Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №9
дисциплины «Программирование на python»

Выполнил:
Кожуховский Виктор Андреевич
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств вычислительной
техники и автоматизированных систем
», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Работа со словарями в языке Python

Цель работы: приобретение навыков по работе со словарями при написании программ с помощью языка программирования Python версии 3.x.

Методика и порядок выполнения работы

1. Изучил теоретический материал работы.

2.Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

3. Выполнил клонирование созданного репозитория.

4. Дополнил файл .gitignore необходимыми правилами для работы с IDE PyCharm.

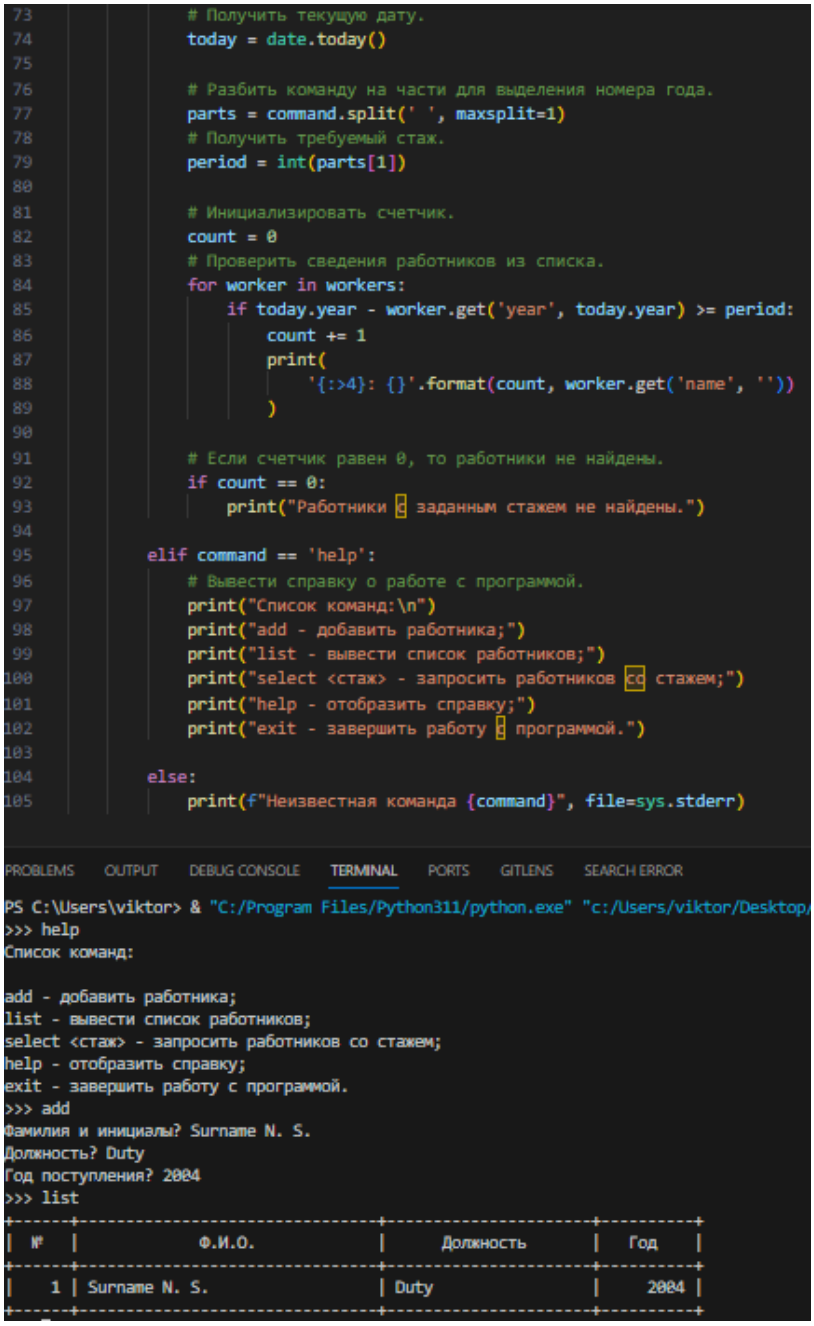
5. Организовал свой репозиторий в соответствие с моделью ветвления git-flow.

