

[220 / 319] Tabular Data

Department of Computer Sciences
University of Wisconsin-Madison

Readings:
Chapter 16 of Sweigart

Learning Objectives Today

CSV format

- purpose
- syntax
- comparison to spreadsheet

Reading CSV files

- without header
- with header
- type casting

Chapter 16 of Sweigart, to (and including)
“Reading Data from Reader Objects in a for
Loop”

Today's Outline

Spreadsheets

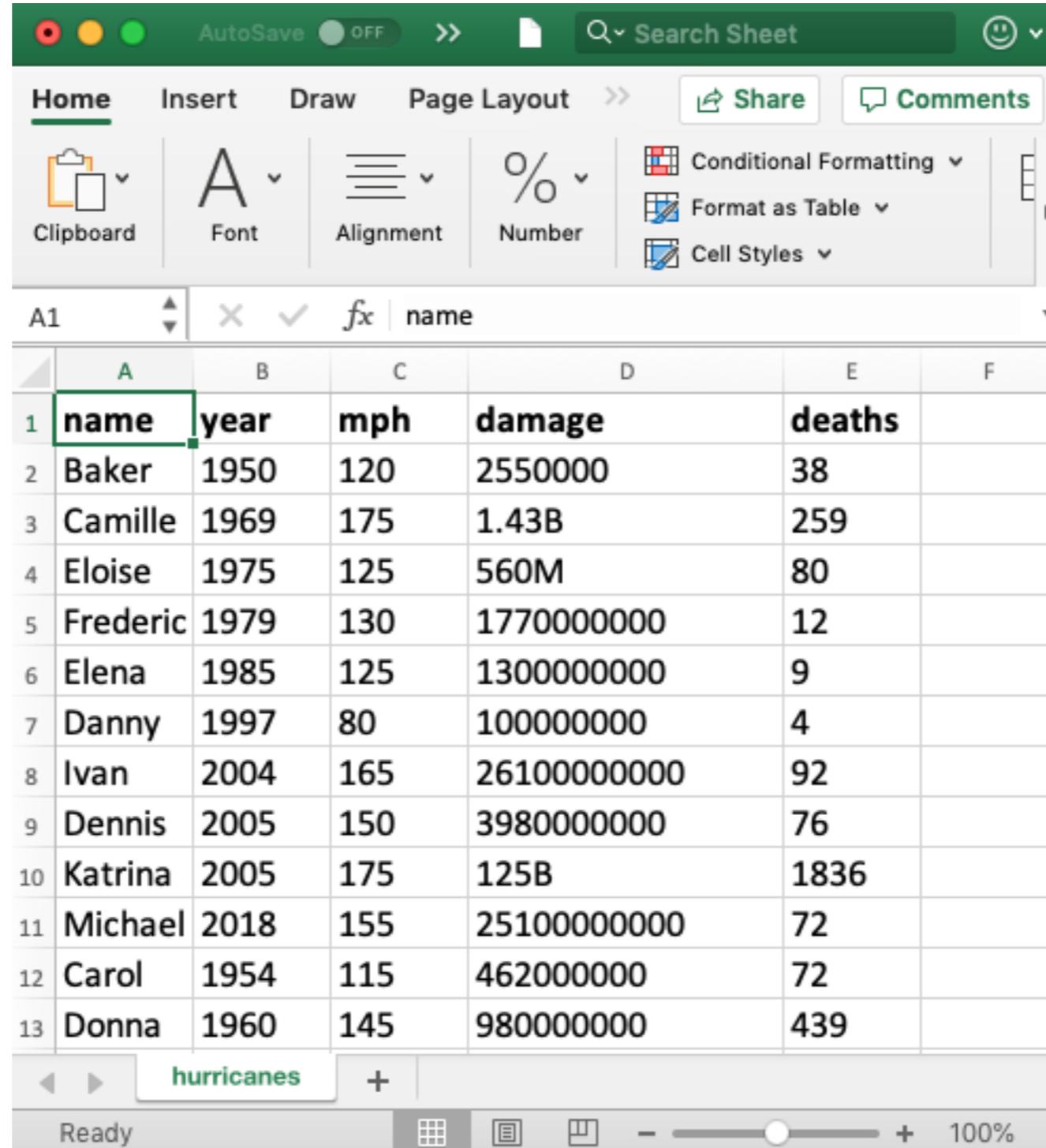
CSVs

Reading a CSV to a list of lists

Coding examples

Spreadsheets (e.g., Excel)

