3 amaliy topshiriq. Eshmatov B

Rasm,video,audio, word va pdf fayllarini topish.

```
pip install PyPDF2
```

```
Collecting PyPDF2
  Downloading pypdf2-3.0.1-py3-none-any.whl.metadata (6.8 kB)
Downloading pypdf2-3.0.1-py3-none-any.whl (232 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 232.6/232.6 kB 3.3 MB/s eta 0:00:00
Installing collected packages: PyPDF2
Successfully installed PyPDF2-3.0.1
```

```
pip install python-docx
```

```
Collecting python-docx
  Downloading python_docx-1.1.2-py3-none-any.whl.metadata (2.0 kB)
Requirement already satisfied: lxml>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from python-docx) (5.3.0)
Requirement already satisfied: typing-extensions>=4.9.0 in /usr/local/lib/python3.10/dist-packages (from python-docx) (4.12.2)
Downloading python_docx-1.1.2-py3-none-any.whl (244 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 244.3/244.3 kB 4.0 MB/s eta 0:00:00
Installing collected packages: python-docx
Successfully installed python-docx-1.1.2
```

```
pip install fuzzywuzzy
```

```
Collecting fuzzywuzzy
  Downloading fuzzywuzzy-0.18.0-py2.py3-none-any.whl.metadata (4.9 kB)
Downloading fuzzywuzzy-0.18.0-py2.py3-none-any.whl (18 kB)
Installing collected packages: fuzzywuzzy
Successfully installed fuzzywuzzy-0.18.0
```

Rasmni topish uchun quyidagicha kodlar yoziladi.

```
import os
import requests
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from PyPDF2 import PdfReader
from docx import Document
from fuzzywuzzy import fuzz
from PIL import Image
from io import BytesIO
import matplotlib.pyplot as plt

# 1. Hujjatlarni va rasmlarni saqlash uchun baza
documents = []
images = []

# 2. PDF fayllarni o'qish funksiyasi
def read_pdf(file_path):
    try:
        reader = PdfReader(file_path)
        text = ""
        for page in reader.pages:
            text += page.extract_text()
        return text
    except Exception as e:
        print(f"PDF faylni o'qishda xato: {e}")
        return ""

# 3. DOCX fayllarni o'qish funksiyasi
def read_docx(file_path):
    try:
        doc = Document(file_path)
        text = "\n".join([paragraph.text for paragraph in doc.paragraphs])
        return text
    except Exception as e:
        print(f"DOCX faylni o'qishda xato: {e}")
        return ""

# 4. Hujjatlarni va rasmlarni yuklash funksiyasi
def load_documents_from_folder(folder_path):
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        if file_name.endswith(".pdf"):
```

```
            text = read_pdf(file_path)
            if text.strip():  # Matn bo'sh bo'lmasligi kerak
                documents.append({"text": text, "link": file_path})
        elif file_name.endswith(".docx"):
            text = read_docx(file_path)
            if text.strip():  # Matn bo'sh bo'lmasligi kerak
                documents.append({"text": text, "link": file_path})
        elif file_name.lower().endswith((".png", ".jpg", ".jpeg")):
            images.append({"name": file_name, "link": file_path})

# 5. GitHubdagi rasmlarni yuklash funksiyasi
def load_images_from_github(image_urls):
    for url in image_urls:
        try:
            response = requests.get(url)
            if response.status_code == 200:
                img_name = os.path.basename(url)
                images.append({"name": img_name, "link": url})
            else:
                print(f"Rasmni yuklashda xato: {url}")
        except Exception as e:
            print(f"GitHubdan rasm yuklashda xato: {e}")

# 6. Foydalanuvchi qidiruv tizimini yaratish
def search_documents_and_images(query):
    if not documents and not images:
        print("Hujjatlar yoki rasmlar mavjud emas.")
        return

    results_found = False

    # 6.1 Matnli hujjatlarni qidirish
    if documents:
        print("\nMatnli hujjatlar bo‘yicha qidiruv natijalari:")
        texts = [doc["text"] for doc in documents]
        vectorizer = TfidfVectorizer(stop_words='english')
        doc_vectors = vectorizer.fit_transform(texts)
        query_vector = vectorizer.transform([query])

        similarities = cosine_similarity(query_vector, doc_vectors).flatten()
        sorted_indices = similarities.argsort()[::-1]

        for idx in sorted_indices:
            if similarities[idx] > 0:
                results_found = True
                doc = documents[idx]
                print(f"- Topildi: {similarities[idx]:.2f}")
                print(f"  Link: {doc['link']}\n")

    # 6.2 Rasmlarni qidirish
    if images:
        print("\nRasmlar bo‘yicha qidiruv natijalari:")
        for image in images:
            similarity = fuzz.partial_ratio(query.lower(), image["name"].lower())
            if similarity >= 30:
                results_found = True
                print(f"- Topildi: {similarity}%")
                print(f"  Rasm nomi: {image['name']}")
                print(f"  Link: {image['link']}")
                # Rasmlarni tasvirlash
                try:
                    if image["link"].startswith("http"):
                        response = requests.get(image["link"])
                        img = Image.open(BytesIO(response.content))
                    else:
                        img = Image.open(image["link"])
                    plt.imshow(img)
                    plt.axis('off')
                    plt.title(f"Rasm: {image['name']}")
                    plt.show()
                except Exception as e:
                    print(f"Rasmni ochishda xato: {e}")

    if not results_found:
        print("Hech qanday mos natija topilmadi.")

# 7. Fayllarni yuklash
folder_path = "/content"  # Mahalliy fayllar uchun papka yo'li
os.makedirs(folder_path, exist_ok=True)
load_documents_from_folder(folder_path)

# 8. GitHubdan rasmlarni yuklash
github_image_urls = [
```

```
    "https://raw.githubusercontent.com/username/repository/branch/image1.png",
    "https://raw.githubusercontent.com/username/repository/branch/image2.jpg"
]
load_images_from_github(github_image_urls)

# 9. Qidiruvni boshlash
query = input("Qidiruvni kiriting: ").strip()
search_documents_and_images(query)
```

```
Qidiruvni kiriting: ss

Rasmlar bo‘yicha qidiruv natijalari:
- Topildi: 100%
  Rasm nomi: ss.JPG
  Link: /content/ss.JPG
```

