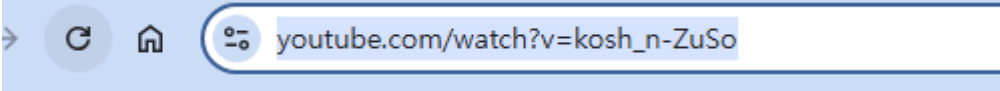
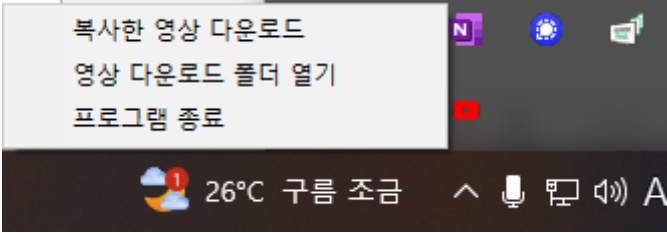
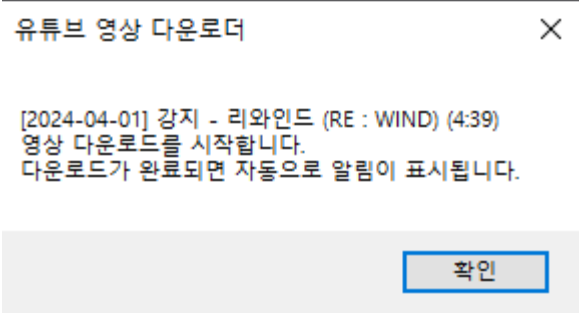




2025 중학생 알고리즘 멘토링 창의적 프로그래밍 프로젝트

- 원하는 주제로 프로그램을 작성해 보고 아래 서식을 채워주시면 됩니다. 해당 자료를 활용해서 수업 당일 간단한 발표와 시연을 하는 시간을 가질 것입니다. 추가적인 PPT 작성은 필요 없습니다.
- 아래 서식의 푸른색 글씨를 전부 지우고 검은색 글씨로 작성해 주시면 됩니다.
- Chat GPT 등 프로그램의 사용을 프로그램 작성할 때는 지양합니다. 서식 작성이나 발표 준비에는 필요에 따라 사용해도 됩니다.
- 발표 및 시연 시간은 발표 및 시연 5분 (필요에 따라 길어질 수 있습니다), 간단한 질의응답 3분 정도로 생각하고 있습니다.
- 수업 시간에 배운 파이썬 문법을 복습한다는 생각으로 하시면 됩니다. 수업 시간에 다루지 않은 문법이나 라이브러리는 배워서 마음껏 사용하셔도 됩니다.
- 해당 프로젝트는 평가가 아니라 저희 조끼리만 하는 활동임으로 편한 마음으로 하시면 될 것 같습니다.
- 반드시 수업 시작 전까지 마무리 해주세요.

