

TorahBibleCodes: Free, Open-Source Python Bible Search and Research Software – User-Guide for Software Update Version 0.2

18 / November / 2024

Daniel Azariah
TorahBibleCodes.com | AzariahBible.com
info@TorahBibleCodes.com

ABSTRACT: Presented here is a user guide for TorahBibleCodes – the Free, Open-Source Python Bible Search and Research Software for Update Version 0.2. This software provides scientific (mathematical) tools, measurements, and a standardized mathematical ID system that enable custom search and research of three of the extant (and publicly available) Masoretic Hebrew Bible Codices and Manuscripts: Leningrad Codex, Koren Codex (currently Torah Only), and Miqra According to the Mesorah (MAM) Collection of Manuscripts (based on the Aleppo Codex).

All the research data and findings that are presented here – as well as that are made possible through the use of this software – are mathematically, statistically, as well as visually-verifiable.

0.1) Introduction

1.0) Download and Install Python

2.0) Download and/or “Pip Install” other necessary Python libraries: Numpy

2.1) Download and/or “Pip Install” other necessary Python libraries: Pandas

2.2) Download and/or “Pip Install” other necessary Python libraries: IPython (optional)

3.0) Download via GitHub “clone” command and/or via ZipFile from TorahBibleCodes’ GitHub

4.0) Open PowerShell CLI Console (SHIFT + Right Mouse Click) in the TorahBibleCodes Software Root Folder

5.0) Type `ipython` in the CLI to start IPython (Interactive Python CLI Console)

6.0) Type `%run p.py` to start the TorahBibleCodes ELS search program

7.0) Choose Codex (Collection of Manuscripts) to search

8.0) Koren Codex: Choose from the five (5) texts of Torah

9.0) Leningrad Codex: Choose from the 39 texts of Tanakh

10.0) MAM Collection of Manuscripts: Choose from the 39 texts of Tanakh

11.0) Choose Size of the 2D Matrix

12.0) Select Type of Input (Manual Type or Copy/Paste vs. Import from CSV file)

13.0) Manual Input: Choose the number of ELS Search Terms, e.g. 8

13.1) Manual Input: Type each ELS Search Term (with or without spaces ok), e.g. 8

13.2) Manual Input: Type each ELS Search Term (e.g. 8 terms): **המשיח, משיחי, משיחו, משיח, מי המשיח, מי משיחי, מי משיחו, מי משיח**

14.0) Import via CSV Input: Type the file name of the CSV file that contains the ELSs (with or without spaces ok)

14.1) CSV file that contains the ELSs must be within the same TorahBibleCodes folder

14.2) The CSV file can contain one or multiple (unlimited) ELSs in the first column (e.g. 8 terms): **המשיח, משיחי, משיחו, משיח, מי המשיח, מי משיחי, מי משיחו, מי משיח**

15.0) Choose Skip-Distance (d) range of (+1) to conduct traditional searches in the Bible texts, i.e. to search for words (or phrases with or without spaces) with an ELS Skip-Distance (d) of (+1), e.g. **המשיח, משיחי, אחרית הימים**

15.1) Torah: ELS Skip-Distance (d) of (+1), e.g. **המשיח, משיחי, אחרית הימים**

15.2) Torah: All Positive (+) ELS Matches with Skip-Distance (d) of (+1), e.g. **המשיח, משיחי, אחרית הימים**

16.0) Choose Skip-Distance (d) range to conduct ELS Search (unlimited range), e.g. (-100) to (+100)

16.1) TorahBibleCodes will search for the ELS terms and provide CSV data files

16.2) TorahBibleCodes will let you know when the search is complete

17.0) Your CSV files will be saved to the folder: **USER_GENERATED_FILES**

18.0) Sort files by **Name for easier organization and readability of files**

19.0) The **DATA SUMMARY file provides overall information about the ELS Searches – NOTE: TorahBibleCodes can be used as a Gematria calculator for words and phrases**

20.0) The **BY LETTER FIRST POSITIVE file provides “positive” (+) ELS matches by first letter in the positive (+) direction, i.e. counting forwards**

20.1) The **BY LETTER FIRST NEGATIVE file provides “negative” (-) ELS matches by first letter in the negative (-) direction, i.e. counting backwards**

20.2) The **BY LETTER LAST POSITIVE file provides “positive” (+) ELS matches by last letter in the positive (+) direction, i.e. counting forwards**

20.3) The **BY LETTER LAST NEGATIVE file provides “negative” (-) ELS matches by last letter in the negative (-) direction, i.e. counting backwards**

21.0) By default, the data is ordered in ascending order by **(n, d, k), specifically **Skip-Distance (d)****

21.1) To order by chronological letter/word and to visually & mathematically find ELS proximity, select the column **Found in Word Number in Text, and sort by ascending order**

21.2) Extend selection

21.3) Now your ELS Matches are sorted according to their chronological appearances as well as according to proximity by **Word Numbers (Word Coordinates)** and proximity by **Letter Numbers (Letter Coordinates)**

21.4) For example, these first two (2) ELSs (**מֶשִׁיחַ**) “*My Messiah*” and (**מֶשִׁיחָ**) “*Messiah*” are the first appearances of these two ELSs within the Torah, and are found in close proximity within Genesis 1:16, and indeed share four of the same letters, begin on the same letter (**מ/n**), and overlap with a Skip-Distance (*d*) of (+89).

21.5) These same two (2) ELSs (**מֶשִׁיחַ**) “*My Messiah*” and (**מֶשִׁיחָ**) “*Messiah*” also appear in their last instances of these two ELSs within the Torah, and are found in close proximity within Deuteronomy 33:9, and indeed again share four of the same letters, begin on the same letter (**מ/n**), and overlap with the same Skip-Distance (*d*) of (+89).

22.0) For each ELS Match, there corresponds an individual CSV file for that ELS Match

22.1) For ELS #6, ELS Match #498, with **(n, d, k) == (102774, 2, 7)**, **מֵי מֶשִׁיחַ**,

22.2) For ELS #8, ELS Match #499, with **(n, d, k) == (102774, 2, 6)**, **מֵי מֶשִׁיחָ**

23.0) For any text(s) selected, a CSV file is created with **Word Numbers (Word Coordinates)** and **Letter Positions (n)** as well as **Gematria Letter and Word Numerical Values**

23.1) Sort by **Gematria Word Total** to group words with the same **Gematria Numerical Value**

23.2) Extend selection

23.3) Now the words in the selected text(s) are sorted according to their **Gematria Word Total**

24.0) All ELS Matches can be plotted on a 2D Matrix CSV of any size according to the **5-Integer Tuple Letter Coordinates Mathematical ID system**; The **5th Integer (n)** is the **critical “primary key”**, and the same **Letter Position (n)** of **(n, d, k)**

24.1) For ELS #6, ELS Match #498, **(n, d, k) == (102774, 2, 7)**, **מֵי מֶשִׁיחַ**,
and ELS #8, ELS Match #499, **(n, d, k) == (102774, 2, 6)**, **מֵי מֶשִׁיחָ**,
2D Matrix of 20 x 15241

25.0) For any text(s) selected, a CSV file is created with Letter Statistics to allow calculation of statistical significance of ELS searches (i.e. requires knowledge of statistics to calculate according to your own definitions and algorithms for “statistical significance”)

26.0) The ELS Search Algorithm

27.0) Equidistant Letters Sequences (ELSSs)

28.0) The Ancient Method of Counting by Eye and Hand

29.0) Choosing a Size for the 2D Matrix

30.0) Unique VerseIDs, LetterIDs, WordIDs, and ELS IDs Coordinates System

30.1) (*Book#*, *Chapter#*, *Verse#*) – 3-Integer Tuple VerseID

30.2) (*Book#*, *Chapter#*, *Verse#*, *Letter#InVerse*) – 4-Integer LetterID

30.3) (*Book#*, *Chapter#*, *Verse#*, *Letter#InVerse*, *Letter#InText*) – 5-Integer LetterID

30.4) (*n*, *d*, *k*) – 3-Integer ELS ID

30.5) (*Book#*, *Chapter#*, *Verse#*, *Word#InVerse*) – 4-Integer WordID

30.6) (*Book#*, *Chapter#*, *Verse#*, *Word#InVerse*, *Word#InText*) – 5-Integer WordID

31.0) Plotting ELS Matches on a 2D Matrix

32.0) Appendix (Including Links to Software and Research Data)

32.1) Analysis (SUMMARY): Leningrad Codex Torah vs. Koren Codex Torah

32.2) Analysis: Leningrad Codex Genesis vs. Koren Codex Genesis

32.3) Analysis: Leningrad Codex Exodus vs. Koren Codex Exodus

32.4) Analysis: Leningrad Codex Leviticus vs. Koren Codex Leviticus

32.5) Analysis: Leningrad Codex Numbers vs. Koren Codex Numbers

32.6) Analysis: Leningrad Codex Deuteronomy vs. Koren Codex Deuteronomy

33.0) Author's Afterword

34.0) References

0.1) Introduction

TorahBibleCodes is free, open-source Python Bible Search and Research Software that enables serious, scientific (i.e. measurable, reproducible, and verifiable), mathematical (thus statistical), academic search and research of ancient Hebrew Masoretic biblical codices (e.g. Koren Codex, Leningrad Codex), texts (i.e. individual texts or groups of texts as sections within these codices), as well as collections of manuscripts (e.g. the Miqra according to the Masorah (MAM) based upon the Aleppo Codex).

The following is a user-guide intended to explain how to download, install, use, and utilize this software by presenting some of its features and functionalities.

It should be remembered that because this is free, open-source software (for non-commercial use, i.e. free for educational and/or personal or academic research purposes), it is possible to customize this software and develop new features if you have the Python Development and/or Mathematical & Statistical Inclination to imagine and develop them: please freely share your contributions and research with the world.

For example, if you would like an interactive GUI (e.g. via Python, React, Vue, Angular, etc.) in order to produce dynamic visualizations, and/or if would like to integrate this software into Artificial Intelligence (AI) applications (e.g. PyTorch, TensorFlow, OpenAI ChatGPT, etc.), it is entirely possible for you to develop it if you can conceive of the idea, cultivate your imagination, and visualize it into existence and fruition.

This current version 0.2 of TorahBibleCodes Bible Search and Research Software provides the mathematical/statistical measuring system with data points that encompass a unique, standardized mathematical ID system that makes possible dynamic interaction with the Python Data Objects via the IPython (Interactive Python) CLI Console. This enables the possibility to do scientific (i.e. reproducible and verifiable) research with a simple, intuitive, logical, human-friendly, mathematical ID system that is available to identify all individual books, chapters, verses, words, and letters of the Hebrew Bible in several codices.

This software has been developed to be available for free at no cost, and we develop this in our free time to share freely with the world as time allows. Your kind understanding and patience are appreciated.

If you enjoy the software and/or believe in this project and would like to help, you can also contact us (or just anonymously help) to financially support further R&D of this Bible Search and Research software, as well as consequential scientific, academic biblical research. Contributions are welcome and appreciated, as is sharing links to our websites TorahBibleCodes.com, GitHub.com/TorahBibleCodes, YouTube, Academia, ResearchGate, GiveSendGo, etc. on your social media. Every little bit, every post, and every share with sincere word-of-mouth recommendations helps us to give away this free, open-source software to more people, and thus freely share with the world.

<https://TorahBibleCodes.com> (or <http://TorahBibleCodes.com>)
<https://GitHub.com/TorahBibleCodes>

1.0) Download and Install Python: <https://www.python.org/downloads/>

The screenshot shows the Python.org website's download section. At the top, there are tabs for Python, PSF, Docs, PyPI, Jobs, and Community. Below the tabs is a search bar with a magnifying glass icon and a 'GO' button. A large banner features the Python logo and the text "Download the latest version for Windows". A call-to-action button says "Download Python 3.12.7". Below the banner, there are links for other operating systems: "Python for Windows, Linux/UNIX, macOS, Other" and "Want to help test development versions of Python 3.13? Prereleases, Docker images". To the right of the banner is a cartoon illustration of two boxes with yellow and white striped parachutes falling from the sky. Below the banner, there's a section titled "Active Python Releases" with a table showing maintenance status, first release date, end of support, and release schedule for Python 3.13 and 3.12. The table is as follows:

Python version	Maintenance status	First released	End of support	Release schedule
3.13	prerelease	2024-10-07 (planned)	2029-10	PEP 719
3.12	bugfix	2023-10-02	2028-10	PEP 693

At the bottom of the page, there's a search bar with placeholder text "Type here to search" and a taskbar with various application icons.

