – цифра та символ китайською	21 5月 2021
29	mm dd, yyyy, де mm – рядок китайською	五月 21, 2021
30	yyyymmdd, де yyyy – чотиризначне число та символ китайською, mm - цифра(и) та символ китайською, dd – цифра(и) та символ китайською	2022年6月12日

31	dd.mm.yy	8.9.2010
32	mm.dd.yy	11.14.2009
33	mm yyyy, де mm – рядок	June 2013
34	mm-yyyy, де mm – рядок	June-2013
35	dd mm yy, де dd – цифри(и) та дві букви , mm – рядок	11th May 22
36	dd mm yy, де mm – рядок	1 Aug 23

ДОДАТОК Б

Визначення дати з рядку тексту за допомогою Dateparser

```
import dateparser
import time

start_time = time.time()
dates = [
    "21 June 2018",
    "16 Februrari 2023",
    "16 Fevral 2023",
    "16 février 2023",
    "16 Φεβρουαρίου 2023",
    "1er mars 2023",
    "13 de enero de 2023",
    "五月 26, 2022",
    "2023年01月13日",
    "September 2023",
]
for date in dates:
    try:

        parsed_date = dateparser.parse(date)
        print(parsed_date, "     ", date)
    except Exception as ex:
        continue

end_time = time.time()
print("Час виконання програми: ", end_time - start_time, "c")
```

ДОДАТОК В

Пошук та визначення дати з великого фрагменту тексту за допомогою Dateparser

```
import dateparser
import time
import re

date_patterns = [
   (  #mm.dd.yyyy
      r"\d{1,2}\.\d{1,2}\.(?:19|20|21)\d{2}",
      lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split(".")))),
   ),
      (   #dd mm yyyy де mm - рядок
      r"\d{1,2}\s[^\W\d_]+\s(?:19|20|21)\d{2}(?![/-])",
      lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split(" ")))),
   ),
]
start_time = time.time()
date_text = """Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі

Відповідно до наказу Нацагентства від 27.03.2023 року № 623-Е, у період із 12 по 14 квітня 2023 року, в Запорізькому національному університеті (ZNU, Zaporizhzhia National University) експертна група проводитиметься у віддаленому (дистанційному) режимі акредитаційну експертизу за спеціальністю 033 «Філософія» освітньої програми «Філософія» за третім (освітньо-науковим) рівнем вищої освіти.
```

ZNU Zaporizhzhia National University ЗНУ Запорізький національний університет факультет соціології (управління філософія Національне агентство забезпечення якості вищої освіти освітній процес акредитаційна експертиза Соціології та управління (https://www.znu.edu.ua/ukr/489/490/504)

Головні новини (https://www.znu.edu.ua/ukr/489/znu-front-news)

29 переглядів

07.04.2023 09:20

"""

```
for pattern in date_patterns:
    matches = re.findall(pattern[0], date_text, re.IGNORECASE)
    for match in matches:
        parsed_date = dateparser.parse(match)
        print(parsed_date, "    " , match)

end_time = time.time()
print("Час виконання програми: ", end_time - start_time, "c")
```

ДОДАТОК Г

Пошук та визначення дати за допомогою qddate

```
import qddate
import time

start_time = time.time()
dates = [
    "21 June 2018",
    "16 Februrari 2023",
    "16 Fevral 2023",
    "16 février 2023",
    "16 Φεβρουαρίου 2023",
    "1er mars 2023",
    "13 de enero de 2023",
    "五月 26, 2022",
    "2023年01月13日",
    "September 2023",
]

parser = qddate.DateParser()

for i in dates:
    try:
        print(parser.parse(i))
    except Exception as ex:
        print(ex)
```

```
end_time = time.time()
print("Час виконання програми: ", end_time - start_time, "c")
```

