***example1.py***

import csv

import os

def process\_data(file\_path):

if not os.path.exists(file\_path):

print("File not found!")

return

with open(file\_path, 'r') as file:

reader = csv.reader(file)

headers = next(reader)

data = []

for row in reader:

processed\_row = {}

for i, value in enumerate(row):

try:

processed\_row[headers[i]] = int(value)

except ValueError:

processed\_row[headers[i]] = value

data.append(processed\_row)

report = {}

for row in data:

category = row.get("Category", "Unknown")

if category in report:

report[category] += 1

else:

report[category] = 1

print("Category Report:")

for category, count in report.items():

print(f"{category}: {count}")

if \_\_name\_\_ == "\_\_main\_\_":

process\_data("data.csv")

***example2.py***

import csv

import os

from typing import List, Dict

def read\_csv(file\_path: str) -> List[Dict[str, str]]:

"""Reads a CSV file and returns structured data."""

if not os.path.exists(file\_path):

raise FileNotFoundError(f"File not found: {file\_path}")

with open(file\_path, 'r') as file:

reader = csv.reader(file)

headers = next(reader)

return process\_rows(reader, headers)

def process\_rows(reader: csv.reader, headers: List[str]) -> List[Dict[str, str]]:

"""Processes CSV rows and returns structured data."""

data = []

for row in reader:

processed\_row = {headers[i]: parse\_value(value) for i, value in enumerate(row)}

data.append(processed\_row)

return data

def parse\_value(value: str):

"""Attempts to convert a string to an integer, otherwise returns the original string."""

try:

return int(value)

except ValueError:

return value

def generate\_report(data: List[Dict[str, str]]) -> None:

"""Generates a report from the processed data."""

report = {}

for row in data:

category = row.get("Category", "Unknown")

report[category] = report.get(category, 0) + 1

print("Category Report:")

for category, count in report.items():

print(f"{category}: {count}")

def main():

"""Main execution function."""

try:

data = read\_csv("data.csv")

generate\_report(data)

except FileNotFoundError as e:

print(e)

except Exception as e:

print(f"Unexpected error: {e}")

if \_\_name\_\_ == "\_\_main\_\_":

main()