1

import streamlit as st

import csv

import io

st.title("Filtrar correos electrónicos por edad")

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo CSV", type="txt")

if uploaded\_file is not None:

# Leer archivo

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

csv\_reader = csv.reader(input\_file, delimiter=',')

for row in csv\_reader:

if len(row) >= 3:

nombre = row[0].strip()

try:

edad = int(row[1].strip())

except ValueError:

continue

email = row[2].strip()

if edad > 18:

output\_file.write(email + '\n')

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("Filtrado completado. Revisa los correos electrónicos filtrados abajo.")

# Mostrar los correos electrónicos filtrados

st.text\_area("Correos electrónicos filtrados:", output\_file.getvalue(), height=200)

# Descarga del archivo filtrado

st.download\_button(

label="Descargar archivo filtrado",

data=output\_file.getvalue(),

file\_name="emails\_filtrados.txt",

mime="text/plain"

)

2

import streamlit as st

import io

st.title("Cálculo de Promedios de Notas")

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo de texto", type="txt")

if uploaded\_file is not None:

# Leer archivo

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

for line in input\_file:

partes = line.strip().split(' ', 1)

if len(partes) < 2:

output\_file.write(f"{partes[0]}, Promedio: No se pueden calcular las notas\n")

continue

nombre = partes[0]

materia\_nota = partes[1]

suma\_notas = 0.0

num\_materias = 0

ss = materia\_nota.split()

i = 0

while i < len(ss):

materia = ss[i]

i += 1

if i < len(ss):

try:

nota = float(ss[i])

suma\_notas += nota

num\_materias += 1

except ValueError:

pass

i += 1

if num\_materias > 0:

promedio = suma\_notas / num\_materias

output\_file.write(f"{nombre}, Promedio: {promedio:.2f}\n")

else:

output\_file.write(f"{nombre}, Promedio: No se pueden calcular las notas\n")

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("El cálculo de promedios ha sido completado. Revisa los resultados abajo.")

# Mostrar los resultados

st.text\_area("Resultados:", output\_file.getvalue(), height=200)

# Descarga del archivo de salida

st.download\_button(

label="Descargar resultados",

data=output\_file.getvalue(),

file\_name="salida.txt",

mime="text/plain"

)

3

import streamlit as st

import io

st.title("Conversión de Precios de Dólares a Soles")

# Tasa de conversión

tasa\_conversion = 3.85

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo de texto", type="txt")

if uploaded\_file is not None:

# Leer archivo

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

for line in input\_file:

partes = line.strip().split()

if len(partes) != 2:

continue

producto = partes[0]

try:

precio\_dolares = float(partes[1])

except ValueError:

continue

# Conversión de precio

precio\_soles = precio\_dolares \* tasa\_conversion

output\_file.write(f"{producto}, {precio\_soles:.2f}\n")

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("La conversión ha sido completada. Revisa los resultados abajo.")

# Mostrar los resultados

st.text\_area("Resultados:", output\_file.getvalue(), height=200)

# Descarga del archivo de salida

st.download\_button(

label="Descargar resultados",

data=output\_file.getvalue(),

file\_name="salida.txt",

mime="text/plain"

)

4

import streamlit as st

import io

st.title("Cálculo de Temperaturas Máximas y Mínimas")

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo de texto", type="txt")

if uploaded\_file is not None:

max\_temp = float('-inf')

min\_temp = float('inf')

fecha\_max = ""

fecha\_min = ""

try:

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

for line in input\_file:

partes = line.strip().split(',')

if len(partes) != 2:

continue

fecha = partes[0].strip()

try:

temperatura = float(partes[1].strip())

except ValueError:

continue

if temperatura > max\_temp:

max\_temp = temperatura

fecha\_max = fecha

if temperatura < min\_temp:

min\_temp = temperatura

fecha\_min = fecha

output\_file.write(f"Día de temperatura máxima: {fecha\_max}, {max\_temp}\n")

output\_file.write(f"Día de temperatura mínima: {fecha\_min}, {min\_temp}\n")

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("El cálculo de temperaturas ha sido completado. Revisa los resultados abajo.")