6. Создал проект в папке репозитория.

7. Проработал примеры лабораторной работы. Создал для каждого примера отдельный модуль языка Python. Зафиксировал изменения в репозитории.

```python
Click here to ask Blackbox to help you code faster
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys
from datetime import date


if __name__ == '__main__':
    # Список работников.
    workers = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о работнике.
            name = input("Фамилия и инициалы? ")
            post = input("Должность? ")
            year = int(input("Год поступления? "))

            # Создать словарь.
            worker = {
            'name': name,
            'post': post,
            'year': year,
            }

            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            # Заголовок таблицы.
            line = '+-{}-+-{}-+-{}-+-{}-+'.format(
                '-' * 4,
                '-' * 30,
                '-' * 20,
                '-' * 8
            )
            print(line)
            print(
                '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
                )
            )
            print(line)

            # Вывести данные о всех сотрудниках.
            for idx, worker in enumerate(workers, 1):
                print(
                    '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    worker.get('name', ''),
                    worker.get('post', ''),
                    worker.get('year', 0)
                    )
                )

            print(line)

        elif command.startswith('select '):
```

Рисунок 1. Код примера



```
            # Получить текущую дату.
            today = date.today()

            # Разбить команду на части для выделения номера года.
            parts = command.split(' ', maxsplit=1)
            # Получить требуемый стаж.
            period = int(parts[1])

            # Инициализировать счетчик.
            count = 0
            # Проверить сведения работников из списка.
            for worker in workers:
                if today.year - worker.get('year', today.year) >= period:
                    count += 1
                    print(
                        '{:>4}: {}'.format(count, worker.get('name', ''))
                    )

            # Если счетчик равен 0, то работники не найдены.
            if count == 0:
                print("Работники с заданным стажем не найдены.")

        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - вывести список работников;")
            print("select <стаж> - запросить работников со стажем;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")

        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS  SEARCH ERROR

PS C:\Users\viktor> & "C:/Program Files/Python311/python.exe" "c:/Users/viktor/Desktop/
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Surname N. S.
Должность? Duty
Год поступления? 2004
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | Surname N. S.                    | Duty                 |     2004 |
+------+----------------------------------+----------------------+----------+
```

Рисунок 2. Вторая часть кода примера и вывод

9. Решите задачу: создайте словарь, связав его с переменной school, и наполните данными, которые бы отражали количество учащихся в разных классах (1а, 1б, 2б, 6а, 7в и т. п.). Внесите изменения в словарь согласно следующему: а) в одном из классов изменилось количество учащихся, б) в школе появился новый класс, с) в школе был расформирован (удален) другой класс. Вычислите общее количество учащихся в школе.