Rasm: ss.JPG

```
pip install mutagen fuzzywuzzy
```

```
Collecting mutagen
  Downloading mutagen-1.47.0-py3-none-any.whl.metadata (1.7 kB)
Requirement already satisfied: fuzzywuzzy in /usr/local/lib/python3.10/dist-packages (0.18.0)
Downloading mutagen-1.47.0-py3-none-any.whl (194 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.4/194.4 kB 3.4 MB/s eta 0:00:00
Installing collected packages: mutagen
Successfully installed mutagen-1.47.0
```

Qo'shiqni topish uchun quyidagicha kodlar yoziladi.

```
import os
from fuzzywuzzy import fuzz

# 1. Qo'shiqlarni saqlash uchun baza
songs = []

# 2. Qo'shiq haqida ma'lumotni o'qish funksiyasi
def get_song_metadata(file_path):
    try:
        metadata = {}
        metadata["title"] = os.path.basename(file_path)  # Fayl nomini qo'shiq sarlavhasi sifatida olamiz
        metadata["artist"] = "Noma'lum"  # Agar metadata mavjud bo'lmasa
        metadata["album"] = "Noma'lum"  # Agar metadata mavjud bo'lmasa
        metadata["link"] = file_path
        return metadata
    except Exception as e:
        print(f"Qo'shiq metadata o'qishda xato: {e}")
        return None

# 3. Qo'shiqlarni papkadan yuklash funksiyasi
def load_songs_from_folder(folder_path):
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        if file_name.lower().endswith((".mp3", ".wav", ".aac", ".flac", ".ogg")):
            metadata = get_song_metadata(file_path)
            if metadata:
                songs.append(metadata)

# 4. Qidiruv funksiyasi
def search_songs(query):
    if not songs:
```

```
            print("Qo'shiqlar mavjud emas.")
            return

        results_found = False
        print("\nQo'shiq qidiruv natijalari:")

        for song in songs:
            title_similarity = fuzz.partial_ratio(query.lower(), song["title"].lower())
            artist_similarity = fuzz.partial_ratio(query.lower(), song["artist"].lower())

            # Natijani 50% dan yuqori moslikda ko'rsatish
            if title_similarity >= 50 or artist_similarity >= 50:
                results_found = True
                print(f"- Qo'shiq: {song['title']} ({song['artist']})")
                print(f"  Albom: {song['album']}")
                print(f"  Fayl yo'li: {song['link']}\n")

        if not results_found:
            print("Hech qanday mos qo'shiq topilmadi.")

# 5. Papkadan qo'shiqlarni yuklash
folder_path = "/content/songs"  # Papka yo'lini kiriting
os.makedirs(folder_path, exist_ok=True)
load_songs_from_folder(folder_path)

# 6. Foydalanuvchi qidiruvini boshlash
query = input("Qidiruvni kiriting (qo'shiq nomi yoki ijrochi): ").strip()
search_songs(query)
```

```
Qidiruvni kiriting (qo'shiq nomi yoki ijrochi): janona

Qo'shiq qidiruv natijalari:
- Qo'shiq: janona.mp3 (Noma'lum)
  Albom: Noma'lum
  Fayl yo'li: /content/songs/janona.mp3
```

```
pip install python-Levenshtein
```

```
Collecting python-Levenshtein
  Downloading python_Levenshtein-0.26.1-py3-none-any.whl.metadata (3.7 kB)
Collecting Levenshtein==0.26.1 (from python-Levenshtein)
  Downloading levenshtein-0.26.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.2 kB)
Collecting rapidfuzz<4.0.0,>=3.9.0 (from Levenshtein==0.26.1->python-Levenshtein)
  Downloading rapidfuzz-3.11.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (11 kB)
Downloading python_Levenshtein-0.26.1-py3-none-any.whl (9.4 kB)
Downloading levenshtein-0.26.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (162 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 162.6/162.6 kB 3.3 MB/s eta 0:00:00
Downloading rapidfuzz-3.11.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.1/3.1 MB 31.4 MB/s eta 0:00:00
