중위소득뽑기

import csv

import re

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_median\_income.csv"

# 중위소득을 저장할 리스트

median\_incomes = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

# 중위소득 추출하기

median\_income = re.findall(r'\d', data)

if median\_income:

median\_incomes.append(median\_income[0])

else:

median\_incomes.append("-")

# CSV 파일 쓰기

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "중위소득"])

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row, median\_income in zip(reader, median\_incomes):

writer.writerow([row["bizid"], median\_income])

장학금 지원 구간뽑기

import csv

import re

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_income\_out.csv"

# 구간 정보를 저장할 리스트

income\_quintiles = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

# 학자금 지원이 포함된 경우

if "학자금 지원" in data:

# 구간 정보 추출하기

quintile = re.findall(r'\d', data)

if quintile:

# 구간 정보가 1~9 구간이면 9로 저장

if "9구간" in quintile[0]:

income\_quintiles.append("9")

else:

income\_quintiles.append(quintile[0][0])

else:

income\_quintiles.append("-")

else:

income\_quintiles.append("-")

# CSV 파일 쓰기

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "학자금 지원 구간"])

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row, income\_quintile in zip(reader, income\_quintiles):

writer.writerow([row["bizid"], income\_quintile])

무주택자뽑기

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_homelessness.csv"

# 데이터를 저장할 리스트

homelessness\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 무주택 포함 여부 확인

if "무주택" in data:

homelessness\_data.append((biz\_id, "1"))

else:

homelessness\_data.append((biz\_id, "-"))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "무주택 여부"])

writer.writerows(homelessness\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

2년제 4년제

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_education.csv"

# 데이터를 저장할 리스트

education\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 2년제 여부 확인

if "2년제" in data:

education\_data.append((biz\_id, "2"))

elif "4년제" in data:

education\_data.append((biz\_id, "4"))

else:

education\_data.append((biz\_id, "-"))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "학제"])

writer.writerows(education\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

1인가구 뽑기

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_education.csv"

# 데이터를 저장할 리스트

single\_household\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 1인가구 포함 여부 확인

is\_single\_household = "1인가구" in data

# 1인가구인 경우 1로, 그렇지 않은 경우 -

single\_household\_value = "1" if is\_single\_household else "-"

single\_household\_data.append((biz\_id, single\_household\_value))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "1인가구 여부"])

writer.writerows(single\_household\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

차상위뽑기

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_near\_poverty.csv"

# 데이터를 저장할 리스트

education\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 차상위계층 포함 여부 확인

is\_middle\_class = "차상위" in data

# 차상위계층이 포함된 경우 1로, 그렇지 않은 경우 -

if is\_middle\_class:

education\_value = "1"

else:

education\_value = "-"

education\_data.append((biz\_id, education\_value))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "차상위계층"])

writer.writerows(education\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

기초수급및생계급여

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_11.csv"

# 데이터를 저장할 리스트

near\_poverty\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 기초생활이나 생계급여 포함 여부 확인

is\_near\_poverty = "기초생활" in data or "생계급여" in data

# 기초생활이나 생계급여가 포함된 경우 1로, 그렇지 않은 경우 -

education\_value = "1" if is\_near\_poverty else "-"

near\_poverty\_data.append((biz\_id, education\_value))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "기초생활 및 생계급여"])

writer.writerows(near\_poverty\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

신혼부부뽑기

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_11.csv"

# 데이터를 저장할 리스트

near\_poverty\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 기초생활이나 생계급여 포함 여부 확인

is\_near\_poverty = "신혼" in data

# 기초생활이나 생계급여가 포함된 경우 1로, 그렇지 않은 경우 -

education\_value = "1" if is\_near\_poverty else "-"

near\_poverty\_data.append((biz\_id, education\_value))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "신혼부부"])

writer.writerows(near\_poverty\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

농업인뽑기

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_11.csv"

# 데이터를 저장할 리스트

near\_poverty\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 기초생활이나 생계급여 포함 여부 확인

is\_near\_poverty = "농업인" in data

# 기초생활이나 생계급여가 포함된 경우 1로, 그렇지 않은 경우 -

education\_value = "1" if is\_near\_poverty else "-"

near\_poverty\_data.append((biz\_id, education\_value))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "농업인"])

writer.writerows(near\_poverty\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")

소상공인

import csv

# 입력 파일 경로

input\_file = "all\_income\_quintile.csv"

# 출력 파일 경로

output\_file = "all\_out\_11.csv"

# 데이터를 저장할 리스트

near\_poverty\_data = []

# CSV 파일 읽기

with open(input\_file, "r", encoding="utf-8") as file:

reader = csv.DictReader(file)

for row in reader:

# 거주지 및 소득 데이터 가져오기

data = row["거주지 및 소득"]

biz\_id = row["bizid"]

# 소상공인 포함 여부 확인

is\_near\_poverty = "소상공인" in data

# 기초생활이나 생계급여가 포함된 경우 1로, 그렇지 않은 경우 -

education\_value = "1" if is\_near\_poverty else "-"

near\_poverty\_data.append((biz\_id, education\_value))

# 추출된 데이터를 CSV 파일로 저장

with open(output\_file, "w", encoding="utf-8", newline="") as file:

writer = csv.writer(file)

writer.writerow(["bizid", "소상공인"])

writer.writerows(near\_poverty\_data)

print("추출이 완료되었습니다. 결과는", output\_file, "파일에 저장되었습니다.")