Applesoft coding with Claude 3.7 Sonnet

Please write an Applesoft Basic program that reads a text file containing OPML and outputs a text file of executable DATA statements where each data statement contains a level number derived from the level of the OPML statement

I'll write an Applesoft Basic program that processes an OPML file and generates DATA statements with level information.

opml-to-data.txt

This program reads an OPML file and converts each outline element to DATA statements that include the nesting level of each item. Here's how it works:

1. It prompts for input and output filenames
2. Opens both files (for reading and writing)
3. Processes the OPML file line by line:
    - Skips XML and OPML headers
    - Tracks the nesting level by incrementing when it encounters an opening outline tag
    - Decrements the level when it finds a closing outline tag
    - Extracts the "text" attribute from each outline element
    - Outputs a DATA statement with the format: `[line number] DATA [level],”[text]"`

For example, if your OPML looks like:

```
<OPML version="1.0">

<outline text="Level 1 Item">

  <outline text="Level 2 Item">

    <outline text="Level 3 Item"/>

  </outline>

  <outline text="Another Level 2 Item"/>

</outline>

</OPML>
```

The output would be:

```
1000 DATA 1,"Level 1 Item"

1010 DATA 2,"Level 2 Item"
```

```
1020 DATA 3,"Level 3 Item"

1030 DATA 2,"Another Level 2 Item”
```

Would you like me to make any adjustments to this program?

Please write a program that will read the DATA statements formatted as in the previous program and display a given number of them (controlled by a variable) on an 80 column screen, truncating any text that exceeds 80 columns, where the level number is used to indent the statement by 2 spaces

I'll create an Applesoft BASIC program that reads those DATA statements and displays them with proper indentation based on level.

display-data-statements.txt

This program displays DATA statements with indentation based on their level number. Here's how it works:

1. It uses a variable `MAX_DISPLAY` (set to 20) to control how many items are displayed
2. Each level adds 2 spaces of indentation (`INDENT_SPACES`)
3. It reads each DATA statement, which contains a level number and text string
4. For each item:
    - Calculates the indentation based on level (level × 2 spaces)
    - Checks if the text will fit in the remaining screen width (80 columns minus indentation)
    - Truncates the text if necessary to fit on the screen
    - Displays the item with proper indentation

The program includes some sample DATA statements at the end (lines 820-890). In a real application, these would be replaced with the DATA statements generated by the previous program.

For example, with the sample data, the output would look like:

```
INDENTED DATA DISPLAY (MAX 20 ITEMS)

================================================



Level 1 Item

  Level 2 Item

    Level 3 Item

  Another Level 2 Item
```

```
Second Level 1 Item
  Child of Second Level 1 Item with a very long text that will need to be truncated because
Third Level 1 Item
  Final Level 2 Item


DISPLAYED 8 ITEMS.
```

Note that line 870 would be truncated to fit the screen width.

To use this with your previous program, you would:

Run the first program to generate DATA statements from OPML

Take those DATA statements and add them to this program (replacing lines 820-890)

Adjust the MAX_DISPLAY variable if you want to show more or fewer items

Would you like me to modify anything about this program?