Explicación de solución punto 1

Por: Sebastian Carmona Estrada

Fecha: 06/02/2024

Prueba: **CARGA DE INFORMACIÓN**

“Cargar un data set, realizar el cargue y depuración del archivo OFEI1204.txt. Se debe entregar una tabla con las columnas: Agente Planta Hora_1 Hora_2 Hora_3 ... Hora_24 Solamente procesar los registros Tipo D. Enviar junto con la tabla resultante el código utilizado. Explicar el paso a paso en un archivo de texto (.doc o .pdf).”

Solución:

```python
"""
DataKnow Technical test

autor: Sebastian Carmona Estrada
"""
import numpy as np
import pandas as pd

def custom_parser(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        temp_dict = {}
        temp_dict["Agente"] = []
        temp_dict["Planta"] = []
        for h in np.arange(1,25):
            temp_dict["Hora_{h}".format(h=h)] = []

        for row in file:
            if not row.isspace():
                if row.find("AGENTE") != -1:
                    agent = row.split(":")[1].strip()

                elif 'agent' in locals():
                    if row.split(",")[1].strip() == 'D':
                        temp_dict["Agente"].append(agent)
                        temp_dict["Planta"].append(row.split(',')[0].strip())

                        for h in np.arange(1,25):
                            temp_dict["Hora_{h}".format(h=h)].append(row.split(',')[int(h + 1)].strip())


        return pd.DataFrame(temp_dict)

def main():
    df = custom_parser(r"punto_1\OFEI1204.txt")
    df.to_excel(r"punto_1\resultado.xlsx")
    print(df)

if __name__ == "__main__":
    main()
```

Explicación:

1. Para este código, se crea una función llamada *custom_parser()* para recolectar y organizar la información requerida.
2. La función *custom_parser()* tiene un parámetro el cual corresponde a la ruta del archivo a procesar.
3. Línea 10: Se abre el archivo haciendo uso de *with().* Aquí se especifica el tipo de codificación del archivo. En este caso, corresponde a una codificación tipo “utf-8”. Además, se abre el archivo en modo lectura “r”.
4. Línea 11: Se crea un diccionario en el cual se va a guarda la información requerida de forma estructurada.
5. Línea 12 – 15: Se crean las llaves correspondientes a los datos deseados, en este caso, “Agente”, “Planta” y 24 columnas correspondiente a la “Hora_n”, con n entre 1 y 24.
6. Línea 17: Se realiza un ciclo el cual itera en cada una de las líneas del documento procesado, utilizando la variable “row” como la variable de iteración.
7. Línea 18: Ya que el archivo contiene líneas en blanco, solo se tiene en cuenta las líneas con información diferente de espacios, utilizando la expresión, *no row.isspace().*
8. Línea 19: Se verifica que la cadena *“AGENTE”* se encuentra en *row.* Si esto es verdad, la expresión *row.find("AGENTE")* da un valor diferente de -1, lo cual hace verdadero el condicional.
9. Línea 20: Se extrae el nombre del AGENTE con el método split().
10. Línea 22: Si el condicional mencionado es falso, se ejecuta esta línea. Aquí, se verifica que la variable *agent* está definida con el fin de validar que se tiene un valor de agente. Esto se realiza para evitar que el condicional sea verdadero en la primera línea del dataset, la cual no contiene la cadena de “AGENTE”, por lo que se ejecutaría el resto de código sin tener la variable *agent* definida, lo cual resultaría en un error.
11. Línea 23: En esta línea se verifica que se trata de un registro tipo “D” con el método split(). Se utiliza el método strip() para eliminar caracteres de espacio que puedan ser invisibles y se generen falsos negativos.
12. Línea: 24 – 28: Se extrae la información requerida de row mediante el método split() para crear una lista de cadenas en las que se crea un nuevo elemento cada vez que se encuentra con el carácter “,” en row. Además, se retiran los espacios al principio y final de cada elemento creado con el método strip().
13. Línea 31: Se crea un objeto pandas.dataFrame pasando como parámetro el diccionario con los datos requeridos ordenados.
14. Línea 33: Se define la función main() con el fin de contener todo el código y siguiendo buenas practicas de programación.
15. Línea 35: Se guarda el pandas.dataFrame resultante en un archivo de Excel.

El código puede mejorar en términos de buenas prácticas de programación, añadiendo comentarios para documentar las funciones creadas, especificando los parámetros de entrada, sus tipos de datos u objeto y las variables de retorno de la función con su tipo de objeto.