ДОДАТОК Д

Пошук та визначення дати за допомогою qddate

```
import qddate
import time
import re

start_time = time.time()

date_patterns = [
   (  #mm.dd.yyyy
      r"\d{1,2}\.\d{1,2}\.(?:19|20|21)\d{2}",
      lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split(".")))),
   ),
      (   #dd mm yyyy де mm - рядок
      r"\d{1,2}\s[^\W\d_]+\s(?:19|20|21)\d{2}(?![/-])",
      lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split(" ")))),
   ),
]

date_text = """Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі

Відповідно до наказу Нацагентства від 27.03.2023 року № 623-Е, у період із 12 по 14 квітня 2023 року, в Запорізькому національному університеті (ZNU, Zaporizhzhia National University) експертна група проводитиметься у віддаленому (дистанційному) режимі акредитаційну експертизу за спеціальністю 033 «Філософія» освітньої програми «Філософія» за третім (освітньо-науковим) рівнем вищої освіти.
```

ZNU Zaporizhzhia National University ЗНУ Запорізький національний університет факультет соціології (управління філософія Національне агентство забезпечення якості вищої освіти освітній процес акредитаційна експертиза Соціології та управління (https://www.znu.edu.ua/ukr/489/490/504)

Головні новини (https://www.znu.edu.ua/ukr/489/znu-front-news)

29 переглядів

07.04.2023 09:20

"""

```
parser = qddate.DateParser()

for patern in date_patterns:
    matches = re.findall(patern[0], date_text, re.IGNORECASE)
    for match in matches:
        print(parser.parse(match), "    ", match)

end_time = time.time()
print("Час виконання програми: ", end_time - start_time, "c")
```

ДОДАТОК Е

Визначення дати за допомогою dateutil

```
from dateutil.parser import parse
import time

start_time = time.time()
dates = [
    "21 June 2018",
    "16 Februrari 2023",
    "16 Fevral 2023",
    "16 février 2023",
    "16 Φεβρουαρίου 2023",
    "1er mars 2023",
    "13 de enero de 2023",
    "五月 26, 2022",
    "2023年01月13日",
    "September 2023",
]

for i in dates:
    try:
        print(parse(i))
    except Exception as ex:
        print(ex)

end_time = time.time()
print("Час виконання програми: ", end_time - start_time, "c")
```

ДОДАТОК Ж

Пошук та визначення дати з великого фрагменту тексту за допомогою dateutil

```
from dateutil.parser import parse
import time
import re

start_time = time.time()

date_patterns = [
   (  #mm.dd.yyyy
      r"\d{1,2}\.\d{1,2}\.(?:19|20|21)\d{2}",
      lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split(".")))),
   ),
      (   #dd mm yyyy де mm - рядок
      r"\d{1,2}\s[^\W\d_]+\s(?:19|20|21)\d{2}(?![/-])",
      lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split(" ")))),
   ),
]

date_text = """Експертна група Національного агентства із забезпечення якості вищої освіти запрошує долучитися до відкритої зустрічі

Відповідно до наказу Нацагентства від 27.03.2023 року № 623-Е, у період із 12 по 14 квітня 2023 року, в Запорізькому національному університеті (ZNU, Zaporizhzhia National University) експертна група проводитиметься у віддаленому (дистанційному) режимі акредитаційну експертизу за
```

спеціальністю 033 «Філософія» освітньої програми «Філософія» за третім (освітньо-науковим) рівнем вищої освіти.

ZNU Zaporizhzhia National University ЗНУ Запорізький національний університет факультет соціології (управління філософія Національне агентство забезпечення якості вищої освіти освітній процес акредитаційна експертиза Соціології та управління (https://www.znu.edu.ua/ukr/489/490/504)

Головні новини (https://www.znu.edu.ua/ukr/489/znu-front-news)

29 переглядів

07.04.2023 09:20

"""

```
for patern in date_patterns:
    matches = re.findall(patern[0], date_text, re.IGNORECASE)
    for match in matches:
        try:
            print(parse(match), "    ", match)
        except Exception as ex:
            print(ex)
end_time = time.time()
print("Час виконання програми: ", end_time - start_time, "c")
```