Spreadsheets are tables of cells, organized by rows and columns



The image shows a screenshot of a spreadsheet application interface. The top bar includes window controls, an 'AutoSave' toggle (OFF), a search bar labeled 'Search Sheet', and a smiley face icon. Below this is a ribbon with tabs for 'Home', 'Insert', 'Draw', and 'Page Layout'. The 'Home' tab is active, showing options for 'Clipboard', 'Font', 'Alignment', 'Number', 'Conditional Formatting', 'Format as Table', and 'Cell Styles'. The formula bar shows 'A1' and the formula '=name'. The main area contains a table with 13 rows and 6 columns. The columns are labeled 'name', 'year', 'mph', 'damage', and 'deaths'. The data rows list hurricanes: Baker, Camille, Eloise, Frederic, Elena, Danny, Ivan, Dennis, Katrina, Michael, Carol, and Donna. The status bar at the bottom shows 'Ready', a grid icon, a zoom slider set to 100%, and a plus sign.

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	26100000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	25100000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

Spreadsheets (e.g., Excel)

Spreadsheets are tables of cells, organized by rows and columns

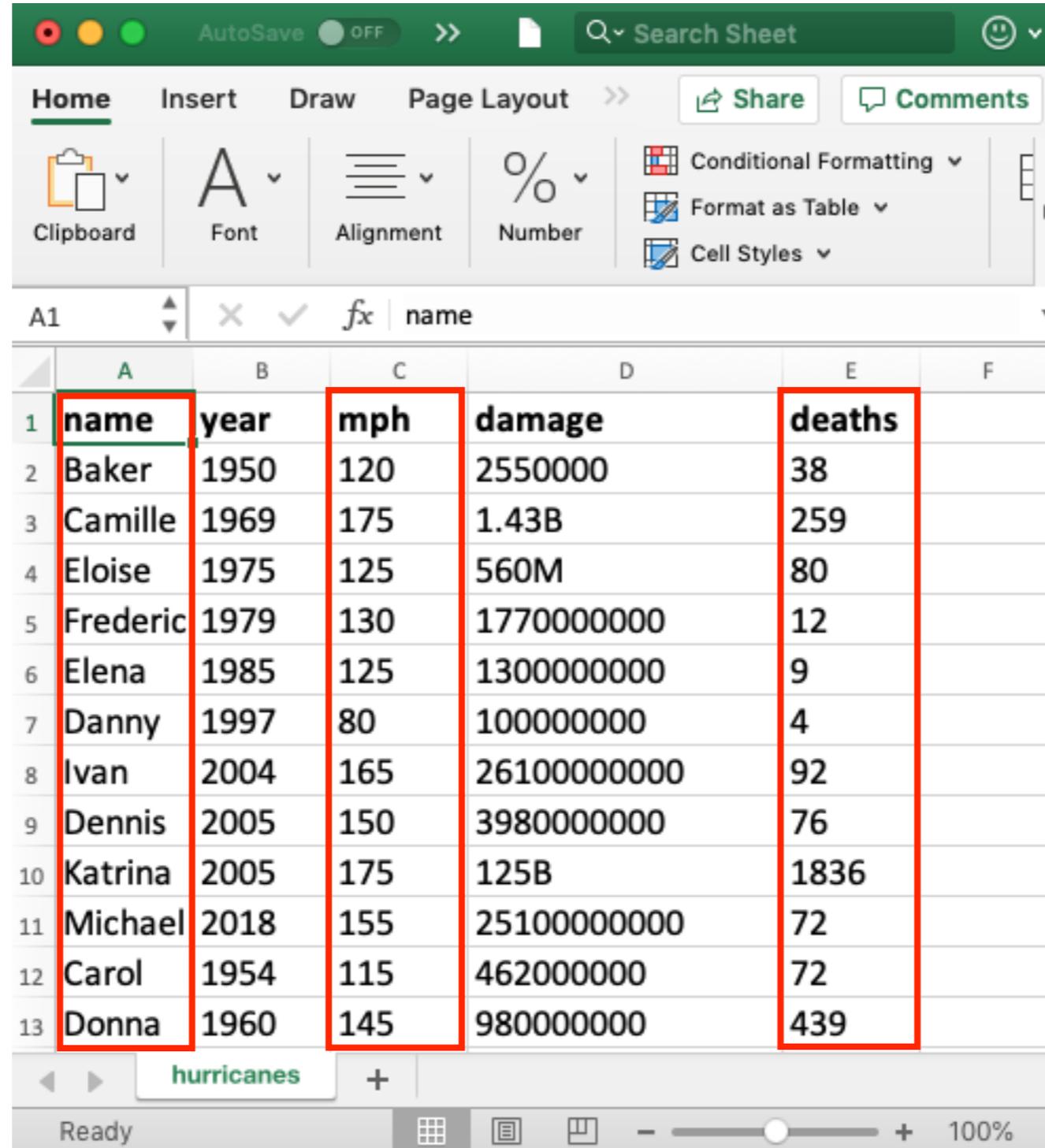
cells

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	2610000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	2510000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

Spreadsheets (e.g., Excel)

Spreadsheets are tables of cells, organized by rows and columns

columns



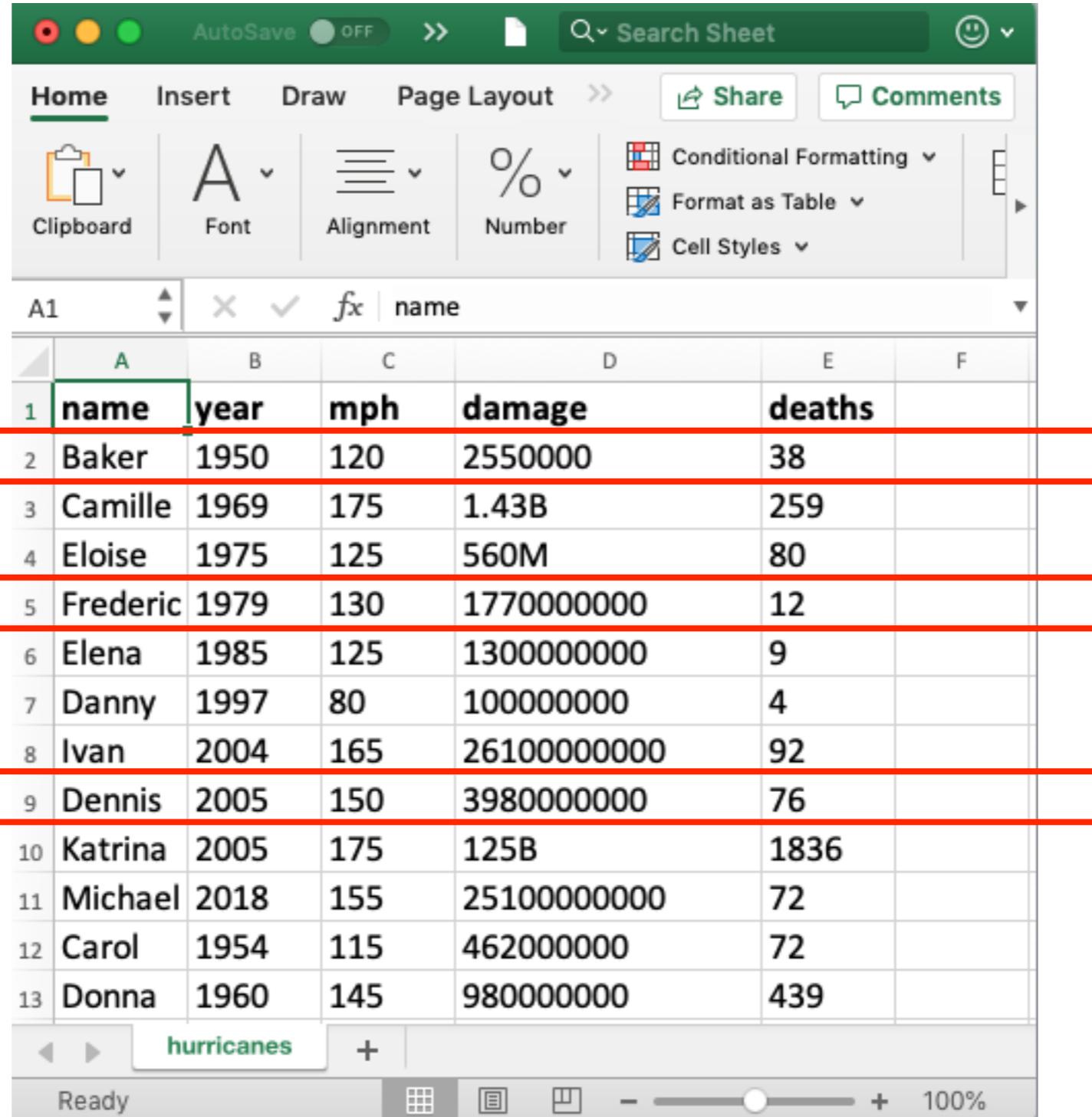
The screenshot shows the Microsoft Excel interface. The ribbon is set to 'Home', and the 'Clipboard', 'Font', 'Alignment', and 'Number' groups are visible. The formula bar shows 'name' in cell A1. The spreadsheet contains a table with 13 rows and 6 columns. The columns are labeled 'name', 'year', 'mph', 'damage', and 'deaths'. The 'name' and 'deaths' columns are highlighted with red boxes. The 'hurricanes' sheet is selected at the bottom.