2.0) Download and/or “Pip Install” other necessary Python libraries: Numpy

The screenshot shows the NumPy installation guide. The top navigation bar includes links for Install, Documentation, Learn, Community, About Us, News, Contribute, and English. On the left, there's a sidebar with "On this page" sections for Recommendations, Beginning users, Advanced users, Python package management, Pip & conda, Reproducible installs, NumPy packages & accelerated linear algebra libraries, and Troubleshooting. The main content area starts with a heading "PIP" and a note: "If you use pip, you can install NumPy with: `pip install numpy`". It also mentions using a virtual environment. Below this is a large heading "Python and NumPy installation guide". A paragraph explains that managing packages in Python is complicated and provides recommendations. The "Beginning users" section follows, with a note about starting with recommendations based on experience level. At the bottom, there's a search bar and a taskbar with various application icons.

2.1) Download and/or “Pip Install” other necessary Python libraries: Pandas

The screenshot shows a web browser window with the URL pandas.pydata.org/docs/getting_started/install.html. The page is titled "Installing from PyPI". It provides instructions for installing pandas using pip:

```
pip install pandas
```

Note: You must have `pip>=19.3` to install from PyPI.

Note: It is recommended to install and run pandas from a virtual environment, for example, using the Python standard library's `venv`.

pandas can also be installed with sets of optional dependencies to enable certain functionality. For example, to install pandas with the optional dependencies to read Excel files:

```
pip install "pandas[excel]"
```

The full list of extras that can be installed can be found in the [dependency section](#).

Handling ImportErrors

2.2) Download and/or “Pip Install” other necessary Python libraries: IPython (optional)

The screenshot shows a web browser window with the URL ipython.org/install.html. The page is titled "I already have Python". It provides instructions for getting IPython if Python is already installed:

```
pip install ipython
```

I am getting started with Python

For new users who want to install a full Python environment for scientific computing and data science, we suggest installing the Anaconda or Canopy Python distributions, which provide Python, IPython and all of its dependencies as well as a complete set of open source packages for scientific computing and data science.

1. Download and install Continuum's [Anaconda](#) or the free edition of Enthought's [Canopy](#).
2. Update IPython to the current version using the Terminal:

Anaconda:

```
conda update conda  
conda update ipython
```

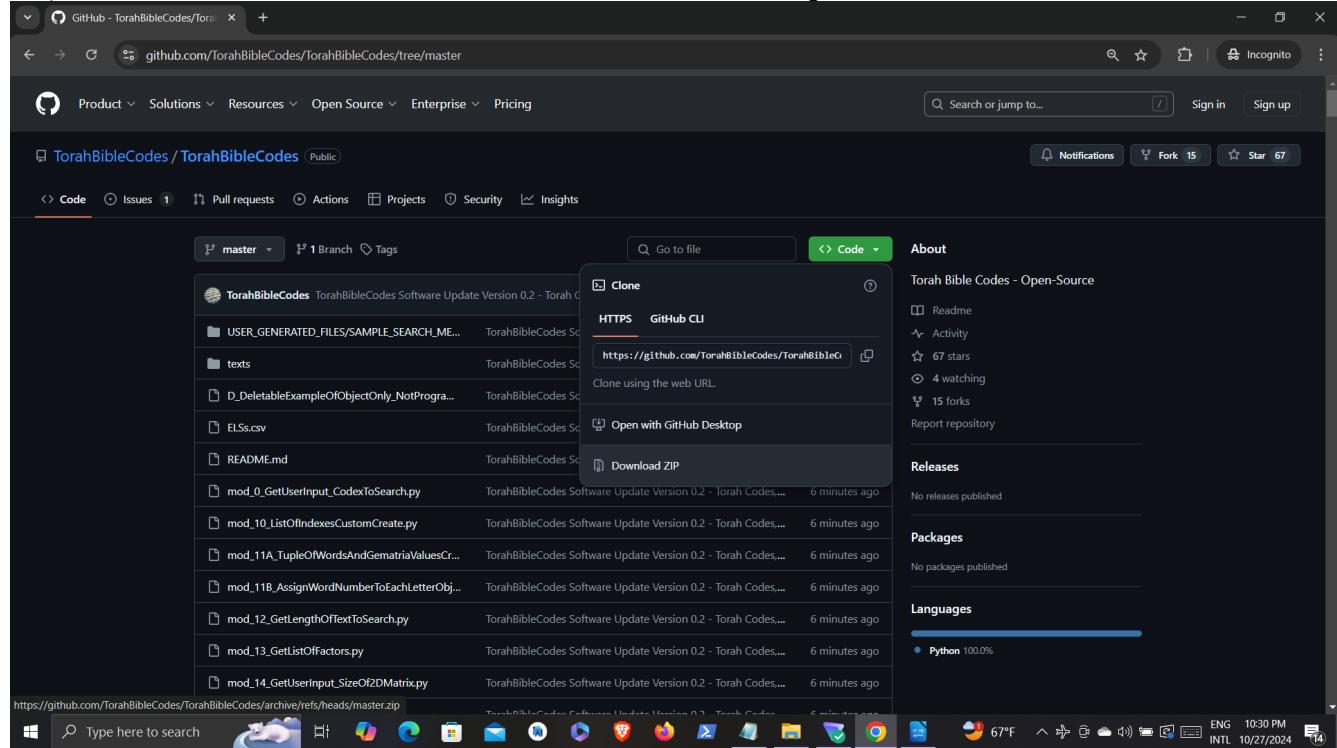
Enthought Canopy:

```
enpkg ipython
```

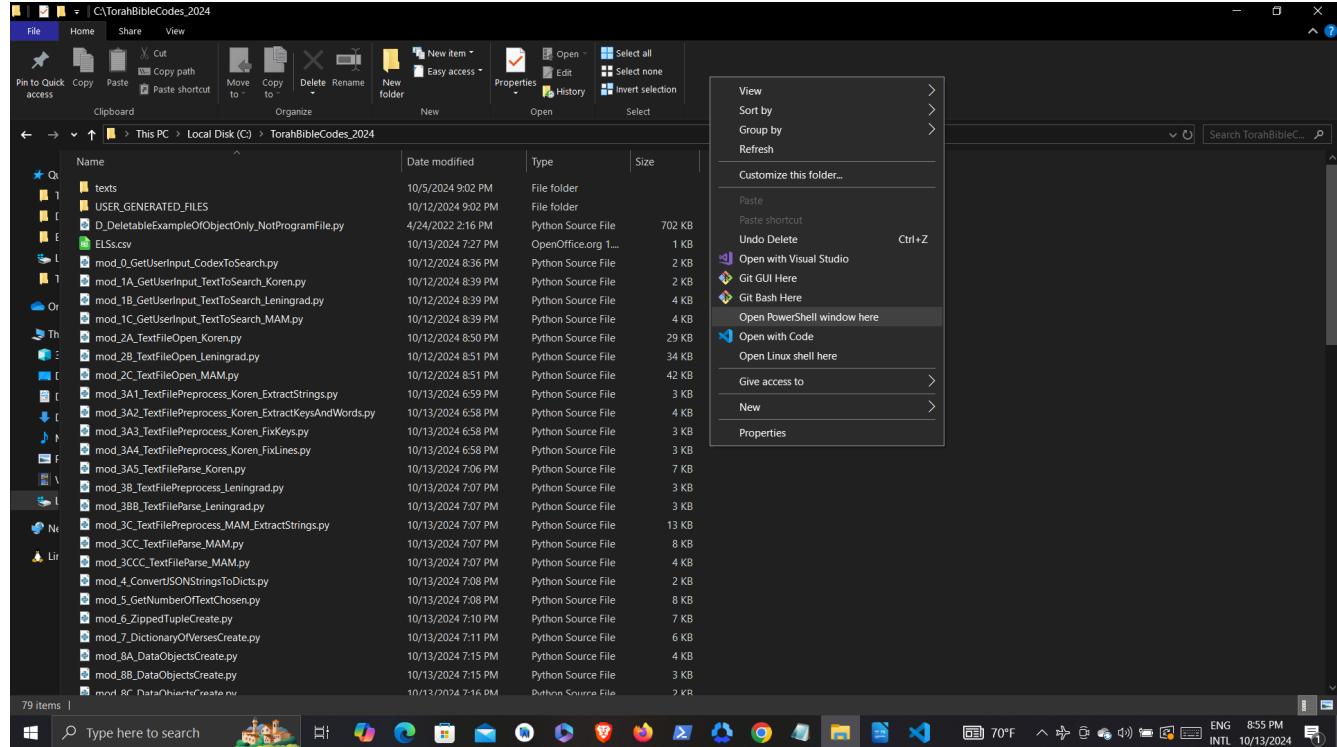
Downloads

You can manually download IPython from [GitHub](#) or [PyPI](#). To install one of these versions, unpack it and run the following from the top-level source directory using the Terminal:

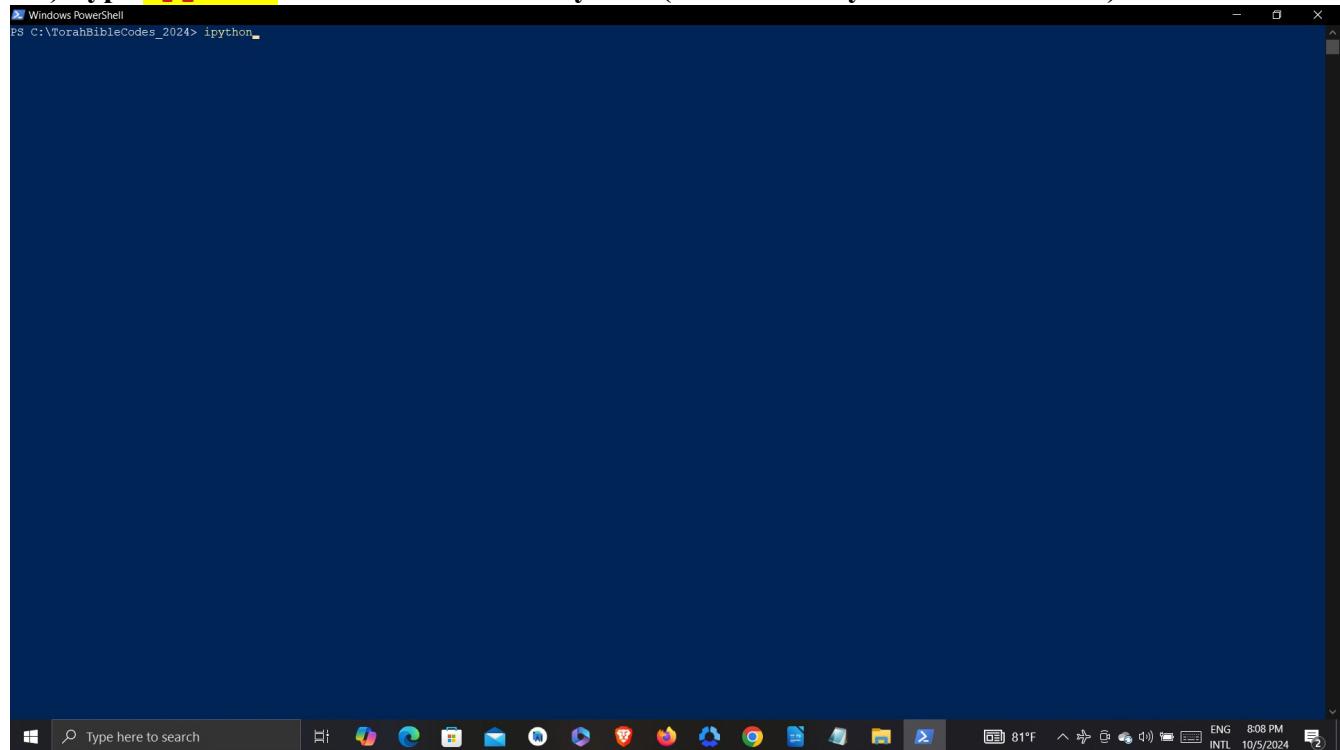
3.0) Download via GitHub “clone” command and/or via ZipFile from TorahBibleCodes’ GitHub



4.0) Open PowerShell CLI Console (SHIFT + Right Mouse Click) in the TorahBibleCodes Software Root Folder