ДОДАТОК И

Програма пошуку та виявлення дат

```
import re
import dateparser
import logging
import time
import jsonlines

date_patterns = [

    (# Regex for yyyy-mm-dd
        r"(?:19|20|21)\d{2}-\d{1,2}-\d{1,2}",
        None,
        None,
    ),
    (   # Regex for mm dd, yyyy where mm is a string
        r"[^\W\d_]+\s\d{1,2}(?:,)\s(?:19|20|21)\d{2}",
        lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split(" ")))),
        None
    ),
    (   # Regex for dd mm yyyy where mm is a string
        r"\d{1,2}\s[^\W\d_]+\s(?:19|20|21)\d{2}(?![/-])",
        lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split(" ")))),
        None,
    ),
    (  # Regex for yyyy-mm-dd where mm is a string
        r"(?:19|20|21)\d{2}-[^\W\d_]+-\d{1,2}",
```

```
        lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split("-")))),
        None,
    ),
    (  # Regex for yyyy/mm/dd
        r"(?:19|20|21)\d{2}\/\d{1,2}\/\d{1,2}",
        lambda s: "{y}-{m}-{d}".format(**dict(zip(("y", "m", "d"), s.split("/")))),
        None,
    ),
    (  # Regex for yyyy/mm/dd where mm is a string
        r"(?:19|20|21)\d{2}\/[^\W\d_]+\/\d{1,2}",
        lambda s: "{m} {d} {y}".format(**dict(zip(("d", "m", "y"), s.split("/")))),
        None,
    ),
    (  # Regex for yyyy.mm.dd
        r"(?:19|20|21)\d{2}\.\d{1,2}\.\d{1,2}",
        lambda s: "{y}-{m}-{d}".format(**dict(zip(("y", "m", "d"), s.split(".")))),
        None,
    ),
    (  # Regex for yyyy.mm.dd where mm is a string
        r"(?:19|20|21)\d{2}\.[^\W\d_]+\.\d{1,2}",
        lambda s: "{d} {m} {y}".format(**dict(zip(("y", "m", "d"), s.split(".")))),
        None,
    ),
    (  # Regex for dates like 13 OCT 2021
        r"\d{1,2}[., ]+[^\W\d_]+[., ]+(?:19|20|21)\d{2}",
        None,
        None,
    ),
    # 11th May 2022
    (
```

```
    r"\d{1,2}\w{2}[., ]+[^\W\d_]+[., ]+(?:19|20|21)\d{2}",
    None,
    None,
),
(  # Regex for dates like December 2nd 2020
    r"[^\W\d_]+[, ]+\d{1,2}[^\W\d_]{1,2}[, ]+(?:19|20|21)\d{2}",
    None,
    None,
),
(  # Regex for dates like December 31, 2020
    r"[^\W\d_]+[, ]+\d{1,2}[, ]+(?:19|20|21)\d{2}",
    None,
    None,
),
(  # dd.mm.yyyy
    r"\d{1,2}\.\d{1,2}\.(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("d", "m", "y"), s.split(".")))),
    None,
),
(  # mm.dd.yyyy
    r"\d{1,2}\.\d{1,2}\.(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split(".")))),
    lambda d,s: s==d.strftime("%m.%d.%Y"),
),
(  # dd.mm.yyyy where mm is a string
    r"\d{1,2}\.[^\W\d_]+\.(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("d", "m", "y"), s.split(".")))),
    lambda d,s: s==d.strftime("%d.%m.%Y"),
),
(  # mm.dd.yyyy where mm is a string
```

```
    r"[^\W\d_]+\.\d{1,2}\.(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("m", "d", "y"), s.split(".")))),
    None,
),
(  # dd-mm-yyyy,
    r"\d{1,2}-\d{1,2}-(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("d", "m", "y"), s.split("-")))),
    None,
),
(  # mm-dd-yyyy,
    r"\d{1,2}-\d{1,2}-(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split("-")))),
    None,
),
(  # dd-mm-yyyy,  where mm is a string
    r"\d{1,2}-[^\W\d_]+-(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("d", "m", "y"), s.split("-")))),
    None,
),
(  # mm-dd-yyyy,  where mm is a string
    r"[^\W\d_]+-\d{1,2}-(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("m", "d", "y"), s.split("-")))),
    None,
),
(  # dd_mm_yyyy,