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	2610000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	2510000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

Spreadsheets (e.g., Excel)

Spreadsheets are tables of cells, organized by rows and columns

rows



The screenshot shows a spreadsheet application window with a green title bar. The ribbon includes 'Home', 'Insert', 'Draw', and 'Page Layout'. The 'Home' ribbon is active, showing options for Clipboard, Font, Alignment, Number, Conditional Formatting, Format as Table, and Cell Styles. The active cell is A1, containing the text 'name'. The spreadsheet contains a table with the following data:

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	2610000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	2510000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

The spreadsheet is titled 'hurricanes' and is in 'Ready' state. The zoom level is 100%.

Spreadsheets (e.g., Excel)

Spreadsheets are tables of cells, organized by rows and columns

header

The screenshot shows a spreadsheet application interface. The top menu bar includes 'AutoSave OFF', 'Search Sheet', and 'Home' tabs. The 'Home' tab is active, showing options for Clipboard, Font, Alignment, Number, Conditional Formatting, Format as Table, and Cell Styles. The active cell is A1, containing the text 'name'. Below the menu is a table with 13 rows and 6 columns. The first row is highlighted with a red border and labeled 'header'. The data in the table is as follows:

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	2610000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	2510000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

The bottom of the screenshot shows the 'hurricanes' sheet tab, a status bar with 'Ready', and a zoom level of 100%.

Spreadsheets (e.g., Excel)

Spreadsheets often allow different **data types**

The screenshot shows a spreadsheet application interface with a table of hurricane data. The table has columns for name, year, mph, damage, and deaths. The third row contains the data for Hurricane Camille, with the name 'Camille' and the death toll '259' highlighted by red boxes and arrows. The arrows are labeled 'text' and 'numbers' respectively.

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	26100000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	25100000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

Spreadsheets (e.g., Excel)