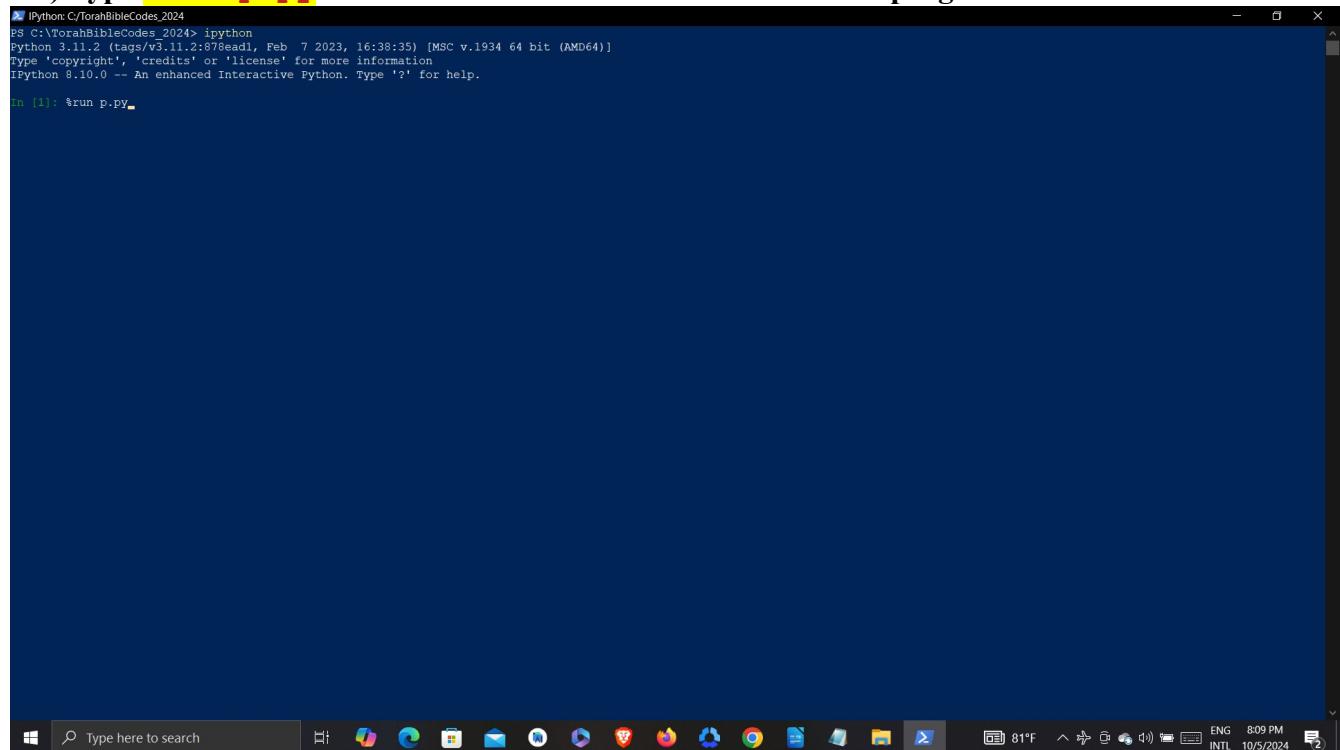


5.0) Type **ipython** in the CLI to start IPython (Interactive Python CLI Console)



```
PS C:\TorahBibleCodes_2024> ipython
```

6.0) Type **%run p.py** to start the TorahBibleCodes ELS search program



```
PS C:\TorahBibleCodes_2024> ipython
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.10.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: %run p.py
```

7.0) Choose Codex (Collection of Manuscripts) to search

```
IPython: C:\TorahBibleCodes_2024> ipython
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.10.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: %run p.py

WITHIN FUNCTION: BEGIN FUNCTION #0 - GET USER INPUT; CHOOSE CODEX TO SEARCH; ## RETURNS INTEGER

Please select Hebrew Bible codex to search:

1 - Koren Codex - Claremont Michigan Transliteration: [Torah: 304805]

2 - Leningrad Codex: [Torah: 304850; Prophets: 553785; Writings: 338407; Tanach: 1197042]

3 - Miqra According to the Masorah (MAM) Collection of Manuscripts: [Torah: 304801; Prophets: 553698; Writings: 338340; Tanach: 1196839]

Please select codex to search (1: Koren; 2: Leningrad; 3: MAM):
```

8.0) Koren Codex: Choose from the five (5) texts of Torah

```
IPython: C:\TorahBibleCodes_2024>

WITHIN FUNCTION: BEGIN FUNCTION #1A - GET USER INPUT; CHOOSE TEXT TO SEARCH

Please select text to search in the Koren Codex:

1 - Genesis - 78064 letters
2 - Exodus - 63529 letters
3 - Leviticus - 44790 letters
4 - Numbers - 63530 letters
5 - Deuteronomy - 54892 letters
40 - Pentateuch (Torah) - 304805 letters

Please select text to search:
```

9.0) Leningrad Codex: Choose from the 39 texts of Tanakh

```
python C:/ToraBibleCodes_2024

WITHIN FUNCTION: BEGIN FUNCTION #1B - GET USER INPUT: CHOOSE TEXT TO SEARCH

Please select text to search in the Leningrad Codex:

1 - Genesis - 78069 letters
2 - Exodus - 63531 letters
3 - Leviticus - 47103 letters
4 - Numbers - 63165 letters
5 - Deuteronomy - 54910 letters
6 - Joshua - 39807 letters
7 - Judges - 38944 letters
8 - Samuel - 51510 letters
9 - II Samuel - 42178 letters
10 - I Kings - 50623 letters
11 - II Kings - 47837 letters
12 - Isaiah - 66888 letters
13 - Jeremiah - 50101 letters
14 - Ezekiel - 74499 letters
15 - Hosea - 9385 letters
16 - Joel - 3972 letters
17 - Amos - 8033 letters
18 - Obadiah - 5110 letters
19 - Jonah - 2700 letters
20 - Micah - 5570 letters
21 - Nahum - 4252 letters
22 - Habakkuk - 2994 letters
23 - Zephaniah - 2896 letters
24 - Haggai - 2336 letters
25 - Zechariah - 12432 letters
26 - Malachi - 3450 letters
27 - Psalms - 78838 letters
28 - Proverbs - 23507 letters
29 - Job - 31862 letters
30 - Song of Songs - 5151 letters
31 - Ruth - 4947 letters
32 - Lamentations - 10969 letters
33 - Ecclesiastes - 10969 letters
34 - Esther - 12112 letters
35 - Daniel - 24291 letters
36 - Ezra - 19762 letters
37 - Nehemiah - 32133 letters
38 - I Chronicles - 44558 letters
39 - II Chronicles - 54920 letters

40 - Pentateuch (Torah) - 304850 letters
41 - Prophets (Nevi'im) - 553785 letters
42 - Writings (K'tuvim) - 338407 letters
43 - Hebrew Bible (Tanach) - 1197042 letters

Please select text to search:
```



10.0) MAM Collection of Manuscripts: Choose from the 39 texts of Tanakh

```
python C:/ToraBibleCodes_2024

WITHIN FUNCTION: BEGIN FUNCTION #1C - GET USER INPUT: CHOOSE TEXT TO SEARCH

Please select text to search in the Migra According to the Masorah (MAM) Collection of Manuscripts:

1 - Genesis - 78063 letters
2 - Exodus - 63527 letters
3 - Leviticus - 47103 letters
4 - Numbers - 63169 letters
5 - Deuteronomy - 54952 letters
6 - Joshua - 39730 letters
7 - Judges - 38952 letters
8 - Samuel - 51510 letters
9 - II Samuel - 42179 letters
10 - I Kings - 50625 letters
11 - II Kings - 47822 letters
12 - Isaiah - 66874 letters
13 - Jeremiah - 50103 letters
14 - Ezekiel - 74510 letters
15 - Hosea - 9389 letters
16 - Joel - 3972 letters
17 - Amos - 8033 letters
18 - Obadiah - 5119 letters
19 - Jonah - 2700 letters
20 - Micah - 5571 letters
21 - Nahum - 4255 letters
22 - Habakkuk - 2995 letters
23 - Zephaniah - 2895 letters
24 - Haggai - 2336 letters
25 - Zechariah - 12433 letters
26 - Malachi - 3450 letters
27 - Psalms - 78839 letters
28 - Proverbs - 24500 letters
29 - Job - 31851 letters
30 - Song of Songs - 5141 letters
31 - Ruth - 4949 letters
32 - Lamentations - 10969 letters
33 - Ecclesiastes - 10968 letters
34 - Esther - 12110 letters
35 - Daniel - 24280 letters
36 - Ezra - 19763 letters
37 - Nehemiah - 32367 letters
38 - I Chronicles - 44559 letters
39 - II Chronicles - 54917 letters

40 - Pentateuch (Torah) - 304801 letters
41 - Prophets (Nevi'im) - 553693 letters
42 - Writings (K'tuvim) - 338340 letters
43 - Hebrew Bible (Tanach) - 1196939 letters

Please select text to search:
```



11.0) Choose Size of the 2D Matrix

```
IPython: C/TorahBibleCodes_2024
WITHIN FUNCTION: BEGIN FUNCTION #14 - GET USER INPUT: SIZE OF 2D MATRIX:

LengthOfTextToSearch = 304805

List of 4 Factors from the text(s) that you selected: [1, 5, 60961, 304805]

Choose a size for the 2D matrix from a number from the List of 4 Factors (i.e. # of x columns to output for the 2D Matrix)
OR Choose ANY number, and the 2D Matrix will be calculated according to your selection.

NOTE: Larger numbers above 800 for x / Width / Columns may (or may not) exceed the maximum allowed by Microsoft Office (Excel), LibreOffice, Apache Open Office, etc.

Type the number here: -
```



12.0) Select Type of Input (Manual Type or Copy/Paste vs. Import from CSV file)

```
IPython: C/TorahBibleCodes_2024
SpaceToFillInLastRow 15
NewLengthOfTextToOutput 304820
YH 15241
XW 20

Length of LLL 304820

WITHIN FUNCTION: END FUNCTION #15 - CALCULATE XW / YH

WITHIN FUNCTION: BEGIN FUNCTION #16A - GET USER INPUT: TYPE OF SEARCH INPUT ## RETURNS TypeOfSearchInput;

Please select the type of input for ELS Search Terms:

1 - Manual Input of ELS Search Term(s) via Keyboard (i.e. type OR copy/paste Hebrew one-by-one)
2 - Import Multiple ELS Search Term(s) from CSV file (i.e. list of one OR more Hebrew search terms in CSV file)

Please select type of input: -
```



13.0) Manual Input: Choose the number of ELS Search Terms, e.g. 8

```
IPython: C/TorahBibleCodes_2024
Please select type of input: 1

You have chosen input type number # 1.

WITHIN FUNCTION: END FUNCTION #16A - GET USER INPUT: TYPE OF SEARCH INPUT ## RETURNS TypeOfSearchInput;
Input Type: Manual by Keyboard

WITHIN FUNCTION: BEGIN FUNCTION #16 - GET USER INPUT - NUMBER OF SEARCH TERMS;

How many total number of ELS Search-Terms would you like to search for within the selected text?:
8

8
```



13.1) Manual Input: Type each ELS Search Term (with or without spaces ok), e.g. 8

```
IPython: C/TorahBibleCodes_2024
WITHIN FUNCTION: BEGIN FUNCTION #16 - GET USER INPUT - NUMBER OF SEARCH TERMS;

How many total number of ELS Search-Terms would you like to search for within the selected text?:
8

8

You have chosen to search for 8 ELS Search Terms

WITHIN FUNCTION: END FUNCTION #16 - GET USER INPUT - NUMBER OF SEARCH TERMS;

WITHIN FUNCTION: BEGIN FUNCTION #17A - GET USER INPUT - ELS SEARCH TERMS;

Number of ELS Search Terms: 8
Please input each ELS Search Term (in Hebrew) for the 8 terms that you specified

ELS Search Term 1 (type OR copy/paste Hebrew): ■
```



13.2) Manual Input: Type each ELS Search Term (e.g. 8 terms): המשיח, מישיחו, משיח, מי המשיח, מי מישיחו, מי משיחו, מי משיח

```
IPython: C/TorahBibleCodes.2024
You have chosen to search for 8 ELS Search Terms

WITHIN FUNCTION: END FUNCTION #16 - GET USER INPUT - NUMBER OF SEARCH TERMS;

WITHIN FUNCTION: BEGIN FUNCTION #17A - GET USER INPUT - ELS SEARCH TERMS;

Number of ELS Search Terms: 8
Please input each ELS Search Term (in Hebrew) for the 8 terms that you specified

ELS Search Term 1 (type OR copy/paste Hebrew): יְהוָה
ELS Search Term (type OR copy/paste Hebrew): יְהוָה
ELS Search Term (R-T-L): יְהוָה

ELS Search Term 2 (type OR copy/paste Hebrew): יְהוָה
ELS Search Term (R-T-L): יְהוָה

ELS Search Term 3 (type OR copy/paste Hebrew): יְהוָה
```