    r"\d{1,2}_\d{1,2}_(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("d", "m", "y"), s.split("_")))),
    None,
),
(  # mm_dd_yyyy,
```

```
    r"\d{1,2}_\d{1,2}_(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split("_")))),
    None,
),
(  # dd_mm_yyyy,  where mm is a string
    r"\d{1,2}_[^\W\d_]+_(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("d", "m", "y"), s.split("_")))),
    # lambda d,s: s==d.strftime("%d_%m_%Y"),
    None,
),
(  # mm_dd_yyyy,  where mm is a string
    r"[^\W\d_]+_\d{1,2}_(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("m", "d", "y"), s.split("_")))),
    None,
),
(  # dd/mm/yyyy
    r"\d{1,2}\/\d{1,2}\/(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("d", "m", "y"), s.split("/")))),
    None,
),
(  # mm/dd/yyyy
    r"\d{1,2}\/\d{1,2}\/(?:19|20|21)\d{2}",
    lambda s: "{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"), s.split("/")))),
    None,
),
(  # dd/mm/yyyy,  where mm is a string
    r"\d{2}\/[^\W\d_]+\/(?:19|20|21)\d{2}",
    lambda s: "{m} {d} {y}".format(**dict(zip(("m", "d", "y"), s.split("/")))),
    None,
),
```

```
        (  # mm/dd/yyyy,  where mm is a string
           r"[^\W\d_]+\/\d{1,2}\/(?:19|20|21)\d{2}",
           lambda s: "{m} {d} {y}".format(**dict(zip(("d", "m", "y"), s.split("/")))),
           None,
        ),
        (  # "21 de Octubre de 2014"
           r"(\d{1,2}[, ]+de[, ]+[^\W\d_]+[, ]+de[, ]+(?:19|20|21)\d{2})",
           None,
           None,
        ),
        (  # chinese date: 21 5月 2021
           r"(\d{1,2}[, ]+[0-9][\u4e00-\u9fa5]+[, ]+(?:19|20|21)\d{2})",
           None,
           None,
        ),
        (  # chinese date: 五月 21, 2021
           r"([\u4e00-\u9fa5][., ]*[\u4e00-\u9fa5][., ]*\d{1,2}[., ]+(?:19|20|21)\d{2})",
           None,
           None,
        ),
        (  # chinese date: 2022年6月12日
           r"((?:19|20|21)\d{2}[\u4e00-\u9fa5][., ]*\d{1,2}[., ]*[\u4e00-\u9fa5][., ]*\d{1,2}[., ]*[\u4e00-\u9fa5])",
           None,
           None,
        ),
        #
        # below are patterns of incomplete dates
        (  # dd.mm.yy
```

```
            r"\d{1,2}\.\d{1,2}\.[012345]\d{1}(?![0-9])",
            lambda s: "20{y}-{m}-{d}".format(**dict(zip(("d", "m", "y"),
s.split(".")))),
            None,
        ),
        (  # dd.mm.yy
            r"\d{1,2}\.\d{1,2}\.[6789]\d{1}(?![0-9])",
            lambda s: "19{y}-{m}-{d}".format(**dict(zip(("d", "m", "y"),
s.split(".")))),
            None,
        ),
        (  # mm.dd.yy
            r"\d{1,2}\.\d{1,2}\.[01234]\d{1}(?![0-9])",
            lambda s: "20{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"),
s.split(".")))),
            None,
        ),
        (  # mm.dd.yy
            r"\d{1,2}\.\d{1,2}\.[6789]\d{1}(?![0-9])",
            lambda s: "19{y}-{m}-{d}".format(**dict(zip(("m", "d", "y"),
s.split(".")))),
            None,
        ),
        # May 2022
        (
            r"[^\W\d_]+[, ]+(?:19|20|21)\d{2}",
            lambda x: f"1 {x}",
            None,
        ),
        (  # Regex to match dates like Dec-2021
```

```
        r"[^\W\d_]+-(?:19|20|21)\d{2}",
        lambda s: "1 {m} {y}".format(**dict(zip(("m", "y"), s.split("-")))),
        None,
    ),
    # 11th May 22
    (r"\d{1,2}\w{2}[., ]+[^\W\d_]+[., ]+\d{2}(?![0-9])", None, None),

    (   # 1 Aug 23
        r"\d{1,2}[., ]+[^\W\d_]+[., ]+\d{2}(?![0-9])",
        lambda s: "{m} {d} {y}".format(**dict(zip(("y", "m", "d"), s.split(" ")))),
        None
    ),


]


# Get date from string
def __get_date_from_string(txt: str, statistics={}):