Spreadsheets often allow different **fonts**

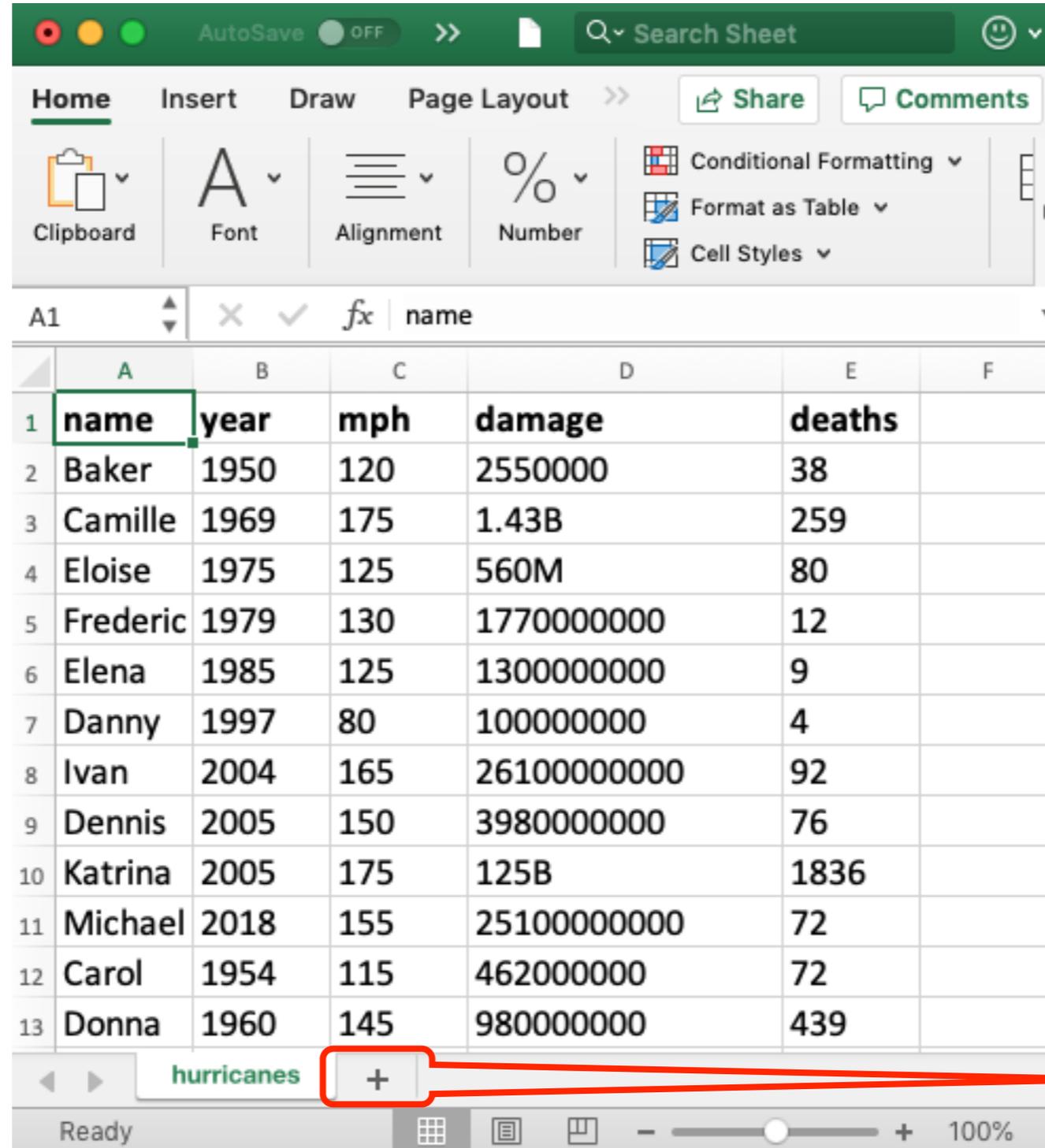
The screenshot shows a spreadsheet application interface. The ribbon includes 'Home', 'Insert', 'Draw', and 'Page Layout'. The 'Home' ribbon has sections for 'Clipboard', 'Font', 'Alignment', 'Number', 'Conditional Formatting', 'Format as Table', and 'Cell Styles'. The active cell is A1, containing the text 'name'. The spreadsheet data is as follows:

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	26100000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	25100000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

Red callouts point to the 'name' header (labeled 'bold') and the 'Eloise' row (labeled 'regular').