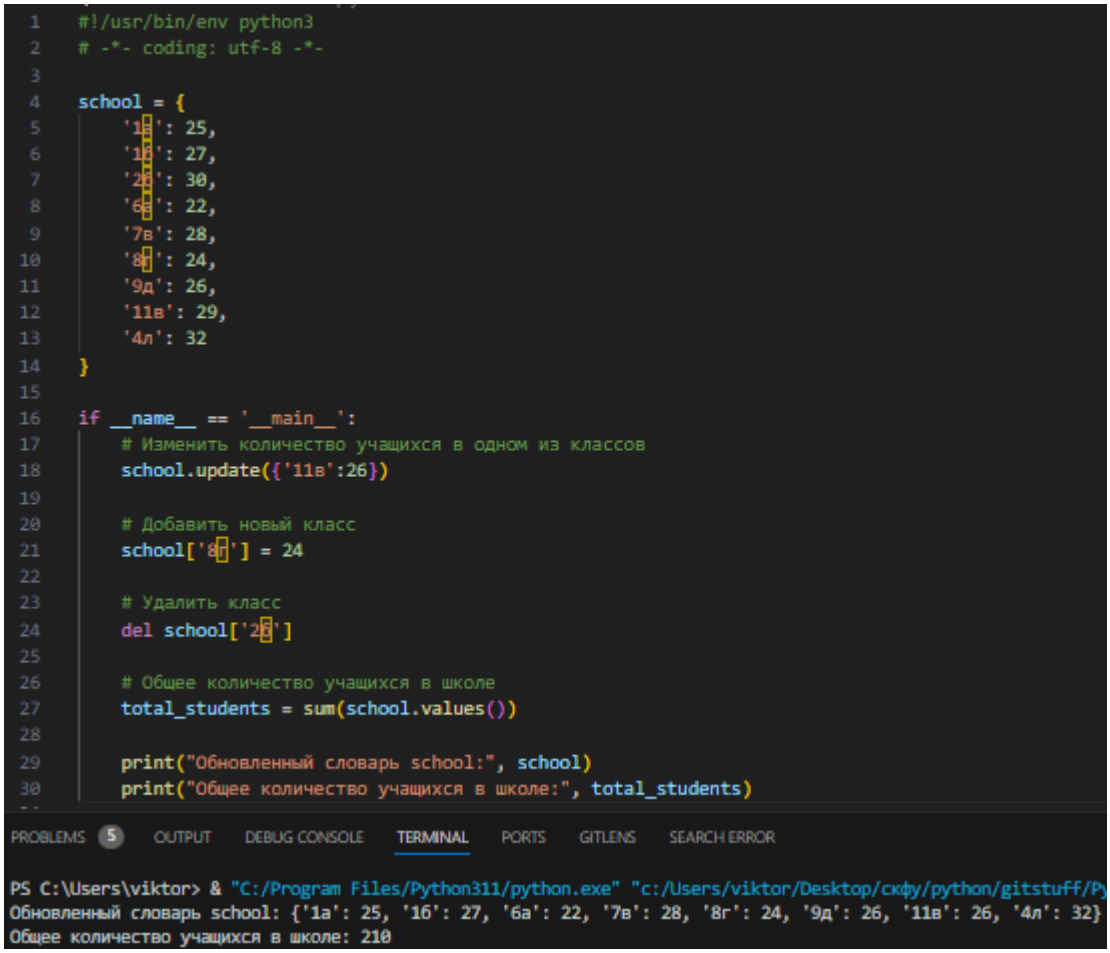


Рисунок 3. Код для решения задачи и его вывод

10. Зафиксировал сделанные изменения в репозитории.

11. Решите задачу: создайте словарь, где ключами являются числа, а значениями – строки. Примените к нему метод items(), с помощью полученного объекта dict_items создайте новый словарь, "обратный" исходному, т. е. ключами являются строки, а значениями – числа.

Рисунок 4. Код для решения задачи и его вывод

12. Зафиксировал сделанные изменения в репозитории.

13. Приведите в отчете скриншоты работы программ и UML-диаграммы деятельности решения индивидуального задания.

Использовать словарь, содержащий следующие ключи: фамилия, имя; знак Зодиака; дата рождения (массив из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по знакам Зодиака; вывод на экран информации о людях, родившихся в месяц, значение которого введено с клавиатуры; если таких нет, выдать на дисплей соответствующее сообщение.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys
from datetime import date


if __name__ == '__main__':
    # Список людей со строками для демонстрации.
    people = [
        {'name': 'Иван', 'surname': 'Иванов', 'date_of_birth': [
            1, 2, 1990], 'zodiac_sign': 'Овен'},
        {'name': 'Мария', 'surname': 'Петрова', 'date_of_birth': [
            15, 7, 1985], 'zodiac_sign': 'Рак'},
        {'name': 'Алексей', 'surname': 'Сидоров', 'date_of_birth': [
            10, 11, 2000], 'zodiac_sign': 'Скорпион'},
        {'name': 'Елена', 'surname': 'Козлова', 'date_of_birth': [
            3, 4, 1995], 'zodiac_sign': 'Овен'},
        {'name': 'Сергей', 'surname': 'Игнатов', 'date_of_birth': [
            22, 9, 1982], 'zodiac_sign': 'Дева'},
        {'name': 'Анна', 'surname': 'Кузнецова', 'date_of_birth': [
            5, 12, 1988], 'zodiac_sign': 'Стрелец'}
    ]