Installing collected packages: rapidfuzz, Levenshtein, python-Levenshtein
Successfully installed Levenshtein-0.26.1 python-Levenshtein-0.26.1 rapidfuzz-3.11.0
```

```
pip install pymediainfo
```

```
Collecting pymediainfo
  Downloading pymediainfo-6.1.0.tar.gz (446 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 446.5/446.5 kB 6.1 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: pymediainfo
  Building wheel for pymediainfo (setup.py) ... done
  Created wheel for pymediainfo: filename=pymediainfo-6.1.0-py2.py3-none-any.whl size=9237 sha256=3df38e6a22f40d7fa1bee4e575a1ec61f7
  Stored in directory: /root/.cache/pip/wheels/84/22/6b/374964dcdd11a5b38e46041739ca2cd5db9dc679c8373b19c3
Successfully built pymediainfo
Installing collected packages: pymediainfo
Successfully installed pymediainfo-6.1.0
```

```
pip install opencv-python
```

```
Requirement already satisfied: opencv-python in /usr/local/lib/python3.10/dist-packages (4.10.0.84)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-packages (from opencv-python) (1.26.4)
```

```
pip install fuzzywuzzy python-Levenshtein opencv-python
```

```
Requirement already satisfied: fuzzywuzzy in /usr/local/lib/python3.10/dist-packages (0.18.0)
Requirement already satisfied: python-Levenshtein in /usr/local/lib/python3.10/dist-packages (0.26.1)
Requirement already satisfied: opencv-python in /usr/local/lib/python3.10/dist-packages (4.10.0.84)
Requirement already satisfied: Levenshtein==0.26.1 in /usr/local/lib/python3.10/dist-packages (from python-Levenshtein) (0.26.1)
```

```
Requirement already satisfied: rapidfuzz<4.0.0,>=3.9.0 in /usr/local/lib/python3.10/dist-packages (from Levenshtein==0.26.1->python
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-packages (from opencv-python) (1.26.4)
```

Videoni topish uchun quyidagicha kodlar yoziladi.

```
import os
from fuzzywuzzy import fuzz
from IPython.display import display, HTML
import ipywidgets as widgets

# Video fayllar ro'yxatini saqlash uchun
videos = []

# 1. Videolarni yuklash funksiyasi
def load_videos_from_folder(folder_path):
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        if file_name.lower().endswith((".mp4", ".avi", ".mov", ".mkv")):
            videos.append({"name": file_name, "link": file_path})

# 2. Videolarni qidirish funksiyasi
def search_video(query):
    if not videos:
        print("Videolar mavjud emas.")
        return

    print("\nVideo fayllar bo‘yicha qidiruv natijalari:")
    for video in videos:
        similarity = fuzz.partial_ratio(query.lower(), video["name"].lower())
        if similarity >= 30:  # 30% yoki undan yuqori moslik
            print(f"- Topildi: {similarity}%")
            print(f"  Video nomi: {video['name']}")
            print(f"  Link: {video['link']}")
            play_video(video["link"])  # Videoni ijro qilish
            return

    print("Hech qanday mos natija topilmadi.")

# 3. Videoni ijro qilish funksiyasi
def play_video(video_path):
    print(f"Videoni ijro qilish: {video_path}")
    video = widgets.Video(value=open(video_path, "rb").read(), format='mp4', width=640, height=360)
    display(video)

# 4. Fayllarni yuklash
folder_path = "/content/videos"  # Bu yerga to'g'ri papka yo'lini kiriting
os.makedirs(folder_path, exist_ok=True)
load_videos_from_folder(folder_path)

# 5. Qidiruvni boshlash
query = input("Qidiruvni kiriting (video nomi): ").strip()
search_video(query)
```

```
Qidiruvni kiriting (video nomi): salom

Video fayllar bo‘yicha qidiruv natijalari:
- Topildi: 100%
  Video nomi: salom.mp4
  Link: /content/videos/salom.mp4
Videoni ijro qilish: /content/videos/salom.mp4
```