Spreadsheets (e.g., Excel)

Spreadsheets often support **multiple sheets**



The screenshot shows a spreadsheet application interface. At the top, there is a green header bar with window controls, 'AutoSave OFF', and a search bar. Below this is a ribbon with tabs for 'Home', 'Insert', 'Draw', and 'Page Layout'. The 'Home' tab is active, showing options for Clipboard, Font, Alignment, Number, Conditional Formatting, Format as Table, and Cell Styles. The active cell is A1, containing the text 'name'. The main area is a table with 13 rows and 6 columns. The columns are labeled 'name', 'year', 'mph', 'damage', and 'deaths'. The data rows contain hurricane names and their corresponding statistics. At the bottom, there is a sheet tab labeled 'hurricanes' and a '+' button to add more sheets, which is highlighted with a red box and an arrow pointing to the text 'more tables of data'.

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	
7	Danny	1997	80	100000000	4	
8	Ivan	2004	165	2610000000	92	
9	Dennis	2005	150	3980000000	76	
10	Katrina	2005	175	125B	1836	
11	Michael	2018	155	2510000000	72	
12	Carol	1954	115	462000000	72	
13	Donna	1960	145	980000000	439	

more tables of data

Excel Files

Extension: .xlsx

Format: **binary** just 0's and 1's, not human-readable characters.
Need special software...

```
lec-15 — -bash — 67x24
ty-mac:lec-15$ cat hurricanes.xlsx
P!b?h^[Content_Types].xml ?(????N?0E?H?C?-J5??*Q>?e[c[?ii????B?j7??
?{2??h?nm????2R

????U^/???%??rZY?1__?f??q??R4D?AJ?h>????V?ε

????????NV
78h?????ji){^??-I?"{?v^?P!XS)bR?r??K?s(??3`c?0?????????7M4??????ZEk+?|
\|z?(???P??6h_-[@?!???Pk????2n?}????L??? ??%????????dN"m,?ÄD097*?~??φ
8?0?c|n???E??????B??!$}?????;{???[????2????P!?U0#?L

_rels/.rels ?(???M0?0
??9L?3?sbg_|?l!??USH9i?b?r:"y_dl??D??|-N??R"4?2?G?%??Z?4?"y?? ë??
? ?????P!>???xl/_rels/workbook.xml.rels ?(??RMK?0?T~?I????$T?G?~??
??<???!??4??;#?w????qu*&r?Fq???v?????GJy(v??*?????K??#F??D??W
?=??Z?MY?b???BS?????????ç? ??

????w?v?t/"?UN)?&!

3~??]X?K/o?y???v?5????+??zl?;o??b???G????

?s?>??,?8??(%???"D??4j?0u2j
s??MY?~???S葵 ??? ?)f???C????y?? Iy????!+??E??fMy?k???
??K?5=|?t ??G)?s墙 ?U??tB??)???,???f????????P!u???
```

Writing code to read data from Excel files is tricky, unless you use special modules

Today's Outline

Spreadsheets

CSVs

Reading a CSV to a list of lists

Coding examples

CSVs

CSV is a simple data format that stands for **Comma-Separated Values**

CSVs are like simple spreadsheets

- organize cells of data into rows and columns
 - only one sheet per file
 - only holds strings
 - no way to specify font, borders, cell size, etc
- you'll do lots of type casting/conversion!
- 

CSV Files

Extension: .csv

Format: **plain text** → just open in any editor (notepad, textedit, idle, etc) and you'll be able to read it

```
lec-16 — -bash — 46x21
ty-mac:lec-16$ ls
h10.csv          h10.xlsx
ty-mac:lec-16$ cat h10.csv
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
Elena,1985,125,1300000000,9
Danny,1997,80,100000000,4
Ivan,2004,165,26100000000,92
Dennis,2005,150,3980000000,76
Katrina,2005,175,125B,1836ty-mac:lec-16$
```

Writing code that understands CSV files is easy

Basic Syntax

Table

Name	Date	Time	Status	Latitude	Longitude	WindSpeed	Ocean
HEIDI	19671019	1200	TD	20.5N	54.0W	25	Atlantic
OLAF	19850822	0	TD	12.9N	102.2W	25	Pacific
TINA	19920917	1200	TD	10.4N	98.5W	25	Pacific
EMMY	19760820	1200	TD	14.0N	48.0W	20	Atlantic