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о человеке.
            name = input("Фамилия: ")
            surname = input("Имя: ")
            date_of_birth = list(map(int, input(
                "Введите дату рождения (в формате ДД.ММ.ГГГГ через точку): ").split('.')))
            zodiac_sign = input("Знак зодиака: ")

            # Создать словарь.
            person = {
                'name': name,
                'surname': surname,
                'date_of_birth': date_of_birth,
                'zodiac_sign': zodiac_sign
            }

            # Добавить словарь в список.
            people.append(person)
            # Отсортировать список по знакам Зодиака.
            people.sort(key=lambda item: item.get('zodiac_sign', ''))

        elif command == 'list':
            # Заголовок таблицы.
            line = '+-{}-+-{}-+-{}-+-{}-+-{}-+'.format(
                '-' * 4,
                '-' * 20,
                '-' * 20,
                '-' * 15,
                '-' * 13
            )
            print(line)
            print(
                '| {:^4} | {:^20} | {:^20} | {:^15} | {:^12} |'.format(
                    "№",
                    "Имя",
                    "Фамилия",
                    "Знак Зодиака",
                    "Дата рождения"
                )
```

Рисунок 5. Первая часть кода индивидуального задания

```
    )
    print(line)

    # Вывести данные о всех сотрудниках.
    for idx, person in enumerate(people, 1):
        # Преобразование даты рождения в строку для вывода
        birth_date_str = '.'.join(
            map(str, person.get('date_of_birth', '')))
        print(
            '| {:^4} | {:<20} | {:<20} | {:<15} | {:<13} |'.format(
                idx,
                person.get('name', ''),
                person.get('surname', ''),
                person.get('zodiac_sign', ''),
                birth_date_str
            )
        )

    print(line)

elif command.startswith('select '):
    # Разбить команду на части для выделения номера месяца.
    parts = command.split(' ', maxsplit=1)
    # Получить требуемый месяц.
    month = int(parts[1])

    # Инициализировать счетчик.
    count = 0
    # Проверить данные о людях из списка.
    for person in people:
        if person.get('date_of_birth', [])[1] == month:
            count += 1
            print(
                '{:>4}: {} {}'.format(count, person.get(
                    'name', ''), person.get('surname', ''))
            )

    # Если счетчик равен 0, то люди не найдены.
    if count == 0:
        print("Люди, родившиеся в указанном месяце, не найдены.")

elif command == 'help':
    # Вывести справку о работе с программой.
    print("Список команд:\n")
    print("add - добавить человека;")
    print("list - вывести список людей;")
    print(
        "select <месяц> - вывод на экран информации о людях, родившихся в указанный месяц (цифра)")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")

else:
    print(f"Неизвестная команда {command}", file=sys.stderr)
```

Рисунок 6. Вторая часть кода индивидуального задания