14.0) Import via CSV Input: Type the file name of the CSV file that contains the ELSs (with or without spaces ok)

```
IPython: C/TorahBibleCodes.2024
WITHIN FUNCTION: BEGIN FUNCTION #16A - GET USER INPUT: TYPE OF SEARCH INPUT ## RETURNS TypeOfSearchInput;

Please select the type of input for ELS Search Terms:

1 - Manual Input of ELS Search Term(s) via keyboard (i.e. type OR copy/paste Hebrew one-by-one)
2 - Import Multiple ELS Search Term(s) from CSV file (i.e. list of one OR more Hebrew search terms in CSV file)

Please select type of input: 2

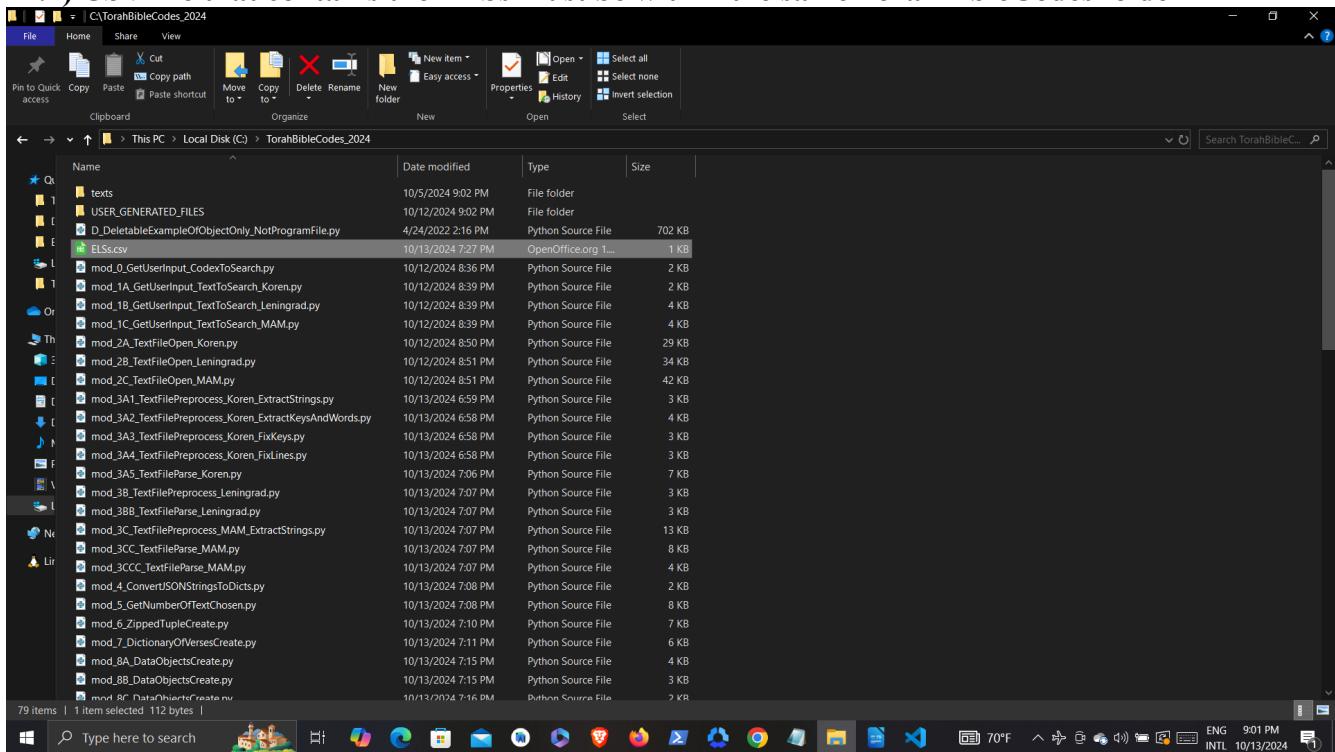
You have chosen input type number # 2.

WITHIN FUNCTION: END FUNCTION #16A - GET USER INPUT: TYPE OF SEARCH INPUT ## RETURNS TypeOfSearchInput;
Input Type: CSV Import

WITHIN FUNCTION: BEGIN FUNCTION #16AA - GET USER INPUT: CSV FILE NAME FOR SEARCH INPUT ## RETURNS ;

Please input the CSV File Name (in the root directory where you are running this Torah Bible Codes search program) that contains your ELS Search Terms (e.g. FileName4ELSS.csv): ELSS.csv
```

14.1) CSV file that contains the ELSs must be within the same TorahBibleCodes folder



14.2) The CSV file can contain one or multiple (unlimited) ELSs in the first column (e.g. 8 terms): המשיח, משיחי, משיחו, משיח, מי המשיח, מי משיח, מי משיחו, מי משיחי

The screenshot shows a LibreOffice Calc spreadsheet with the following data in column A:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	המשיח																
2	משיחי																
3	משיחו																
4	משיח																
5	מי המשיח																
6	מי משיח																
7	מי משיחו																
8	מי משיחי																

The bottom status bar indicates: Sheet 1 of 1, Selected: 1,048,576 rows, 1 column, Default, English (USA), Average: ; Sum: 0, ENG 9:01 PM, 10/13/2024.

15.0) Choose Skip-Distance (d) range of (+1) to conduct traditional searches in the Bible texts, i.e. to search for words (or phrases with or without spaces) with an ELS Skip-Distance (d) of (+1), e.g. המשיח, משיחי, אחרית הימים

```

ELS Search Term (type OR copy/paste Hebrew): אחורית הימים
ELS Search Term (R-T-L): ימיהם אחורית

You have entered the following ELS search terms (NO SPACES - IF ANY): ['סימיה מיראא', 'ויזטט', 'ויזטט']
(1: 'הנאה', 2: 'ויזטט', 3: 'סימיה מיראא')

You have entered the following ELS search terms (WITH SPACES - IF ANY): ['סימיה מיראא', 'ויזטט', 'ויזטט']
(1: 'הנאה', 2: 'ויזטט', 3: 'סימיה מיראא')

WITHIN FUNCTION: END FUNCTION #17A - GET USER INPUT - ELS SEARCH TERMS;

WITHIN FUNCTION: BEGIN FUNCTION #17B - GET USER INPUT - SKIP DISTANCES D MINIMUM / MAXIMUM;

What is the lowest/smallest (NEGATIVE or POSITIVE) skip distance (d) you would like to use for this ELS search? (e.g. -100, -50, -10, -1, 1, 10, 50, 100):
1

What is the highest/largest (NEGATIVE or POSITIVE) skip distance (d) you would like to use for this ELS search? (e.g. -100, -50, -10, -1, 1, 10, 50, 100):
1

```

15.1) Torah: ELS Skip-Distance (d) of (+1), e.g. המשיח, משיחי, אחרית הימים

A	B	C	D	E	F	G	H
ELS Search Term	(ELS Search Term #, [Gematria LetterValues], Gematria WordTotal)	Gematria WordTotal	# Positive ELS Matches	# Negative ELS Matches			
1 המשיח(1, [5, 40, 300, 10, 8], 363)		363	4	0			
2 משיחי(2, [40, 300, 10, 8, 10], 368)		368	1	0			
3 אחרית הימים(3, [1, 8, 200, 10, 400, 5, 10, 40, 10, 40], 724)		724	4	0			
4							
5							
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							
20							
21							
22							
23							
24							
25							
26							
27							
28							
29							

המשיח, משיח', א' המשיח, אחרית הימים

The screenshot shows a LibreOffice Calc spreadsheet titled "USER_FILE_WordsOfELSSs.ELSMatches_BY LETTER_FIRST_POSITIVE_Koren_40TORAH.Instruction_20x15241.csv". The table has columns: A (Index), B (Word), C (Total of ELS), D (ELS Search Term), E (WordNumber In Text), F (WordCoordinatesDW), G (Found in Word In Text), H (LetterPositionIndex In Word), I (LetterCoordinatesDSK of First). The data includes rows for 'משיח' and 'אחרית הימים' with various coordinates.

D1	A	B	C	D	E	F	G	H	I
	(n, d, k)	[Gematria of ELS]	Gematria Word Total of ELS	ELS Search Term	Found in WordNumber In Text	WordCoordinatesDW	Found in Word In Text	LetterPositionIndex In Word	LetterCoordinatesDSK of First
1	(n, d, k)	[Gematria of ELS]	363	משיח	38052(3, 4, 3, 3, 38052)		משיח	1(3, 4, 3, 7, 144370)	
2	(144370, 1, 5)	[5, 40, 300, 10, 8]	363	משיח	38089(3, 4, 5, 3, 38089)		משיח	1(3, 4, 5, 9, 144409)	
3	(144409, 1, 5)	[5, 40, 300, 10, 8]	363	משיח	38270(3, 4, 16, 3, 38270)		משיח	1(3, 4, 16, 10, 145150)	
4	(145150, 1, 5)	[5, 40, 300, 10, 8]	363	משיח	39199(3, 6, 15, 2, 39199)		משיח	1(3, 6, 15, 6, 148580)	
5	(148589, 1, 5)	[5, 40, 300, 10, 8]	363	משיח	38052(3, 4, 3, 3, 38052)		משיח	2(3, 4, 3, 7, 144371)	
6	(144371, 1, 5)	[40, 300, 10, 8, 10]	368	משיח	19882(1, 49, 1, 13, 19882)		משיח	2(1, 49, 1, 49, 75284)	
7	(75284, 1, 10)	[1, 8, 200, 10, 400, 5, 10, 40, 10, 40]	724	אחרית הימים	60615(4, 24, 14, 12, 60615)		אחרית הימים	2(4, 24, 14, 43, 229716)	
8	(229716, 1, 10)	[1, 8, 200, 10, 400, 5, 10, 40, 10, 40]	724	אחרית הימים	67835(5, 4, 30, 7, 67835)		אחרית הימים	2(5, 4, 30, 25, 258236)	
9	(258236, 1, 10)	[1, 8, 200, 10, 400, 5, 10, 40, 10, 40]	724	אחרית הימים	78828(5, 31, 29, 17, 78828)		אחרית הימים	2(5, 31, 29, 65, 300401)	
10	(300401, 1, 10)	[1, 8, 200, 10, 400, 5, 10, 40, 10, 40]	724	אחרית הימים					
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									
29									
30									
31									
32									
33									
34									

16.0) Choose Skip-Distance (d) range to conduct ELS Search (unlimited range), e.g. (-100) to (+100)

```

Python: C:/torahlibCodes_2024
ELS Search Term (type OR copy/paste Hebrew): ספינה תיראה
ELS Search Term (R-T-L): ספינה תיראה
You have entered the following ELS search terms (NO SPACES - IF ANY):   [спинна тираха, 'спинна тираха', 'спинна тираха'] <class 'list'>
[1: 'спинна тираха', 2: 'спинна тираха', 3: 'спинна тираха']

You have entered the following ELS search terms (WITH SPACES - IF ANY):   [спинна тираха', 'спинна тираха', 'спинна тираха'] <class 'list'>
[1: 'спинна тираха', 2: 'спинна тираха', 3: 'спинна тираха']

WITHIN FUNCTION: END FUNCTION #17A - GET USER INPUT - ELS SEARCH TERMS;

WITHIN FUNCTION: BEGIN FUNCTION #17B - GET USER INPUT - SKIP DISTANCES D MINIMUM / MAXIMUM;

What is the lowest/smallest (NEGATIVE or POSITIVE) skip distance (d) you would like to use for this ELS search? (e.g. -100, -50, -10, -1, 1, 10, 50, 100): -100

What is the highest/largest (NEGATIVE or POSITIVE) skip distance (d) you would like to use for this ELS search? (e.g. -100, -50, -10, -1, 1, 10, 50, 100): 100

```

16.1) TorahBibleCodes will search for the ELS terms and provide CSV data files

```
IPython: C/TorahBibleCodes_2024
WITHIN FUNCTION: END FUNCTION #10 - LIST OF INDEXES (CUSTOM) CREATE

WITHIN FUNCTION: BEGIN FUNCTION #10 - LIST OF INDEXES (CUSTOM) CREATE

WITHIN FUNCTION: END FUNCTION #10 - LIST OF INDEXES (CUSTOM) CREATE

WITHIN FUNCTION: BEGIN FUNCTION #41 - SEARCH FOR ELS LETTER MATCHES VIA PANDAS SERIES;
SEARCH PROGRESS: 100%|██████████| 3/3 [00:01<00:00,  1.76Search-Term/s]

WITHIN FUNCTION: END FUNCTION #41 - SEARCH FOR ELS LETTER MATCHES VIA PANDAS SERIES;

WITHIN FUNCTION: BEGIN FUNCTION #21 - PANDAS OBJECTS CREATE

WITHIN FUNCTION: END FUNCTION #21 - PANDAS OBJECTS CREATE

WITHIN FUNCTION: BEGIN FUNCTION #22A - ELS SEARCH BY FIRST LETTER - FORWARD SEARCH: POSITIVE DIRECTION;

1      3
2      4
3      N
4      B
5      Y
..
304801  Y
304802  B
304803  T
304804  N
304805  Y
Length: 304805, dtype: object

Please wait while your ELS Search is conducted...
SEARCH PROGRESS: 67%|██████████| 2/3 [01:34<00:47, 47.02s/Search-Term] 84°F  עבר 10/5/2024 8:42 PM
```