이름: 장성주

작품 제목	유튜브 영상 다운로드
작품 목적	유튜브를 보다가 간단히 백업할 영상이 있거나 소장하고픈 영상이 있다면 프로그램을 키거나 사이트에 접속해서 링크를 붙여넣는 수고 없이 버튼 클릭 하나로만 영상이 다운로드되게 함.
작품 설명	<p>콘솔 창을 먼저 실행시켜서 최소화하면 프로그램이 실행됨(백그라운드에서 실행되게 하는 pythonw.exe도 사용 가능). 실행 후에는 작업 표시줄 트레이에 유튜브 아이콘이 표시됨.</p> <p>영상 다운로드 방법) 유튜브 영상 링크를 전체 복사(클립보드에 저장)한 후 작업표시줄 트레이에서 유튜브 영상 다운로드(유튜브 아이콘) 우클릭 -> “복사한 영상 다운로드” 메뉴 클릭 -> 자동으로 클립보드에 링크가 저장된 유튜브 영상이 다운로드됨</p> <p>영상 저장 위치) 기본적으로 “c:/yt_downloads“에 저장되며 작업표시줄 트레이에서 유튜브 영상 다운로드(유튜브 아이콘) 우클릭 -> “영상 다운로드 폴더 열기“ 메뉴 클릭을 하여 영상 다운로드 폴더를 열 수 있음. 프로그램이 실행될 때 “c:/yt_downloads” 폴더가 있는지 확인하며 (“os.path.exists“ 함수 사용) 없다면 “os.mkdir” 함수를 사용해 폴더를 만듦.</p> <p>Pytube 라이브러리) 기존 “pytube“ 라이브러리가 지원을 중단하였기 때문에 대체 라이브러리가 “pytube“ 라이브러리를 포크하여 가장 유지보수가 잘 되고 있는 라이브러리인 “pytube-fix”를 사용하기로 결정함.</p> <p>동영상 저장 로직) 유튜브는 일정 화질 이상의 영상에는 비디오와 음원을 따로 저장해서 이용자들에게 따로따로 전송함. 따라서 최고화질로 영상을 받기 위해서는 영상의 비디오와 오디오를 유튜브 서버에서 각각 따로 다운받아 클라이언트(PC) 내부에서 합친다.</p> <p>시간 변환 로직) 초를 받아서 시간:분:초 로 변환하는 함수를 작성함.</p> <p>파일 이름 문제) 파일 이름에 “:”, “\” 등의 특수문자는 윈도우 자체적으로 사용이 불가능함. 이에 자동으로 다운로드시 “:”, “\” 등의 특수문자를 공백으로 바꾸어 오류가 나지 않도록 함.</p> <p>사용한 라이브러리 모듈)</p> <p>Pystray - 작업 표시줄 트레이에 유튜브 아이콘 표시 + 메뉴 기능</p>

	<p>Pillow - 작업 표시줄 트레이에 유튜브 아이콘 로드</p> <p>Pyautogui - 알림 창 띄우기</p> <p>Threading - 동시 다운로드 기능 구현</p> <p>Clipboard - 클립보드 내용 받아오기</p> <p>Os - 폴더 존재여부 확인, 폴더 생성, 경로 문자열 생성</p> <p>Pytubefix - 유튜브 영상, 음원 다운로드</p> <p>Moviepy - 다운받은 유튜브 영상과 음원을 합침</p>
시연 사진	<div></div> <p>(링크 복사)</p> <div></div> <p>(메뉴 클릭)</p> <div></div> <p>(시작 알림창)</p> <div></div> <p>(콘솔창에 진행도)</p> <div></div> <p>(완료 알림창)</p>



(정상적으로 다운로드 완료, 최대 화질인 1080p 60fps 지원 확인)

