```python
#-------------------------------------------------------------------------------
# Name:        module1
# Purpose:
#
# Author:      ikeyan
#
# Created:     21/03/2025
# Copyright:   (c) ikeyan 2025
# Licence:     <your licence>
#-------------------------------------------------------------------------------

import sys
import re
import os
import glob
import subprocess

url = 'https://www.youtube.com/watch?v=W_fHWaoQwkw'
exe_path = r'E:\\98Download\\Youtube\\yt-dlp_win\\'
exe_file = 'yt-dlp_20250126.exe'
cmd_option = '--list-formats'
LIST_FORMATS_TABLE_HEADER_POS = 8
cmd = '{exe_path}{exe_file} {cmd_option} {url}'.format(exe_path = exe_path, exe_file=exe_file, cmd_option=cmd_option, url=url)
cp = subprocess.run(cmd , capture_output=True)
msg = cp.stdout.decode('shift-jis')
print(cmd)

data_string = msg
# 1. ヘッダー行の位置を取得
header_line = data_string.strip().split("\n")[LIST_FORMATS_TABLE_HEADER_POS]


# 固定長の文字位置を取得
column_positions = []
chain_string = True
for i, char in enumerate(header_line):
    if i == 0 or (char != ' ' and header_line[i - 1] == ' '):
        column_positions.append(i)

column_positions.append(len(header_line))  # 最後の位置を追加

# 列名を取得
header_titles = [header_line[column_positions[i]:column_positions[i + 1]].strip() for i in range(len(column_positions) - 1)]
print(header_titles)
# 2. データ行を項目別に辞書型に変換
data_lines = data_string.strip().split("\n")[LIST_FORMATS_TABLE_HEADER_POS+2:]  # ヘッダー行をスキップ
parsed_data = []

for line in data_lines:
    record = {}
    for i in range(len(column_positions) - 1):
        start, end = column_positions[i], column_positions[i + 1]
        record[header_titles[i]] = line[start:end].strip()
    parsed_data.append(record)


# 条件 EXT = mp4 and RESOLUTION = 1280x720 and VCODEC = avc1.4D401F
filtered_data = [item for item in parsed_data if item['EXT'] == 'mp4' and item['RESOLUTION']=='1280x720' and 'avc1.4D401F' in item['VCODEC']]
if len(filtered_data) ==0:
    filtered_data = [item for item in parsed_data if item['EXT'] == 'mp4' and item['RESOLUTION']=='640x360' and 'avc1' in item['VCODEC']]

if len(filtered_data) ==0:
    filtered_data = [item for item in parsed_data if item['EXT'] == 'mp4' and item['RESOLUTION']=='720x720' and 'avc1' in item['VCODEC']]

if len(filtered_data) >= 1:
    video_id = filtered_data[0].get('ID','')
```

```
len(filtered_data) >= 1:
 audio_id = filtered_data[0].get('ID','')
se:
 print('audio_idの取得に失敗しました。処理を中断します。')
 print(msg)
 sys.exit()

ダウンロードコマンド生成＋ダウンロード
d_opt_format_select = '-f {video}+{audio}'.forma
d_opt_out_format = '--merge-output-format mp4'
d = '{exe_path}{exe_file} {format_select} {url}
e_path, exe_file=exe_file,

rint(cmd)
= subprocess.run(cmd , capture_output=True)
```

```
67  else:
68      print('video_idの取得に失敗しました。処理を中断します。')
69      print(msg)
70      sys.exit()
71
72  # 条件 EXT = m4a and RESOLUTION = audio only and VCODEC = mp4a.40.2
73  filtered_data = [item for item in parsed_data if item['EXT'] == 'm4a' and item[
    'RESOLUTION']=='audio only' and 'mp4a' in item['ACODEC']]
74  if len(filtered_data) ==0:
75      filtered_data = [item for item in parsed_data if item['EXT'] == 'mp4' and item[
        'RESOLUTION']=='audio only']
76
77  if len(filtered_data) >= 1:
78      audio_id = filtered_data[0].get('ID','')
79  else:
80      print('audio_idの取得に失敗しました。処理を中断します。')
81      print(msg)
82      sys.exit()
83
84  # ダウンロードコマンド生成＋ダウンロード
85  cmd_opt_format_select = '-f {video}+{audio}'.format(video=video_id, audio=audio_id)
86  cmd_opt_out_format = '--merge-output-format mp4'
87  cmd = '{exe_path}{exe_file} {format_select} {url} {out_format}'.format(exe_path =
    exe_path, exe_file=exe_file,
88                                                                          format_select=
                                                                            cmd_opt_format_select,
89                                                                          url=url,
90                                                                          out_format=
                                                                            cmd_opt_out_format)
91  #print(cmd)
92  cp = subprocess.run(cmd , capture_output=True)
93
94
95
96
97  def main():
98      pass
99
100 if __name__ == '__main__':
101     main()
102
```