16.2) TorahBibleCodes will let you know when the search is complete

```
IPython: C/TorahBibleCodes_2024
WITHIN FUNCTION: END FUNCTION #98 - FILE NAMES CREATE FOR ELS TERMS - NEGATIVE

WITHIN FUNCTION: BEGIN FUNCTION #99 - ITERATE OUTPUT FOR ELS MATCHES ALL WORD AND LETTER DATA

WITHIN FUNCTION: END FUNCTION #99 - ITERATE OUTPUT FOR ELS MATCHES ALL WORD AND LETTER DATA

WITHIN FUNCTION: BEGIN FUNCTION #99 - ITERATE OUTPUT FOR ELS MATCHES ALL WORD AND LETTER DATA

WITHIN FUNCTION: END FUNCTION #99 - ITERATE OUTPUT FOR ELS MATCHES ALL WORD AND LETTER DATA

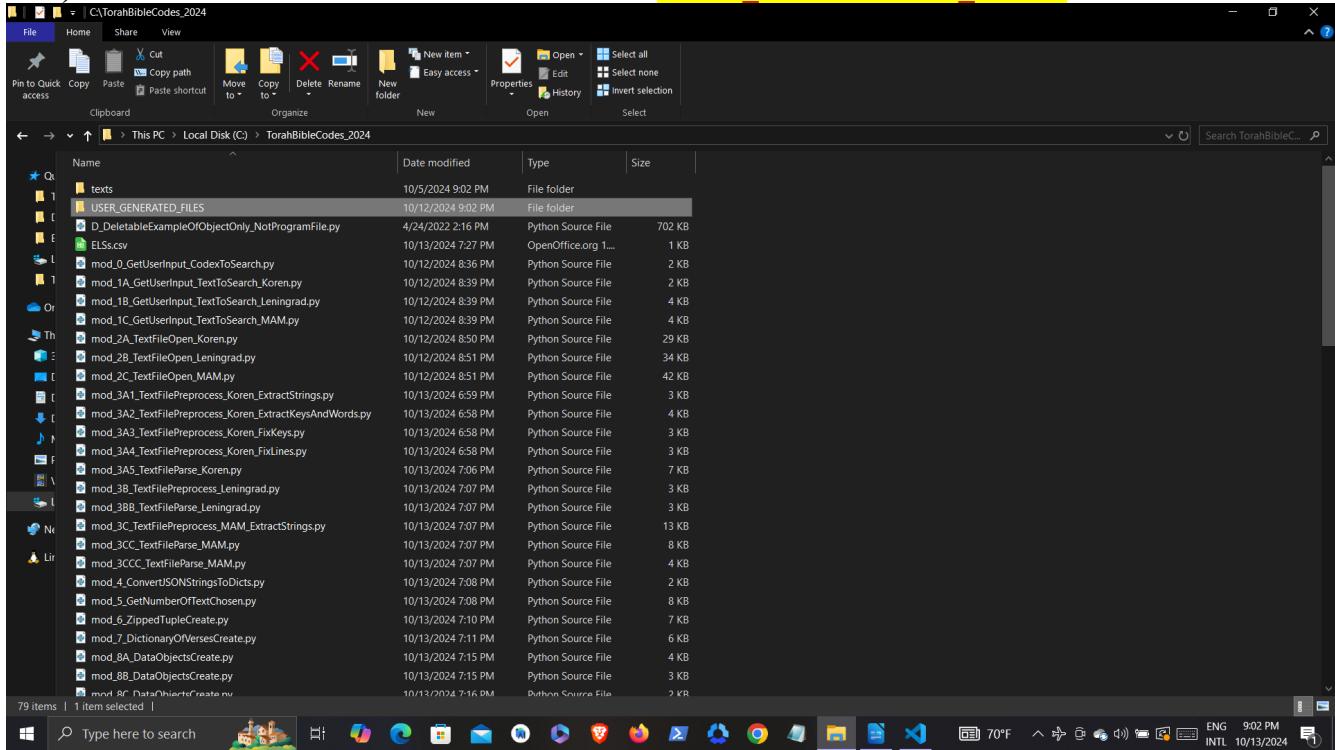
Length of List of Letters of Selected Text : 304805
Length of List of SPACES of 2D MATRIX CSV FILE : 304820

Length of Dictionary : 3

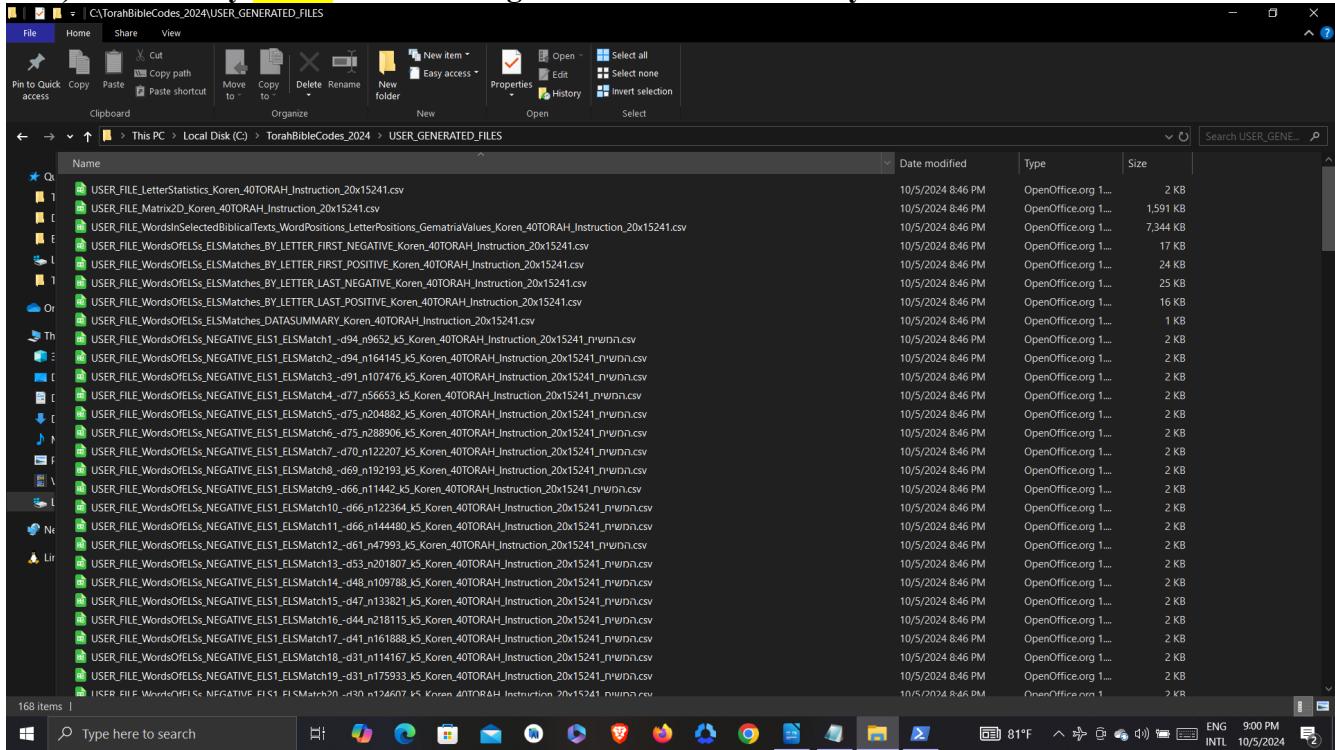
GAME OVER; NO REDOS; END OF GAME FOREVER!
GAME OVER; ...REALLY ...THIS TIME IT'S OVER.

In [3]:
```

17.0) Your CSV files will be saved to the folder: **USER_GENERATED_FILES**



18.0) Sort files by **Name** for easier organization and readability of files



20.3) The **BY LETTER LAST NEGATIVE** file provides “negative” (-) ELS matches by last letter in the negative (-) direction, i.e. counting backwards

Screenshot of LibreOffice Calc showing the content of the file `USER_FILE_WordsOfELSSs.ELSMatches_BY LETTER LAST NEGATIVE_Koren_40TORAH.Instruction_20x15241.csv`.

The table has columns: A (Row Number), B (Letter), C (Word), D (Search Term), E (Found in WordLister In Text), F (WordCoordinatesDW Found in Text), G (LetterCoordinatesDW Found in Text), H (LetterCoordinatesDW Found in Word), I (LetterCoordinatesDW Found of First Letter in ELS), J (Found in Verse).

The data shows various Hebrew words and their ELS matches, ordered by letter and search term.

21.0) By default, the data is ordered in ascending order by (n, d, k) , specifically Skip-Distance (d)

Screenshot of LibreOffice Calc showing the content of the file `USER_FILE_WordsOfELSSs.ELSMatches_BY LETTER FIRST POSITIVE_Koren_40TORAH.Instruction_20x15241.csv`.

The table has columns: A (Row Number), B (Letter), C (Word), D (Search Term), E (Found in WordLister In Text), F (WordCoordinatesDW Found in Text), G (LetterCoordinatesDW Found in Text), H (LetterCoordinatesDW Found in Word), I (LetterCoordinatesDW Found of First Letter in ELS), J (Found in Verse).

The data shows various Hebrew words and their ELS matches, ordered by letter and search term.

21.1) To order by chronological letter/word and to visually & mathematically find ELS proximity, select the column **Found in Word Number in Text**, and sort by ascending order

The screenshot shows a LibreOffice Calc spreadsheet with the following columns:

- A: General
- B: General Word Total of ELS
- C: ELS Search Term
- D: Found in Word Number in Text
- E: WordsProximityDW
- F: Found in Word in Text
- G: LetterPositionIndex
- H: Word
- I: LetterCoordinatesDX of First Letter in ELS
- J: Found in Verse

The 'D' column contains the sorted list of words. A 'Sort Ascending' button is visible at the top of the column.

21.2) Extend selection

The screenshot shows a LibreOffice Calc spreadsheet with the following columns:

- A: General
- B: General Word Total of ELS
- C: ELS Search Term
- D: Found in Word Number in Text
- E: WordsProximityDW
- F: Found in Word in Text
- G: LetterPositionIndex
- H: Word
- I: LetterCoordinatesDX of First Letter in ELS
- J: Found in Verse

A 'Sort Range' dialog box is open over the spreadsheet, containing three buttons: 'Extend selection', 'Current selection', and 'Cancel'. The text inside the dialog box reads: "The cells next to the current selection also contain data. Do you want to extend the sort range to A1:J502, or sort the currently selected range, E1:1048576?".

21.3) Now your ELS Matches are sorted according to their chronological appearances as well as according to proximity by Word Numbers (Word Coordinates) and proximity by Letter Numbers (Letter Coordinates)

Screenshot of LibreOffice Calc showing the results of the search for ELS matches in the Koren 40TORAH instruction. The table has 15 columns:

- A: Gematria Value
- B: Gematria Word Total of ELS
- C: ELS Search Term
- D: Found in Verse
- E: WordNumber
- F: Text
- G: LetterCoordinatesDW
- H: Found in Word
- I: Text
- J: LetterPositionIndex
- K: In Word
- L: LetterCoordinatesOK of First Letter in ELS
- M: Found in Verse
- N: Additional Notes

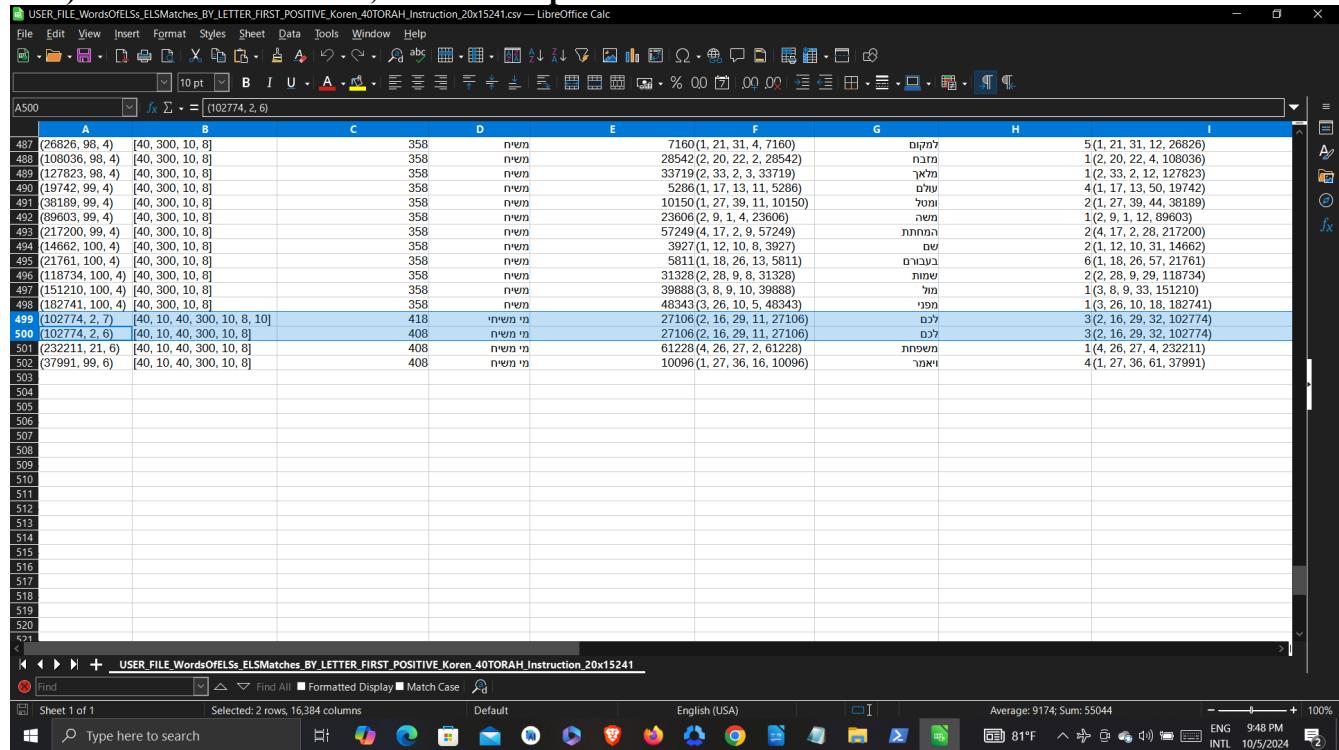
The table contains approximately 1,048,576 rows of data. The data shows various Hebrew words and their corresponding ELS matches, along with their positions and coordinates.