    detected_dates = dict()
    response = dict()
    for i in range(0, len(date_patterns)):
        pattern, preprocess, date_validator = date_patterns[i]
        matches = re.findall(pattern, txt, re.IGNORECASE)
        for match in matches:
            if match not in "".join(detected_dates.keys()):

                try:
                    match_preprocessed = preprocess(match) if preprocess else match
                    date_matched = dateparser.parse(
```

```
                    match_preprocessed,
                    settings={"RETURN_AS_TIMEZONE_AWARE": False},
                )
                if date_matched:
                    if not date_validator or date_validator(date_matched, match):
                        if match not in detected_dates:
                            detected_dates[match]=dict()
                        detected_dates[match][i] = date_matched
                        statistics[i] = statistics.get(i, 0) + 1
            except Exception as ex:
                logging.error(repr(ex))


    for match in detected_dates:
        variants = [(statistics[i], i) for i in detected_dates[match].keys()]
        sorted_variants = sorted(variants, key=lambda x: -x[0])
        best_variant = sorted_variants[0][1]
        response[match] = detected_dates[match][best_variant]

    return response


# Date format tests from a file
def __test_date(txtfile: str):
        statistics = {}
        try:
            with open(txtfile, 'r') as f:
                contents = f.read()
                test = __get_date_from_string(contents, statistics=statistics)
                print(test)
```

```
            __write_date_to_file("aaa.txt", str(test))
            return statistics
        except FileNotFoundError:
            print("File not found")
        except IOError:
            print("Failed to read file")



# Writing dates to a file
def __write_date_to_file(txtfile: str, dict):
        try:
            with open(txtfile, 'a') as f:
                f.write(dict)
        except FileNotFoundError:
            print("File not found")
        except IOError:
            print("Failed to write to file")



# Selecting the most common ambiguous date format
def __max_ambiguous_format_dates(dict, lst):

    max_value = None
    key = None
    key_max_value = None
    for k in lst:
        if k in dict:
            if max_value is None or dict[k] > max_value:
                max_value = dict[k]
```

```
    if max_value is not None:
        for key_s, val in dict.items():
            if val == max_value:
                key = key_s

        key_max_value = [key, max_value]

        return key_max_value
    else:
        print("None")



# Get date from json
def __get_date_from_json(txtfile: str, field:   str):
    dates = ""
    with jsonlines.open(txtfile) as reader:
        for line in reader:
            publication_date = line[field]
            dates = dates + "\n" + str(publication_date)
    return dates

def main():

    start_time = time.time()

    # jobs.jl
    dates = __get_date_from_json("jobs.jl", "publication_date")
    statistics = {}
    print(__get_date_from_string(dates, statistics=statistics))
    print("statistics: ",statistics)
```

```
    r = [2, 9, 28, 1,0,31]
    print(__max_ambiguous_format_dates(statistics, r))

    end_time = time.time()
    print("Час виконання програми: ", end_time - start_time, "c")

if __name__ == "__main__":
    main()
```

