Task 1: Write a function `parse_file` that parses the provided file and returns a list of `Section` objects

The provided file is split into multiple sections of data - each section consists of either line entries or tabular entries

- Line entries represent a name, unit, and a single scalar value
- Tabular entries represent a name, unit, and a list of scalar values

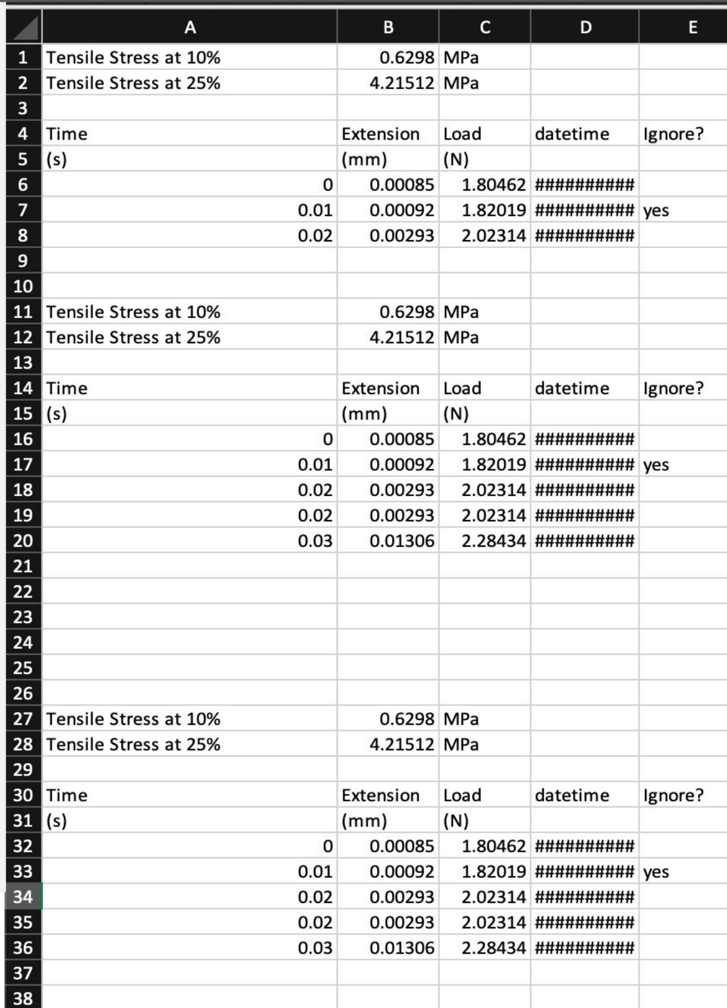
One Section object should be created per section, containing all the line entries or tabular entries in that section.

Bonus: Skip tabular entries where Ignore? == yes

```
# Object types

Entry:
    name: str
    unit: str | None
    data: str | list[str]

Section:
    data: list[Entry]
```

```
# Example function return value

[
    Section {
        data: [
            Entry { name: "Tensile Stress at 10%", unit: "MPa", data: "0.6298" },
            Entry { name: "Tensile Stress at 25%", unit: "MPa", data: "4.21512" }
        ]
    },
    Section {
        data: [
            Entry { name: "Time", unit: "(s)", data: ["0", "0.01", "0.02"] },
            Entry { name: "Extension", unit: "(s)", data: ["0.00085", “0.00092”, 0.00293”] },
            ...
        ]
    },
    ...
]
```

Download file: https://drive.google.com/file/d/1o8dFEjUuVJHX1EjfKMrezmPiXxX5206A/view?usp=sharing



	A	B	C	D	E
1	Tensile Stress at 10%	0.6298	MPa		
2	Tensile Stress at 25%	4.21512	MPa		
3					
4	Time	Extension	Load	datetime	Ignore?
5	(s)	(mm)	(N)		
6	0	0.00085	1.80462	##########	
7	0.01	0.00092	1.82019	##########	yes
8	0.02	0.00293	2.02314	##########	
9					
10					
11	Tensile Stress at 10%	0.6298	MPa		
12	Tensile Stress at 25%	4.21512	MPa		
13					
14	Time	Extension	Load	datetime	Ignore?
15	(s)	(mm)	(N)		
16	0	0.00085	1.80462	##########	
17	0.01	0.00092	1.82019	##########	yes
18	0.02	0.00293	2.02314	##########	
19	0.02	0.00293	2.02314	##########	
20	0.03	0.01306	2.28434	##########	
21					
22					
23					
24					
25					
26					
27	Tensile Stress at 10%	0.6298	MPa		
28	Tensile Stress at 25%	4.21512	MPa		
29					
30	Time	Extension	Load	datetime	Ignore?
31	(s)	(mm)	(N)		
32	0	0.00085	1.80462	##########	
33	0.01	0.00092	1.82019	##########	yes
34	0.02	0.00293	2.02314	##########	
35	0.02	0.00293	2.02314	##########	
36	0.03	0.01306	2.28434	##########	
37					
38					

Task 2: Write a function `excelSum` that accepts an array of cell values and a formula, and returns the sum of all cells specified by the formula.

function excelSum(data, formula)

data is a 2-d array
formula is a string

Returns sum of cells specified in formula. Return null if formula is invalid or out of bounds.

Format of Formula:

“=Sum([arg1], [arg2],…)”

Each argument is either a cell, e.g. “A1” or a range, e.g. “[cell1]:[cell2]”

Restrictions:

- Cells must be contained within table
- Ranges must be valid cell1 must be up and to the left of cell2 (e.g. “C2:B1” is not valid)

Bonus: Columns are lettered, A-Z and then AA – AZ, then BA – BZ…ZA – ZZ, then AAA – AAZ and so on.

	A	B	C	D	E	F	G	H	I
1	A1	B1	C1	D1	E1	F1	G1	H1	I1
2	A2	B2	C2	D2	E2	F2	G2	H2	I2
3	A3	B3	C3	D3	E3	F3	G3	H3	I3
4	A4	B4	C4	D4	E4	F4	G4	H4	I4
5	A5	B5	C5	D5	E5	F5	G5	H5	I5
6	A6	B6	C6	D6	E6	F6	G6	H6	I6
7	A7	B7	C7	D7	E7	F7	G7	H7	I7

Example Formulas:

- **“=Sum(A1)”**
- **“=Sum(A1, A2)”**
- **“=Sum(A1:A7)”**
- **“=Sum(B1, C2:E6)”**
- **“=Sum(A1:I7,E3,F3:G6,A1:A1)”**