21.4) For example, these first two (2) ELSs (משיחי “My Messiah” and משיח “Messiah” are the first appearances of these two ELSs within the Torah, and are found in close proximity within Genesis 1:16, and indeed share four of the same letters, begin on the same letter (ם/n), and overlap with a Skip-Distance (d) of (+89).

21.5) These same two (2) ELSs (משיחי) “*My Messiah*” and (משיח) “*Messiah*” also appear in their last instances of these two ELSs within the Torah, and are found in close proximity within Deuteronomy 33:9, and indeed again share four of the same letters, begin on the same letter (ב/נ), and overlap with the same Skip-Distance (d) of (+89).

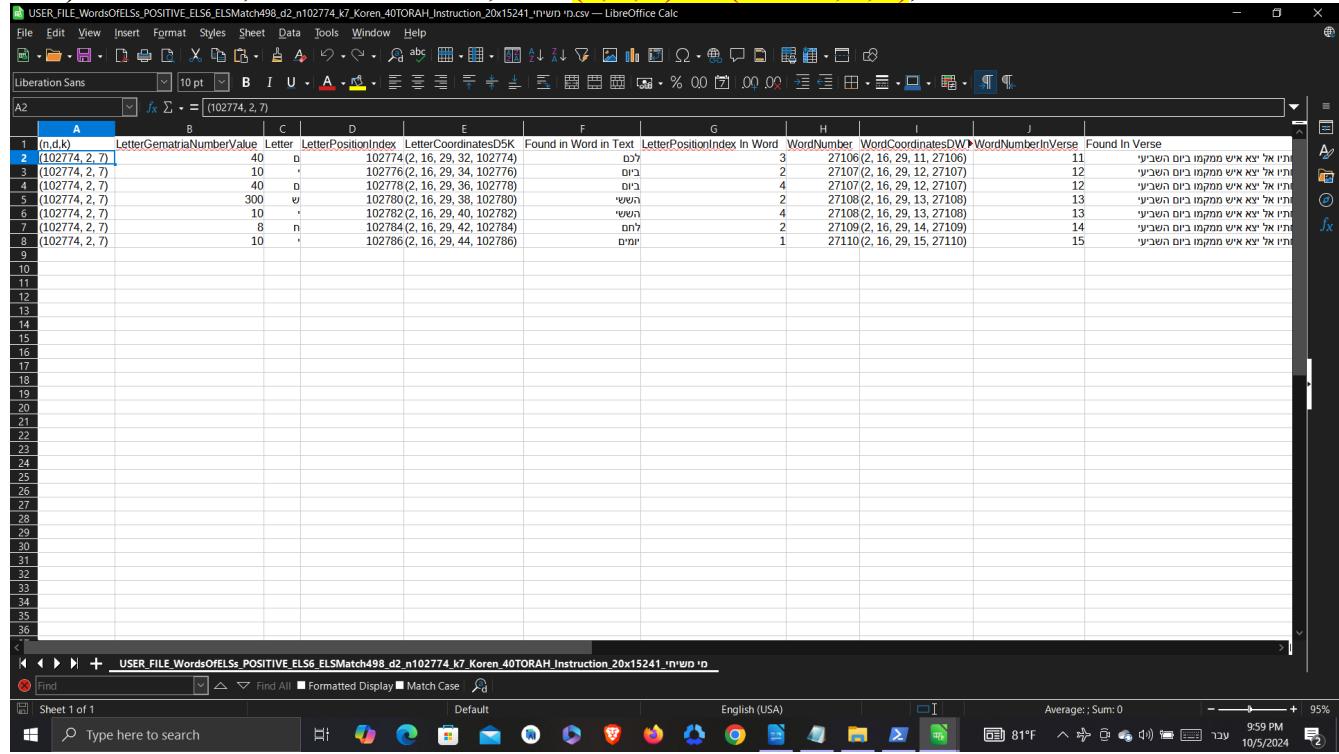
A	B	C	D	E	F	G	H	I
487	(289101, 2, 4)	[40, 300, 10, 8]		358 משיח	75925(5, 26, 2, 18, 75925)	תמיון	5(5, 26, 2, 70, 289101)	
488	(292396, 36, 4)	[40, 300, 10, 8]		358 משיח	76769(5, 28, 17, 3, 76769)	ופארון	2(5, 28, 17, 10, 292396)	
489	(292882, 64, 4)	[40, 300, 10, 8]		358 משיח	76891(5, 28, 27, 5, 76891)	וביפוי	7(5, 28, 27, 25, 292882)	
490	(298099, 52, 4)	[40, 300, 10, 8]		358 משיח	77114(5, 29, 14, 8, 77114)	הון	4(5, 29, 14, 24, 298099)	
491	(299025, 2, 4)	[40, 300, 10, 8]		358 משיח	78465(5, 31, 11, 10, 78465)	תגלות	5(5, 31, 11, 35, 299025)	
492	(300166, 5, 5)	[5, 40, 300, 10, 8]		363 משיחי	78787(5, 31, 26, 11, 78787)	יראות	2(5, 31, 26, 11, 300166)	
493	(300225, 59, 4)	[40, 300, 10, 8]		358 משיח	78794(5, 31, 27, 12, 78794)	ענינים	2(5, 31, 27, 40, 300225)	
494	(301330, 58, 5)	[40, 300, 10, 8, 10]		368 משיחי	79062(5, 32, 20, 15, 79062)	ם	2(5, 32, 20, 53, 301330)	
495	(301330, 59, 4)	[40, 300, 10, 8]		358 משיח	79062(5, 32, 20, 15, 79062)	ם	2(5, 32, 20, 53, 301330)	
496	(301853, 58, 4)	[40, 300, 10, 8]		358 משיח	79149(5, 32, 28, 5, 79149)	דר	2(5, 32, 28, 14, 301853)	
497	(301773, 16, 4)	[40, 300, 10, 8]		358 משיח	79177(5, 32, 31, 6, 79177)	פלילין	6(5, 32, 31, 27, 301773)	
498	(302065, 49, 4)	[40, 300, 10, 8]		358 משיח	79255(5, 32, 39, 8, 79255)	אללה	5(5, 32, 39, 26, 302065)	
499	(302185, 2, 5)	[40, 300, 10, 8, 10]		368 משיחי	79286(5, 32, 41, 12, 79286)	אולס	4(5, 32, 41, 49, 302185)	
500	(302185, 2, 4)	[40, 300, 10, 8]		358 משיח	79286(5, 32, 41, 12, 79286)	אשלם	4(5, 32, 41, 49, 302185)	
501	(303176, 89, 5)	[40, 300, 10, 8, 10]		368 משיחי	79556(5, 33, 9, 1, 79556)	תנור	3(5, 33, 9, 3, 303176)	
502	(303176, 89, 4)	[40, 300, 10, 8]		358 משיח	79556(5, 33, 9, 1, 79556)	תנור	3(5, 33, 9, 3, 303176)	
503								
504								
505								
506								
507								
508								
509								
510								
511								
512								
513								
514								
515								
516								
517								
518								
519								
520								
521								
522								

22.0) For each ELS Match, there corresponds an individual CSV file for that ELS Match



The screenshot shows a LibreOffice Calc spreadsheet titled "USER_FILE_WordsOfELSSs_ELSMatches_BY_LETTER_FIRST_POSITIVE_Koren_40TORAH_Instruction_20x15241.csv". The spreadsheet contains approximately 16,384 columns and 500 rows. The columns are labeled A through I. Column A contains numerical values and coordinates, while columns B through I contain Hebrew text and their corresponding coordinates. The text includes words like "למיין", "למיה", "למיה", "למיון", etc. The bottom status bar shows the date as 10/5/2024.

22.1) For ELS #6, ELS Match #498, with $(n, d, k) == (102774, 2, 7)$



The screenshot shows a LibreOffice Calc spreadsheet titled "USER_FILE_WordsOfELSSs_POSITIVE_ELS6_ELSMatch498_d2_n102774_k7_Koren_40TORAH_Instruction_20x15241.csv". The spreadsheet has a header row with columns A through J. The data starts at row 2, with columns including "LetterGematriaNumberValue", "Letter", "LetterPositionIndex", "LetterCoordinatesD5K", "Found in Word in Text", "LetterPositionIndex in Word", "WordNumber", "WordCoordinatesDW", "WordNumberInVerse", and "Found In Verse". The "Found In Verse" column contains repeated text entries such as "זהו אל יזא אש מתקום ביום השבעי". The bottom status bar shows the date as 10/5/2024.

23.3) Now the words in the selected text(s) are sorted according to their **Gematria Word Total**

The screenshot shows a LibreOffice Calc spreadsheet titled "USER_FILE_WordsInSelectedBiblicalTexts_WordPositions_LetterPositions_GematriaValues_Koren_40TORAH_Instruction_20x15241.csv". The table has columns labeled A through H. Row 1 contains column headers: A, B, C, D, E, F, G, H. Rows 2 through 4427 contain data. Column A lists integers from 24393 to 24427. Column B contains various Gematria values represented as strings or lists of numbers. Column C contains more Gematria values. Column D contains additional Gematria values. Columns E, F, G, and H contain the number 85. The status bar at the bottom indicates "Average: 86; Sum: 2666".

24.0) All ELS Matches can be plotted on a 2D Matrix CSV of any size according to the **5-Integer Tuple Letter Coordinates Mathematical ID system**; The **5th Integer (n)** is the **critical “primary key”**, and the same **Letter Position (n)** of **(n, d, k)**

The screenshot shows a LibreOffice Calc spreadsheet titled "USER_FILE_Matrix2D_Koren_40TORAH_Instruction_20x15241.csv". The table has columns labeled A through Z. Row 1 contains column headers: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z. Rows 2 through 25 contain data. Column A lists integers from 1 to 25. Column B contains Hebrew characters. Column C contains letter positions. Column D contains mathematical symbols. Column E contains letter positions. Column F contains letter positions. Column G contains letter positions. Column H contains letter positions. Column I contains letter positions. Column J contains letter positions. Column K contains letter positions. Column L contains letter positions. Column M contains letter positions. Column N contains letter positions. Column O contains letter positions. Column P contains letter positions. Column Q contains letter positions. Column R contains letter positions. Column S contains letter positions. Column T contains letter positions. Column U contains letter positions. Column V contains letter positions. Column W contains letter positions. Column X contains letter positions. Column Y contains letter positions. Column Z contains letter positions. The status bar at the bottom indicates "Average: ; Sum: 0".

**24.1) For ELS #6, ELS Match #498, $(n, d, k) = (102774, 2, 7)$,
and ELS #8, ELS Match #499, $(n, d, k) = (102774, 2, 6)$**

2D Matrix of 20 x 15241

The screenshot shows a LibreOffice Calc spreadsheet with the following details:

- File Name:** USER_FILE_Matrix2D_Koren_40TORAH_Instruction_20x15241.csv
- Sheet:** Sheet 1 of 1
- Cells:** A1 to X5141
- Data:** The matrix consists of 20 columns and 15241 rows. Each cell contains either a coordinate triplets (e.g., (2, 16, 24, 38, 102580)) or a character from the Hebrew alphabet.
- Highlight:**
 - Row 5140, Column 141: (n, d, k) = (102774, 2, 6)
 - Row 5141, Column 141: (n, d, k) = (102774, 2, 7)
- Toolbar:** Standard Calc toolbar with icons for file operations, cell selection, and data processing.
- Bottom Status Bar:** Shows system information like battery level, temperature (81°F), and date/time (10/5/2024).