ДОДАТОК К

Вхідні дані для тесту 2

1st March 2023
01-08-2023
1er mars 2023 (French format)
2023-06-01
Februar 2023
2023-01-12
3 Jan 2023
11 Jan 2023
2023/09/01
23/02/16

…

November 2023 (German format)
01/Jul/2023
1 de marzo de 2023 (Spanish format)
16 februāris 2023 (Latvian format)
2023.07.01
01/08/23
16 februar 2023 (Danish format)
? 17 février 23
2023年01月01日
1 de abril de 2023 (Spanish format)
1st Dec 2023
16 februari 2023 (Swedish format)
12-Dec-23
February 1, 2023

ДОДАТОК Л

Пошук та виявлення дати зі списку

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

ERROR:root:KeyError('d')

{'2023-06-01': datetime.datetime(2023, 6, 1, 0, 0), '2023-01-12': datetime.datetime(2023, 1, 12, 0, 0), '2023-09-01': datetime.datetime(2023, 9, 1, 0, 0), '2023-01-11': datetime.datetime(2023, 1, 11, 0, 0), '2023-01-01': datetime.datetime(2023, 1, 1, 0, 0), '2023-07-01': datetime.datetime(2023, 7, 1, 0, 0), '2023-01-02': datetime.datetime(2023, 1, 2, 0, 0), '2023-02-17': datetime.datetime(2023, 2, 17, 0, 0), '2023-05-01': datetime.datetime(2023, 5, 1, 0, 0), '2023-01-03': datetime.datetime(2023, 1, 3, 0, 0), '2023-02-15': datetime.datetime(2023, 2, 15, 0, 0), '2023-03-01': datetime.datetime(2023, 3, 1, 0, 0), '2023-12-01': datetime.datetime(2023, 12, 1, 0, 0), '2023-08-01': datetime.datetime(2023, 8, 1, 0, 0), '2023-04-01': datetime.datetime(2023, 4, 1, 0, 0), '2023-01-13': datetime.datetime(2023, 1, 13, 0, 0), '2023-10-01': datetime.datetime(2023, 10, 1, 0, 0), '2023-11-01': datetime.datetime(2023, 11, 1, 0, 0),

…

statistics: {0: 22, 1: 26, 2: 75, 3: 20, 4: 6, 5: 20, 6: 6, 8: 9, 9: 38, 10: 21, 12: 29, 16: 6, 18: 6, 19: 2, 24: 25, 26: 7, 28: 18, 32: 12, 36: 10, 38: 8, 39: 10}

[2, 75]

Час виконання програми: 2.850278615951538 с

ДОДАТОК М

Вхідні дані до тесту 3, файл jobs.jl

{"company_name": "Pets Deli", "company_url": "", "publication_date": "05.04.2022", "job_title": "Werkstudent IT Administration / IT Support (all genders) ", "location_data": "Berlin", "url_to_the_job_posting": "https://wzw1.de/jobklick/?id=000B582A-813E-4538-8065-A59A19082A74", "founding_year": "", "number_of_employees": ""}

….

{"company_name": "Mercedes-Benz Mobility AG", "company_url": "", "publication_date": "24.03.2022", "job_title": "Risk Analyst - Experte von Rating-Themen bei Risk Instruments (w/m/d) ", "location_data": "Stuttgart", "url_to_the_job_posting": "https://wzw1.de/jobklick/?id=8786076B-F95E-4A5E-B3BF-59529F6F0355", "founding_year": "", "number_of_employees": ""}

ДОДАТОК Л

Пошук та виявлення дат з файлу jobs.jl

{'2022-05-04': datetime.datetime(2022, 5, 4, 0, 0), '2022-04-04': datetime.datetime(2022, 4, 4, 0, 0), '2022-03-04': datetime.datetime(2022, 3, 4, 0, 0), '2022-02-04': datetime.datetime(2022, 2, 4, 0, 0), '2022-01-04': datetime.datetime(2022, 1, 4, 0, 0), '2022-03-31': datetime.datetime(2022, 3, 31, 0, 0), '2022-03-30': datetime.datetime(2022, 3, 30, 0, 0), '2022-03-29': datetime.datetime(2022, 3, 29, 0, 0), '2022-03-28': datetime.datetime(2022, 3, 28, 0, 0), '2022-03-27': datetime.datetime(2022, 3, 27, 0, 0), '2022-03-26': datetime.datetime(2022, 3, 26, 0, 0), '2022-03-25': datetime.datetime(2022, 3, 25, 0, 0), '2022-03-24': datetime.datetime(2022, 3, 24, 0, 0)}

…

statistics: {0: 41430}

Час виконання програми: 24.454955339431763 с