Pdf faylni topish uchun quyidagicha kodlar yoziladi.

```
import os
from fuzzywuzzy import fuzz
from IPython.display import display, HTML

# 1. PDF fayllarni saqlash uchun baza
pdf_files = []

# 2. PDF fayllarni papkadan yuklash funksiyasi
def load_pdf_files_from_folder(folder_path):
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        if file_name.lower().endswith(".pdf"):
            pdf_files.append({"name": file_name, "path": file_path})

# 3. PDF fayllarni qidirish funksiyasi
def search_pdf_file(query):
    if not pdf_files:
        print("PDF fayllar mavjud emas.")
        return

    results_found = False
    print("\nPDF fayllar bo‘yicha qidiruv natijalari:")

    for pdf_file in pdf_files:
        name_similarity = fuzz.partial_ratio(query.lower(), pdf_file["name"].lower())

        # Natijani 50% dan yuqori moslikda ko'rsatish
        if name_similarity >= 50:
            results_found = True
            print(f"- Fayl: {pdf_file['name']}")
            print(f"  Fayl yo'li: {pdf_file['path']}")
            print(f"  Moslik: {name_similarity}%")
            print("\nFaylni ochilmoqda...")
            display_pdf_file(pdf_file["path"])  # Faylni ochish
            return

    if not results_found:
        print("Hech qanday mos PDF fayl topilmadi.")

# 4. PDF faylni ochish funksiyasi
def display_pdf_file(file_path):
    try:
        file_url = f"file://{file_path}"
        display(HTML(f"""
            <iframe src="{file_url}" width="800" height="600" style="border: none;"></iframe>
        """))
    except Exception as e:
        print(f"PDF faylni ochishda xato: {e}")

# 5. Papkadan PDF fayllarni yuklash
folder_path = "/content/pdf_files"  # Papka yo'lini kiriting
os.makedirs(folder_path, exist_ok=True)
load_pdf_files_from_folder(folder_path)

# 6. Foydalanuvchi qidiruvini boshlash
query = input("Qidiruvni kiriting (PDF fayl nomi): ").strip()
search_pdf_file(query)
```

```
Qidiruvni kiriting (PDF fayl nomi): bir

PDF fayllar bo‘yicha qidiruv natijalari:
- Fayl: bir.pdf
  Fayl yo'li: /content/pdf_files/bir.pdf
  Moslik: 100%

Faylni ochilmoqda...
```

Word faylini topish uchun quyidagicha kodlar yoziladi.

```
import os
from fuzzywuzzy import fuzz
from IPython.display import display, HTML

# 1. Word fayllarni saqlash uchun baza
word_files = []

# 2. Word fayllarni papkadan yuklash funksiyasi
def load_word_files_from_folder(folder_path):
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        if file_name.lower().endswith((".doc", ".docx")):
            word_files.append({"name": file_name, "path": file_path})

# 3. Qidiruv funksiyasi
def search_word_file(query):
    if not word_files:
        print("Word fayllar mavjud emas.")
        return

    results_found = False
    print("\nWord fayllar bo‘yicha qidiruv natijalari:")

    for word_file in word_files:
        name_similarity = fuzz.partial_ratio(query.lower(), word_file["name"].lower())

        # Natijani 50% dan yuqori moslikda ko'rsatish
        if name_similarity >= 50:
            results_found = True
            print(f"- Fayl: {word_file['name']}")
            print(f"  Fayl yo'li: {word_file['path']}")
            print(f"  Moslik: {name_similarity}%")
            print("\nFaylni ochilmoqda...")
            display_word_file(word_file["path"])  # Faylni ochish
            return

    if not results_found:
        print("Hech qanday mos Word fayl topilmadi.")
```

```
# 4. Word faylni ochish funksiyasi
def display_word_file(file_path):
    try:
        file_url = f"file://{file_path}"
        display(HTML(f"""
            <iframe src="{file_url}" width="800" height="600" style="border: none;"></iframe>
        """))
    except Exception as e:
        print(f"Word faylni ochishda xato: {e}")

# 5. Papkadan Word fayllarni yuklash
folder_path = "/content/word_files"  # Papka yo'lini kiriting
os.makedirs(folder_path, exist_ok=True)
load_word_files_from_folder(folder_path)

# 6. Foydalanuvchi qidiruvini boshlash
query = input("Qidiruvni kiriting (Word fayl nomi): ").strip()
search_word_file(query)
```

```
Qidiruvni kiriting (Word fayl nomi): qwert

Word fayllar bo‘yicha qidiruv natijalari:
- Fayl: qwert.docx
  Fayl yo'li: /content/word_files/qwert.docx
  Moslik: 100%

Faylni ochilmoqda...
```