25.0) For any text(s) selected, a CSV file is created with Letter Statistics to allow calculation of statistical significance of ELS searches (i.e. requires knowledge of statistics to calculate according to your own definitions and algorithms for “statistical significance”)

A	B	C	D	E	F	G	H	I	J	K	L	M	N
Hebrew Letter	Number Of Instances	Length Of Text	Percentage Of Text as Decimal	Percentage Of Text as %									
א	27059	304805	0.0887747904397894	8.87747904397894									
ב	16345	304805	0.0536244484178409	5.36244484178409									
ג	2109	304805	0.006917783500927	0.6917783500927									
ד	7032	304805	0.0230704876888503	2.30704876888503									
ה	28056	304805	0.0920457341579042	9.20457341579042									
ו	30513	304805	0.100106625547481	10.0106625547481									
ז	2198	304805	0.0072116779580387	0.72116779580387									
ח	7189	304805	0.0235855711028362	2.35855711028362									
ט	1804	304805	0.00591853808172438	0.591853808172438									
כ	31531	304805	0.103446465773199	10.3446465773199									
ל	8610	304805	0.0282475681173209	2.82475681173209									
מ	3359	304805	0.0110168796443626	1.10168796443626									
נ	11969	304805	0.0392644477616837	3.92644477616837									
ס	21570	304805	0.0707665556667378	7.07665556667378									
ע	14466	304805	0.04745988513803907	4.745988513803907									
פ	10624	304805	0.03485957012750775	3.485957012750775									
צ	25090	304805	0.0823149226595683	8.23149226595683									
צ'	9867	304805	0.0323715162152852	3.23715162152852									
צ''	4259	304805	0.0139726678991486	1.39726678991486									
צ'''	14126	304805	0.0463443841144338	4.63443841144338									
צ'''	18333	304805	0.00601368087793835	0.601368087793835									
צ''''	11250	304805	0.0369088433588688	3.69088433588688									
צ'''''	3975	304805	0.013041124653467	1.3041124653467									
צ''''''	830	304805	0.00272305244336543	0.272305244336543									
צ'''''''	4805	304805	0.0157641770968324	1.57641770968324									
צ''''''''	2927	304805	0.00960286084545857	0.960286084545857									
צ'''''''''	1035	304805	0.00339561358901593	0.339561358901593									
צ''''''''''	3962	304805	0.012998474344745	1.2998474344745									
צ'''''''''''	4695	304805	0.0154032906284346	1.54032906284346									
צ''''''''''''	18125	304805	0.059464247633733	5.9464247633733									
צ'''''''''''''	15595	304805	0.051163858860583	5.1163858860583									
צ''''''''''''''	17950	304805	0.0588901100703729	5.88901100703729									

26.0) The ELS Search Algorithm

WRR's central search algorithm for each ELS Search Term, *ELS* (*ELSw*),¹ is as follows:

$$n + (n + d) + (n + 2d) + (n + 3d) \dots + (n + (k-1)d)$$

27.0) Equidistant Letters Sequences (ELSs)

As an initial scientific hypothesis (to be refined in research as led by the data), if anything is going to be hidden and encrypted within the Hebrew Bible as Equidistant Letter Sequences (ELSs), the term “*Messiah*” is among the most central concepts in Judaism, and thus among the most likely to be found as ELSs. Therefore, the following *a priori* ELS Search Terms (*ELSw*) – i.e. “*Messiah*” and several permutations – will be chosen as our initial ELS Search-Term variables (*ELSw*) to initialize (and search for) as per the protocol of WRR (1994) et al. WRR would define these ELS Search Terms as initial word anchors (i.e. test-variables) within the text(s) that will anchor us to a 2D Matrix (in that area) that will serve as a basis for a scientific experimental protocol and methodology:

$$ELS1 = “\text{המשיח}” == “\text{HaMashiach}” == “\text{The Messiah.”}$$

¹ e.g. *ELS1*, *ELS2*, *ELS3*, *ELS4*, *ELS5*, *ELS6*...

$ELS2 = \text{“משיחי”} == \text{“Meshichi”} == \text{“My Messiah.”}$

$ELS3 = \text{“משיחו”} == \text{“Meshicho”} == \text{“His Messiah.”}$

$ELS4 = \text{“משיח”} == \text{“Mashiach”} == \text{“Messiah.”}$

$ELS5 = \text{“מי המשיח”} == \text{“Mi HaMashiach”} == \text{“Who [is] the Messiah?”}$

$ELS6 = \text{“מי משיחי”} == \text{“Mi Meshichi”} == \text{“Who [is] My Messiah?”}$

$ELS7 = \text{“מי משיחו”} == \text{“Mi Meshicho”} == \text{“Who [is] His Messiah?”}$

$ELS8 = \text{“מי משיח”} == \text{“Mi Mashiach”} == \text{“Who [is] Messiah?”}$

k == length (k) of an ELS Search-Term, ELS , i.e. the number of letters in the word; here:

- 1.) The length k of $ELS1$ is 5;
- 2.) The length k of $ELS2$ is 5;
- 3.) The length k of $ELS3$ is 5;
- 4.) The length k of $ELS4$ is 4;
- 5.) The length k of $ELS5$ is 7;
- 6.) The length k of $ELS6$ is 7;
- 7.) The length k of $ELS7$ is 7;
- 8.) The length k of $ELS8$ is 6;

d == equidistant skip distance(s) (d) between each letter for each ELS match successfully found within the selected text that is searched; For each instance of a letter found at letter position index number (n), there may be multiple instances of skip distances (d) – of varying lengths – possible to the next letter in an ELS match. In other words, in some cases, there may be shared letters within the same letter position index number (n) for multiple ELSs of the same ELS search term with multiple (different) skip distances (d), and/or for other ELS search terms with a shared letter position index number (n).

n == letter position index number starting position (n) for each instance for each first (or last) letter in the ELS; For example, if you are searching for a word that begins (or ends) with the letter Mem (מ / נ) in the text, index number (n) is the index position number (n) for each instance found of the letter Mem (מ / נ), and for each instance of these letters Mem (מ / נ) found, there is the potential to find multiple ELSs beginning (or ending, or overlapping) on that letter position index number (n).

WRR define (n, d, k) as the unique ELS ID for each ELS match, and indeed these unique mathematical numbers integrate into the standardized mathematical ID system detailed below.

28.0) The Ancient Method of Counting by Eye and Hand

In the ancient Hebrew language, the word for Scribe is *Sofer* (סופר) - literally one who “counts” letters – with each letter also having a numerical value. Counting letters and their positions within the word, verse, and entire text was part of the scribe's responsibilities, and certainly one of the means by

which the scribe ensured accuracy of the transcribed texts, e.g. counting the number of letters in that verse (or word) that he just transcribed letter-by-letter: Mathematics never lies.

It is clear to anyone that studies the code that G-D is responsible for encoding them there: Rabbi Bachya Ben Asher (1255-1340), Rabbi Moshe ben Yaakov Cordovero (1522-1570), Rabbi Michael Dov Weissmandl (1958), WRR (1994), and those following WRR.

29.0) Choosing a Size for the 2D Matrix

After choosing our *a priori* ELS Search-Terms, one can choose size(s) for 2D matrices. The size(s) of the user-selected 2D matrices will be significant mostly for ease of human User Experience (UX) to visualize the ELSs found as they have no effect on the ELSs found. This is because the distance (i.e. proximity) between ELSs will not change according to the 2D matrix size – no matter how big or small – because the distance between letters in ELS matches will never change.

Therefore, do not be deceived by extremely large skip distances (d) with extremely large 2D matrices as these are less statistically (mathematically) significant than smaller skip distances (d) and smaller 2D matrix sizes with clusters of ELS matches because it is these smaller skip distances (d) that indicate close proximity in the underlying text.

Although 2D matrices of other sizes need to be tested for optimal UX, following the scientific method, let us propose an experimental research protocol that initially creates a 2D matrix crossword puzzle of 20 columns (width = x) by the corresponding variable number of rows (height = y) for that particular biblical text selected.

No matter what the size of the 2D matrix, other than edge-case exceptions on either right or left side of the 2D matrix helix, counting letters in any of the eight (8) directions is an equidistant letter sequence: A column-width (x) of any integer number, e.g. 20 or 9, or 10, 18, 27, 30, 33, etc., allows one to potentially find ELSs visually in any linear direction, i.e. if you see words going in one of these eight (8) directions:

- 1.) Up
- 2.) Down
- 3.) Right
- 4.) Left
- 5.) Diagonally Up Right
- 6.) Diagonally Down Right
- 7.) Diagonally Up Left
- 8.) Diagonally Down Left

Even if one is not mathematically-oriented, anyone can visually find ELSs (all of which are potentially possible hidden Torah Bible Codes) as if in a crossword (or word search) puzzle, and nevertheless still be able to present the results to be confirmed by others scientifically and mathematically. That is to say: Any discovery found visually – by anyone whomsoever – in these Torah Bible Codes can be both visually and/or mathematically confirmed accurately to the level of precise mathematics, and repeated scientifically by others (i.e. anyone) using this same

TorahBibleCodes software and/or using the same standardized mathematical ID system.

Although there may be some advantages to various other 2D matrix sizes, for our purposes here for the Koren Codex Torah, Leningrad Codex Tanakh, and/or MAM Collection of Manuscripts (based on the Aleppo Codex) Tanakh, 20 rows seems an excellent (if not optimal) initial choice for human-friendliness: It is relatively easy on the eyes, and allows for easy visual searching for these hidden codes (i.e. the ancient old-fashioned way) through counting by eye and hand with a base of ten (10), with multiples of 2, 3, 4, 5, 6, 7, 8, 9, 10, and 20 (with single-digit +/- permutations) very easy and possible to count and calculate.

With this size of 20 (and evidently any integer number of) columns for the 2D matrix, certain rules become evident after you begin your counting and search, with given: (x) = width = number of columns; (y) = height = number of rows.

1.) Going up or down by one row is an exact ELS with a skip distance (d) that is exactly equal the (x) number of columns, e.g. 20 columns here; For example, 20 columns of text here means that going down vertically by one unit to the next row to the letter directly below it $(-x, -y)$ is exactly (+20) letters in the positive (+) direction with an ELS skip distance (d) of (+20).

2.) Conversely, going up vertically by one unit to the next row to the letter directly above it (x, y) is exactly (-20) letters in the negative (-) direction with an ELS skip distance (d) of (-20).

3.) Going down diagonally by one unit to the left $(-x, -y)$ is an ELS skip distance (d) of (+21) letters from that letter to the next letter.

4.) Going up diagonally by one unit to the right (x, y) is an ELS skip distance (d) of (-21) letters from that letter to the previous letter.

5.) Going down diagonally by one unit to the right $(x, -y)$ is an ELS skip distance (d) of (+19) letters from that letter to the next letter.

6.) Going up diagonally by one unit to the left $(-x, y)$ is an ELS skip distance (d) of (-19) letters from that letter to the previous letter.

There are edge cases in using shortcuts (e.g. $(+/-Ax)$, $(+/-By)$) by diagonal units) for counting and placement as one approaches either side of the 2D matrix helix as one goes up or down – so one must be careful in counting as one reaches either left or right side of the row as one continues diagonally up or down; Especially if one is counting by more than one unit in diagonal directions, the mathematics of the movement must be correct.

The following algebraic formula, and its various graphical representations (e.g. vertical lines, horizontal lines, and diagonal lines characterized by a dynamic L shapes (that can spin 360 degrees on the 2D (x,y) plane), will help one to search, visualize, see, count, and find ELS word patterns and matches in one's word-search by eye and hand:

$$y = mx + b$$

30.0) Unique VerseIDs, LetterIDs, WordIDs, and ELS IDs Coordinates System

In order for us to share and peer-review precise mathematical, statistical, and visual data that is produced from TorahBibleCodes Bible Search and Research Software tools, we build upon a unique 3-integer tuple VerseID, unique for each Book, Chapter, and Verse within the Hebrew Canon. Here we demonstrate and examine the ID system as it would apply to the entire Koren Codex Hebrew Torah together as one text string of letters, with each letter, word, verse, chapter, and book having a unique mathematical ID.

This same ID system is applied to other codices such as the Leningrad Codex and the MAM Collection of Manuscripts (based on the Aleppo Codex) in order to make possible the precise, letter-by-letter, mathematical and scientific comparison between codices and manuscripts.

30.1) (*Book#*, *Chapter#*, *Verse#*) – 3-Integer Tuple VerseID

This 3-integer tuple VerseID is unique to each verse within each separate Book within the Hebrew Bible.