# Mostrar los resultados

st.text\_area("Resultados:", output\_file.getvalue(), height=200)

# Descarga del archivo de salida

st.download\_button(

label="Descargar resultados",

data=output\_file.getvalue(),

file\_name="salida.txt",

mime="text/plain"

)

except IOError as e:

st.error(f"Error al procesar el archivo: {e}")

5

import streamlit as st

import io

st.title("Procesamiento de Ventas Diarias")

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo de texto", type="txt")

if uploaded\_file is not None:

total\_ventas = 0.0

max\_venta = float('-inf')

min\_venta = float('inf')

contador\_ventas = 0

fecha\_max\_venta = ""

fecha\_min\_venta = ""

try:

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

for line in input\_file:

partes = line.strip().split(',')

if len(partes) != 2:

continue

fecha = partes[0].strip()

try:

venta = float(partes[1].strip())

except ValueError:

continue

total\_ventas += venta

contador\_ventas += 1

if venta > max\_venta:

max\_venta = venta

fecha\_max\_venta = fecha

if venta < min\_venta:

min\_venta = venta

fecha\_min\_venta = fecha

if contador\_ventas > 0:

promedio\_ventas = total\_ventas / contador\_ventas

else:

promedio\_ventas = 0.0

output\_file.write(f"Venta total: {total\_ventas}\n")

output\_file.write(f"Promedio de ventas: {promedio\_ventas}\n")

output\_file.write(f"Día de mayor venta: {fecha\_max\_venta}, {max\_venta}\n")

output\_file.write(f"Día de menor venta: {fecha\_min\_venta}, {min\_venta}\n")

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("Los datos han sido procesados y guardados. Revisa los resultados abajo.")

# Mostrar los resultados

st.text\_area("Resultados:", output\_file.getvalue(), height=200)

# Descarga del archivo de salida

st.download\_button(

label="Descargar resultados",

data=output\_file.getvalue(),

file\_name="salida.txt",

mime="text/plain"

)

except IOError as e:

st.error(f"Error al procesar el archivo: {e}")

6

import streamlit as st

import io

st.title("Procesamiento de Horas de Empleados")

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo de texto", type="txt")

if uploaded\_file is not None:

horas\_empleados = {}

try:

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

for line in input\_file:

# Separamos la línea en nombre y horas

partes = line.strip().split(',')

if len(partes) != 2:

continue

nombre = partes[0].strip()

try:

horas = int(partes[1].strip())

except ValueError:

continue

if nombre in horas\_empleados:

horas\_empleados[nombre] += horas

else:

horas\_empleados[nombre] = horas

for nombre, total\_horas in horas\_empleados.items():

output\_file.write(f"{nombre}, Horas Totales: {total\_horas}\n")

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("Los datos han sido procesados y guardados. Revisa los resultados abajo.")

# Mostrar los resultados

st.text\_area("Resultados:", output\_file.getvalue(), height=200)

# Descarga del archivo de salida

st.download\_button(

label="Descargar resultados",

data=output\_file.getvalue(),

file\_name="salida.txt",

mime="text/plain"

)

except IOError as e:

st.error(f"Error al procesar el archivo: {e}")

7

import streamlit as st

import io

st.title("Procesamiento de Errores en Log")

# Subir archivo

uploaded\_file = st.file\_uploader("Elige un archivo de log", type="txt")

if uploaded\_file is not None:

contador\_errores = {}

try:

input\_file = io.StringIO(uploaded\_file.getvalue().decode("utf-8"))

output\_file = io.StringIO()

for linea in input\_file:

# Busca la palabra 'ERROR' en la línea.

if 'ERROR' in linea:

# Extrae el tipo de error y actualiza el contador.

tipo\_error = linea.split(':')[0].strip()

if tipo\_error in contador\_errores:

contador\_errores[tipo\_error] += 1

else:

contador\_errores[tipo\_error] = 1

for tipo\_error, conteo in contador\_errores.items():

output\_file.write(f"{tipo\_error}: {conteo}\n")

# Reseteamos el puntero al inicio del archivo de salida

output\_file.seek(0)

st.success("Los datos han sido procesados y guardados. Revisa los resultados abajo.")

# Mostrar los resultados

st.text\_area("Resumen de errores:", output\_file.getvalue(), height=200)

# Descarga del archivo de salida

st.download\_button(

label="Descargar resumen",

data=output\_file.getvalue(),

file\_name="resumen.txt",

mime="text/plain"

)

except IOError as e:

st.error(f"Error al procesar el archivo: {e}")