소스 코드

```
from pystray import MenuItem, Icon
from PIL import Image
import pyautogui, threading, clipboard, os, pytubefix
from pytubefix import Playlist, YouTube
from moviepy.editor import VideoFileClip, AudioFileClip
DOWNLOAD_DIR = "C:\\yt_downloads"
downloading = []
if not os.path.exists(DOWNLOAD_DIR):
    os.mkdir(DOWNLOAD_DIR)
def s_to_hms(s: int):
    h, m, s = s//3600, s%3600//60, s%3600%60
    return f"{str(h) + ':' if h > 0 else ''}{str(m) + ':' if m > 0 else ''}{str(s) if s > 0 else ''}"
def video_add_audio(video_path, audio_path, output_file_path):
    video = VideoFileClip(video_path)
    new_audio = AudioFileClip(audio_path)
    video = video.set_audio(new_audio)
    video.write_videofile(output_file_path, codec='libx264', audio_codec='aac')
def convert_to_filename(name):
    invalid_chars = '\\:*?<>|'
    for char in invalid_chars:
        name = name.replace(char, '')
    return name
def download_video():
    global downloading
    try:
        # init
        url = clipboard.paste()
        yt = YouTube(url)
        if yt.title in downloading:
            pyautogui.alert(text=f"{yt.title}은 이미 다운로드 중 입니다.", title='유튜브 영상 다운로더', button='OK')
            return

        # 같은이름 파일 중복해서 다운하면 파일깨짐 (중복방지)
        downloading.append(yt.title)

        # 비디오 다운하는거
        resolution_list = []
        for idx, i in enumerate(yt.streams):
```

```

        resolution_list.append(
            int(str(i.resolution).replace("p", "")) if i.resolution else 0
        )
        idx = resolution_list.index(max(resolution_list))
        pyautogui.alert(text=f'{yt.title} ({s_to_hms(int(yt.length))})\n영상 다운로드를
시작합니다.\n다운로드가 완료되면 자동으로 알림이 표시됩니다.', title='유튜브 영상 다운로더',
button='OK')
        yt.streams[idx].download(output_path=DOWNLOAD_DIR) # 임마는 mp4
        # 오디오 다운하는거
        ys = yt.streams.get_audio_only()
        ys.download(output_path=DOWNLOAD_DIR) # 무조건 .m4a
        # 합치기
        video_add_audio(
            video_path=os.path.join(DOWNLOAD_DIR,
f"{convert_to_filename(yt.title)}.mp4"),
            audio_path=os.path.join(DOWNLOAD_DIR,
f"{convert_to_filename(yt.title)}.m4a"),
            output_file_path=os.path.join(DOWNLOAD_DIR,
f"{convert_to_filename(yt.title)}.mp4")
        )
        # 비디오는 덮어써졌으니까 오디오만 삭제 + 다운로드 리스트에서 삭제
        try:
            os.remove(os.path.join(DOWNLOAD_DIR,
f"{convert_to_filename(yt.title)}.m4a"))
            downloading.remove(yt.title)
        except:
            pass
        # nofi
        pyautogui.alert(text=f'{yt.title} 영상 다운로드가 완료되었습니다.', title='유튜브
영상 다운로더', button='OK')
    except:
        pyautogui.alert(text=f'영상 다운로드에 오류가 생겼습니다. 다시 시도해 주세요.',
title='유튜브 영상 다운로더', button='OK')
def on_clicked():
    work = threading.Thread(target=download_video)
    work.start()
def open_folder():
    os.startfile(os.path.realpath(DOWNLOAD_DIR))
def exit_program():
    icon.stop()
    exit()
image = Image.open("./assets/icon.ico")
menu = (
    MenuItem('복사한 영상 다운로드', on_clicked),
    MenuItem('영상 다운로드 폴더 열기', open_folder),
    MenuItem('프로그램 종료', exit_program),
)
icon = Icon("복사한 영상 다운로드", image, "유튜브 다운로더", menu)
icon.run()

(github : https://github.com/sungjujiang/yt\_downloader)

```

느낀 점

뭐 만들지 생각하다가 갑자기 영상이 다운받고 싶어져서 만들게 되었습니다. 간단하게 콘솔창에

	<p>만드려고만 했는데 표시줄 트레이로 옮겨지고 동시 다운로드 기능까지 구현하면서 시간이 1시간 남짓 걸렸습니다. 하지만 활동을 통해 pytube 에 대한 사용법을 더 자세히 알게 되었으며, 작업 표시줄 트레이에 표시하는 것이 사용자 접근성 측면에서 매우 효과적이라는 것을 알게 되었습니다. 앞으로는 GUI를 Tkinter를 사용해서 추가하거나 화질 선택 기능, 구간별 다운로드 기능, 동영상 압축 알고리즘을 통해 용량 줄이기 같은 기능을 추가해보고 싶다. 또한 평소에 코딩할 때 귀찮아서 디버깅이나 오류 해결을 지피티한테 맡기거나 코파일럿을 사용했는데 2-3년 전처럼 공식 문서를 보고 스택오버플로우를 보면서 개발해보니 세상이 빠르게 변하고 있다는 것을 느끼기도 하였다.</p>
--	--