EXAMPLES:

(1, 1, 1) == (Genesis, 1, 1) == Genesis 1:1

(1, 1, 2) == (Genesis, 1, 2) == Genesis 1:2

(1, 1, 3) == (Genesis, 1, 3) == Genesis 1:3

(2, 1, 1) == (Exodus, 1, 1) == Exodus 1:1

(2, 1, 2) == (Exodus, 1, 2) == Exodus 1:2

(2, 1, 3) == (Exodus, 1, 3) == Exodus 1:3

(3, 1, 1) == (Leviticus, 1, 1) == Leviticus 1:1

(3, 1, 2) == (Leviticus, 1, 2) == Leviticus 1:2

(3, 1, 3) == (Leviticus, 1, 3) == Leviticus 1:3

(4, 1, 1) == (Numbers, 1, 1) == Numbers 1:1

(4, 1, 2) == (Numbers, 1, 2) == Numbers 1:2

(4, 1, 3) == (Numbers, 1, 3) == Numbers 1:3

(5, 34, 10) == (Deuteronomy, 34, 10) == Deuteronomy 34:10

(5, 34, 11) == (Deuteronomy, 34, 11) == Deuteronomy 34:11

(5, 34, 12) == (Deuteronomy, 34, 12) == Deuteronomy 34:12

(12, 1, 1) == (Isaiah, 1, 1) == Isaiah 1:1

(12, 1, 2) == (Isaiah, 1, 2) == Isaiah 1:2

(12, 1, 3) == (Isaiah, 1, 3) == Isaiah 1:3

(27, 1, 1) == (Psalms, 1, 1) == Psalms 1:1

(27, 1, 2) == (Psalms, 1, 2) == Psalms 1:2

(27, 1, 3) == (Psalms, 1, 3) == Psalms 1:3

- (35, 12, 10) == (Daniel, 12, 10) == Daniel 12:10
- (35, 12, 11) == (Daniel, 12, 11) == Daniel 12:11
- (35, 12, 12) == (Daniel, 12, 12) == Daniel 12:12

Building upon this unique 3-integer tuple Verse ID, we create several multiple, overlapping (yet each unique) LetterIDs and WordIDs for each Letter and Word respectively within each Codex (or Collection of Manuscripts) Text:

30.2) (*Book#*, *Chapter#*, *Verse#*, *Letter#InVerse*) – 4-Integer LetterID

This 4-integer LetterID will be unique to each individual letter within each separate Book in the Hebrew Bible, and it is one way (of several that overlap for complementary purposes) to uniquely identify letter(s) in our Hebrew Codices; The 4-integer LetterID is helpful when you need to know the exact letter position index number (n) of that letter within that particular verse.

EXAMPLE – ELS #2, ELS Match #88, $(n, d, k) == (726, 89, 5)$, ‘מֶשֶׁנָּה

- “My Messiah”:
- (1, 1, 16, 9) == (Genesis, Chapter 1, Verse 16, Letter 9 in Verse)
 - (1, 1, 17, 19) == (Genesis, Chapter 1, Verse 17, Letter 19 in Verse)
 - (1, 1, 20, 2) == (Genesis, Chapter 1, Verse 20, Letter 2 in Verse)
 - (1, 1, 21, 34) == (Genesis, Chapter 1, Verse 21, Letter 34 in Verse)
 - (1, 1, 22, 34) == (Genesis, Chapter 1, Verse 22, Letter 34 in Verse)

30.3) (*Book#*, *Chapter#*, *Verse#*, *Letter#InVerse*, *Letter#InText*) – 5-Integer LetterID

This 5-integer LetterID is another way to uniquely identify each letter within a text. However, this 5-integer LetterID also takes into account the length of the total text, e.g. here the entire Hebrew Torah with 304,805 letters of the Koren Codex.²

EXAMPLE – ELS #2, ELS Match #88, $(n, d, k) == (726, 89, 5)$, ‘מֶשֶׁנָּה

- “My Messiah”:
- (1, 1, 16, 9, 726) == (Genesis, Chapter 1, Verse 16, Letter 9 in Verse, Letter 726 in Text)
 - (1, 1, 17, 19, 815) == (Genesis, Chapter 1, Verse 17, Letter 19 in Verse, Letter 815 in Text)
 - (1, 1, 20, 2, 904) == (Genesis, Chapter 1, Verse 20, Letter 2 in Verse, Letter 904 in Text)
 - (1, 1, 21, 34, 993) == (Genesis, Chapter 1, Verse 21, Letter 34 in Verse, Letter 993 in Text)
 - (1, 1, 22, 34, 1082) == (Genesis, Chapter 1, Verse 22, Letter 34 in Verse, Letter 1082 in Text)

This 5-integer LetterID is useful because the 5th integer (n) is the same number as the variable for index letter position (n) for each ELS match found, and therefore is the overlapping primary key that enables identifying, matching, and connecting to the unique ELS ID (n, d, k) of each ELS match found:

² vs. 304,850 letters for Leningrad Codex, and 304,801 letters for the MAM Collection of Manuscripts

30.4) (n , d , k) – 3-Integer ELS ID

Thus the above 5-integer LetterIDs are the same letters (and letter positions) that are identified with WRR's unique ELS identifier (n , d , k) that identifies the first (or last) letter positions for that ELS match:

EXAMPLE:

(726, 89, 5) == (Letter#InText n , Skip-Distance d , LengthOfELSSearchTerm k)
(726, 89, 4) == (Letter#InText n , Skip-Distance d , LengthOfELSSearchTerm k)

k == length of ELS term (k), e.g. for (n , d , k) == (726, 89, 5), the length (k) is 5 (**מֶשִׁיחַ**, “Meshichi”, “My Messiah”).

d == skip distance (d), i.e. here (d) is a skip distance (d) of (+89) in the positive direction; The TorahBibleCodes program uses absolute values for search and computation, i.e. begins on both the first and last letters of the potential ELS match and searches in the (+) positive (forward) direction. Therefore the ELS ID (n , d , k) for (-) negative skip distances (d) is associated with the last letter of the ELS match as the primary (beginning) letter in that (-) negative ELS match progression that counts backwards from that last letter;

Whereas other code researchers (and software) may be identifying only the first letter of the ELS match and doing both a (+) positive and (-) negative search, our program and algorithm searches for both first and last letters of each ELS Search Term, and only searches in the (+) positive (forward) direction. This is the same as finding the first letter in the ELS and searching in both (+) positive (forward) and (-) negative (backward) directions;

Therefore, our ELS ID (n , d , k) identifies the first letter of the ELS match when it is a (+) “positive” skip-distance (d), and it identifies the last letter of the ELS match when it is a (-) “negative” skip-distance (d);

The positive (+) and negative (-) ELS matches are each contained in separate CSV files, and by default are ordered by the smallest skip-distance (d) in ascending order.

n == letter position index number starting position (n) for the first (or final/last) letter in the ELS Search-Term; If this were to be a (-) negative skip distance (d), the ELS Search Match would begin with the final letter of the ELS match, and count backwards. However, here the ELS match is (+) positive with (n , d , k) == (726, 89, 5), **מֶשִׁיחַ**, “My Messiah”.

30.5) (*Book#*, *Chapter#*, *Verse#*, *Word#InVerse*) – 4-Integer WordID

EXAMPLE – ELS #2, ELS Match #88, (n , d , k) == (726, 89, 5), **מֶשִׁיחַ**, “My Messiah”:

(1, 1, 16, 2) == (Genesis, Chapter 1, Verse 16, Word 2 in Verse)
(1, 1, 17, 5) == (Genesis, Chapter 1, Verse 17, Word 5 in Verse)
(1, 1, 20, 1) == (Genesis, Chapter 1, Verse 20, Word 1 in Verse)

(1, 1, 21, 9) == (Genesis, Chapter 1, Verse 21, Word 9 in Verse)
(1, 1, 22, 9) == (Genesis, Chapter 1, Verse 22, Word 9 in Verse)

30.6) (*Book#, Chapter#, Verse#, Word#InVerse, Word#InText*) – 5-Integer WordID

EXAMPLE – ELS #2, ELS Match #88, $(n, d, k) == (726, 89, 5)$, 'מֶשֶׁה', "My Messiah"

(1, 1, 16, 2, 186) == (Genesis, Chapter 1, Verse 16, Word 2 in Verse, Word 186 in Text)
(1, 1, 17, 5, 207) == (Genesis, Chapter 1, Verse 17, Word 5 in Verse, Word 207 in Text)
(1, 1, 20, 1, 229) == (Genesis, Chapter 1, Verse 20, Word 1 in Verse, Word 229 in Text)
(1, 1, 21, 9, 252) == (Genesis, Chapter 1, Verse 21, Word 9 in Verse, Word 252 in Text)
(1, 1, 22, 9, 275) == (Genesis, Chapter 1, Verse 22, Word 9 in Verse, Word 275 in Text)

31.0) Plotting ELS Matches on a 2D Matrix

The ELS codes (i.e. ELS Search-Term Matches) presented here (and every search match) can be mathematically located and confirmed to the level of that specific letter with its various unique IDs to identify letter position index number (n) for that letter, word coordinates, verse, chapter, and book.

For the accompanying CSV files in the 2D matrix, the exact verse and letter position(s) of both the first and last letters of each row are available on the right and left sides respectively of each row with a 5-integer LetterID that is dynamically-generated depending upon the size of the 2D matrix generated by the user.

For any text chosen in a 2D matrix, e.g. 20 column-width, the first and last columns on the left and right side of every 2D matrix are identified with numbers indicating the exact letter position (n) of the letter on either the left or right; Thus whether one be mathematically, statistically, or visually-inclined, others can scientifically confirm any discovery made by anybody via this unique VerseID, LetterID, WordID, and ELS ID Coordinates System.

32.0) Appendix (including Links to Software and Research Data)

<https://TorahBibleCodes.com>
<https://github.com/TorahBibleCodes>

33.0) Author's Afterword

ב"ה

If you feel blessed by this scientific biblical research (and dropping of knowledge), please consider supporting financially the continued research, R&D, shared free software, data, discoveries, and insights.

We can receive support at GiveSendGo:
<https://givesendgo.com/TorahBibleCodes>

As well as via Cryptocurrencies via our website:
<https://TorahBibleCodes.com>

Or via the contact details below to donate and support privately and confidentially.

Enjoy the Free, Open-Source Python TorahBibleCodes Bible Search and Research Software!

Daniel Azariah
AzariahBible.com
TorahBibleCodes.com
info@TorahBibleCodes.com
Jerusalem, Israel

34.0) References:

Bombach, N.; Gans, H.; "Patterns of Co-Linear Equidistant Letter Sequences and Verses", 2006, pp. 1-3.

https://www.researchgate.net/publication/220930153_Patterns_of_Co-Linear_Equidistant_Letter_Sequences_and_Verses

Gans, H.; Inbal, Z.; Bombach, N.; "Patterns of Equidistant Letter Sequence Pairs in Genesis", 2005, pp. 1-5

<https://www.semanticscholar.org/paper/Patterns-of-Equidistant-Letter-Sequence-Pairs-in-Gans-Inbal/44d1a1158127212969169accea4e2a1ff3ea60e7>

Haralick, R.M.; Basic Concepts For Testing The Torah Code Hypothesis, 2006A, pp. 1-6.

Haralick, R.M.; Testing The Torah Code Hypothesis: The Experimental Protocol, 2006B, pp. 1-6.

Haralick R.M.; "Torah Codes: New Experimental Protocols", 1998, pp. 1-23.

Levitt, A.; Haralick, R.M.; Rips, E.; "Linguistic Connections among Torah Codes 29 November 2004", 2004, pp. 1-5.

Levitt, A.; Nachum B.; Gans, H.; Haralick, R.; Schwartzman, L.; Stal, C.; "Long Phrases in Torah

Codes", 2004, pp. 1-4.

https://www.academia.edu/21523850/Long_Phrases_in_Torah_Codes

https://www.researchgate.net/publication/242139408_Long_Phrases_in_Torah_Codes

Mamiani, M; "Newton on Prophecy and the Apocalypse", 2002, pp. 387 - 408, within: The Cambridge Companion to Newton, (Editors: Cohen, I.B. and Smith, G.E.), Cambridge University Press, Cambridge, New York, 2002, pp. 1 – 500.

McKay, B.; "Brief notes on Gans' Primer",

http://users.cecs.anu.edu.au/~bdm/codes/StatSci/gans_rep.html

McKay, B.; "Jesus as the Son of Man",

<http://users.cecs.anu.edu.au/~bdm/codes/Jesus/>

McKay, B.; Bar-Natan, D.; Bar-Hillel, M.; Kalai, G.; "Solving the Bible Code Puzzle", *Statistical Science*, 14(5), May 1999, 1999, pp. 150-173.

Wilson, J.; "Bible Code, Revisited"; *Perspectives on Science and Christian Faith*, Volume 71, Number 1, March 2019, 2019, pp. 3-15.

Witztum, D.; Rips, E.; Rosenberg, Y.; "Equidistant Letter Sequences in the Book of Genesis", *Statistical Science*, 9(3), 1994, pp. 429-438.