```
>>> list
+------+----------------------+----------------------+----------------+---------------+
|  №   |         Имя          |       Фамилия        |  Знак Зодиака  | Дата рождения |
+------+----------------------+----------------------+----------------+---------------+
|  1   | Иван                 | Иванов               | Овен           | 1.2.1990      |
|  2   | Мария                | Петрова              | Рак            | 15.7.1985     |
|  3   | Алексей              | Сидоров              | Скорпион       | 10.11.2000    |
|  4   | Елена                | Козлова              | Овен           | 3.4.1995      |
|  5   | Сергей               | Игнатов              | Дева           | 22.9.1982     |
|  6   | Анна                 | Кузнецова            | Стрелец        | 5.12.1988     |
+------+----------------------+----------------------+----------------+---------------+
>>> add
Фамилия: Игнатов
Имя: Олег
Введите дату рождения (в формате ДД.ММ.ГГГГ через точку): 2.2.2000
Знак зодиака: Овен
>>> list
+------+----------------------+----------------------+----------------+---------------+
|  №   |         Имя          |       Фамилия        |  Знак Зодиака  | Дата рождения |
+------+----------------------+----------------------+----------------+---------------+
|  1   | Сергей               | Игнатов              | Дева           | 22.9.1982     |
|  2   | Иван                 | Иванов               | Овен           | 1.2.1990      |
|  3   | Елена                | Козлова              | Овен           | 3.4.1995      |
|  4   | Игнатов              | Олег                 | Овен           | 2.2.2000      |
|  5   | Мария                | Петрова              | Рак            | 15.7.1985     |
|  6   | Алексей              | Сидоров              | Скорпион       | 10.11.2000    |
|  7   | Анна                 | Кузнецова            | Стрелец        | 5.12.1988     |
+------+----------------------+----------------------+----------------+---------------+
>>> select 2
   1: Иван Иванов
   2: Игнатов Олег
```

Рисунок 7. Вывод программы индивидуального задания

14. Зафиксировал сделанные изменения в репозитории.

15. Добавил отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксировал изменения.

16. Выполнил слияние ветки для разработки с веткой main / master.

17. Отправил сделанные изменения на сервер GitHub.

Вопросы для защиты работы

1. Что такое словари в языке Python?

Словарь (dict) представляет собой структуру данных (которая ещё называется ассоциативный массив), предназначенную для хранения произвольных объектов с доступом по ключу.

2. Может ли функция len() быть использована при работе со словарями?

Может.

3. Какие методы обхода словарей Вам известны?

Элементы словаря перебираются в цикле for также, как элементы других сложных объектов. Однако "по-умолчанию" извлекаются только ключи:

```
nums
{1: 'one', 2: 'two', 3: 'three'}
for i in nums:
        print(i)
1
2
3
```

Но по ключам всегда можно получить значения:

```
for i in nums:
         print(nums[i])
one
two
three
```

С другой стороны у словаря как класса есть метод items(), который создает особую структуру, состоящую из кортежей. Каждый кортеж включает ключ и значение:

```
n = nums.items()
>>> n
dict_items([(1, 'one'), (2, 'two'), (3, 'three')])
```

В цикле for можно распаковывать кортежи, таким образом сразу извлекая как ключ, так и его значение:

```
for key, value in nums.items():
... print(key, 'is', value)
1 is one
2 is two
3 is three
```

Методы словаря keys() и values() позволяют получить отдельно перечни ключей и значений. Так что если, например, надо перебрать только значения или только ключи, лучше воспользоваться одним из этих методов:

```
v_nums = []
>>> for v in nums.values():
... v_nums.append(v)
>>> v_nums
['one', 'two', 'three']
```

То же самое можно сделать с помощью списковых включений:

```
v_nums = [v for v in nums.values]
```

4. Какими способами можно получить значения из словаря по ключу?

Метод get() позволяет получить элемент по его ключу:

```
>>> nums.get(1)
'one'
```

Использование квадратных скобок:

```
num = nums[1]
```

5. Какими способами можно установить значение в словаре по ключу?

С помощью setdefault() можно добавить элемент в словарь:

```
>>> nums.setdefault(4, 'four')
'four'
>>> nums
{3: 'three', 4: 'four'}
```

С помощью update() можно добавить в словарь другой словарь:

```
>>> nums.update({6: 'six', 7: 'seven'})
>>> nums
{3: 'three', 4: 'four', 6: 'six', 7: 'seven'}
```

Использование квадратных скобок:

num[1] = ‘Two’

6. Что такое словарь включений?

Словарь включений аналогичен списковым включениям, за исключением того, что он создаёт объект словаря вместо списка.

7. Самостоятельно изучите возможности функции zip() приведите примеры ее использования.

Функция `zip()` в Python используется для объединения элементов из двух или более итерируемых объектов в кортежи.

Примеры:

1. Основное использование:

```
names = ['Иван', 'Мария', 'Алексей']
ages = [25, 30, 22]
# Объединение итерируемых объектов в кортежи
zipped_data = zip(names, ages)
# Преобразование результата в список
result_list = list(zipped_data)
print(result_list)
# Вывод: [('Иван', 25), ('Мария', 30), ('Алексей', 22)]
```

2. Работа с тремя и более итерируемыми объектами:

```
names = ['Иван', 'Мария', 'Алексей']
ages = [25, 30, 22]
cities = ['Москва', 'Санкт-Петербург', 'Новосибирск']
# Объединение трех итерируемых объектов в кортежи
zipped_data = zip(names, ages, cities)
# Преобразование результата в список
result_list = list(zipped_data)
print(result_list)
# Вывод: [('Иван', 25, 'Москва'), ('Мария', 30, 'Санкт-Петербург'), ('Алексей', 22, 'Новосибирск')]
```

3. Работа с различными типами итерируемых объектов:

```
names = ['Иван', 'Мария', 'Алексей']
ages = (25, 30, 22)
is_student = [False, True, True]
# Объединение различных типов итерируемых объектов в кортежи
zipped_data = zip(names, ages, is_student)
# Преобразование результата в список
result_list = list(zipped_data)
print(result_list)
# Вывод: [('Иван', 25, False), ('Мария', 30, True), ('Алексей', 22, True)]
```

Функция `zip()` полезна, когда вам нужно работать с соответствующими элементами нескольких итерируемых объектов одновременно.

8. Самостоятельно изучите возможности модуля datetime. Каким функционалом по работе с датой и временем обладает этот модуль?

Модуль `datetime` в Python предоставляет классы для работы с датой и временем. Вот основные возможности, которыми обладает этот модуль:

1. Класс `datetime.datetime`:

- `datetime.datetime(year, month, day, hour=0, minute=0, second=0, microsecond=0)`: Создает объект `datetime` с указанными значениями года, месяца, дня и т.д.

- `datetime.now()`: Возвращает текущую дату и время.

- `datetime.strptime(string, format)`: Преобразует строку в объект `datetime` согласно указанному формату.

- Методы для работы с компонентами даты и времени, такие как `year`, `month`, `day`, `hour`, `minute`, `second`, `microsecond`, и другие.

2. Класс `datetime.date`:

- `datetime.date(year, month, day)`: Создает объект `date` с указанными значениями года, месяца и дня.

- Методы для работы с компонентами даты, такие как `year`, `month`, `day`.

3. Класс `datetime.time`:

- `datetime.time(hour=0, minute=0, second=0, microsecond=0)`: Создает объект `time` с указанными значениями часов, минут, секунд и т.д.

- Методы для работы с компонентами времени, такие как `hour`, `minute`, `second`, `microsecond`.

4. Работа с интервалами и временными разницами:

- `datetime.timedelta`: Позволяет представлять разницу между двумя датами или временем с точностью до микросекунд.

-`timedelta(days=0, seconds=0, microseconds=0, milliseconds=0, minutes=0, hours=0, weeks=0)`: Создает объект `timedelta` с указанной разницей.

5. Форматирование и парсинг даты и времени:

- `strftime(format)`: Преобразует объект `datetime` в строку согласно указанному формату.

- `strptime(string, format)`: Преобразует строку в объект `datetime` согласно указанному формату.

6. Работа с часовыми поясами:

- `datetime.timezone`: Класс для представления часового пояса.

- `datetime.tzinfo`: Абстрактный класс для представления информации о временной зоне.

7. Модуль `datetime` также включает другие полезные функции, такие как:

- `datetime.today()`: Возвращает текущую дату.

- `datetime.combine(date, time)`: Комбинирует объекты `date` и `time` в объект `datetime`.

- `datetime.min` и `datetime.max`: Представляют минимальную и максимальную даты и времени, соответственно.

Пример использования модуля `datetime`:

```
from datetime import datetime, timedelta
# Получение текущей даты и времени
now = datetime.now()
print("Текущая дата и время:", now)
# Создание объекта datetime
custom_date = datetime(2023, 5, 15, 10, 30)
print("Пользовательская дата и время:", custom_date)
# Вычисление разницы между двумя датами
difference = custom_date - now
print("Разница между датами:", difference)
# Добавление интервала времени
new_date = now + timedelta(days=7)
print("Новая дата после добавления интервала:", new_date)
```