Corresponding CSV

Name,Date,Time,Status,Latitude,Longitude,WindSpeed,Ocean

HEIDI,19671019,1200, TD,20.5N,54.0W,25,Atlantic

OLAF,19850822,0, TD,12.9N,102.2W,25,Pacific

TINA,19920917,1200, TD,10.4N,98.5W,25,Pacific

EMMY,19760820,1200, TD,14.0N,48.0W,20,Atlantic

Each row is a line of the file

Basic Syntax

Table

Name	Date	Time	Status	Latitude	Longitude	WindSpeed	Ocean
HEIDI	19671019	1200	TD	20.5N	54.0W	25	Atlantic
OLAF	19850822	0	TD	12.9N	102.2W	25	Pacific
TINA	19920917	1200	TD	10.4N	98.5W	25	Pacific
EMMY	19760820	1200	TD	14.0N	48.0W	20	Atlantic

Corresponding CSV

Name,Date,Time,Status,Latitude,Longitude,WindSpeed,Ocean

HEIDI,19671019,1200, TD,20.5N,54.0W,25,Atlantic

OLAF,19850822,0, TD,12.9N,102.2W,25,Pacific

TINA,19920917,1200, TD,10.4N,98.5W,25,Pacific

EMMY,19760820,1200, TD,14.0N,48.0W,20,Atlantic

Each row is a line of the file

Basic Syntax

Table

Name	Date	Time	Status	Latitude	Longitude	WindSpeed	Ocean
HEIDI	19671019	1200	TD	20.5N	54.0W	25	Atlantic
OLAF	19850822	0	TD	12.9N	102.2W	25	Pacific
TINA	19920917	1200	TD	10.4N	98.5W	25	Pacific
EMMY	19760820	1200	TD	14.0N	48.0W	20	Atlantic

Corresponding CSV

Name,Date,Time,Status,Longitude,Latitude,WindSpeed,Ocean

HEIDI,19671019,1200,TD,20.5N,54.0W,25,Atlantic

OLAF,19850822,0,TD,12.9N,102.2W,25,Pacific

TINA,19920917,1200,TD,10.4N,98.5W,25,Pacific

EMMY,19760820,1200,TD,14.0N,48.0W,20,Atlantic

Cells...

Basic Syntax

Table



Name	Date	Time	Status	Latitude	Longitude	WindSpeed	Ocean
HEIDI	19671019	1200	TD	20.5N	54.0W	25	Atlantic
OLAF	19850822	0	TD	12.9N	102.2W	25	Pacific
TINA	19920917	1200	TD	10.4N	98.5W	25	Pacific
EMMY	19760820	1200	TD	14.0N	48.0W	20	Atlantic

Corresponding CSV

Name,Date,Time,Status,Longitude,Latitude,WindSpeed,Ocean
HEIDI,19671019,1200,TD,20.5N,54.0W,25,Atlantic
OLAF,19850822,0,TD,12.9N,102.2W,25,Pacific
TINA,19920917,1200,TD,10.4N,98.5W,25,Pacific
EMMY,19760820,1200,TD,14.0N,48.0W,20,Atlantic

... are separated by commas

Basic Syntax

Table

Name	Date	Time	Status	Latitude	Longitude	WindSpeed	Ocean
HEIDI	19671019	1200	TD	20.5N	54.0W	25	Atlantic
OLAF	19850822	0	TD	12.9N	102.2W	25	Pacific
TINA	19920917	1200	TD	10.4N	98.5W	25	Pacific
EMMY	19760820	1200	TD	14.0N	48.0W	20	Atlantic

Cor We call characters that act as separators “**delimiters**”

Name Date Time Status Latitude Longitude WindSpeed Ocean

HEIDI,19671019,1200,TD,20.5N,54.0W,25,Atlantic

Newlines delimit rows

OLAF

TINA,19920917,1200,TD,10.4N,98.5W,25,Pacific

The comma is a delimiter between cells in a row

EMMY,19760820,1200,TD,14.0N,48.0W,20,Atlantic

... are separated by commas

Advanced Syntax

We won't go into details here, but there are some complexities

Motivation for more complicated syntax

- *what if* a cell contains a newline?
- *what if* we want a comma inside a cell?
- *what if* a cell contains a quote?
- *what if* we want to use different delimiters between rows/cells?

usually better to use a general CSV module than roll your own

Today's Outline

Spreadsheets

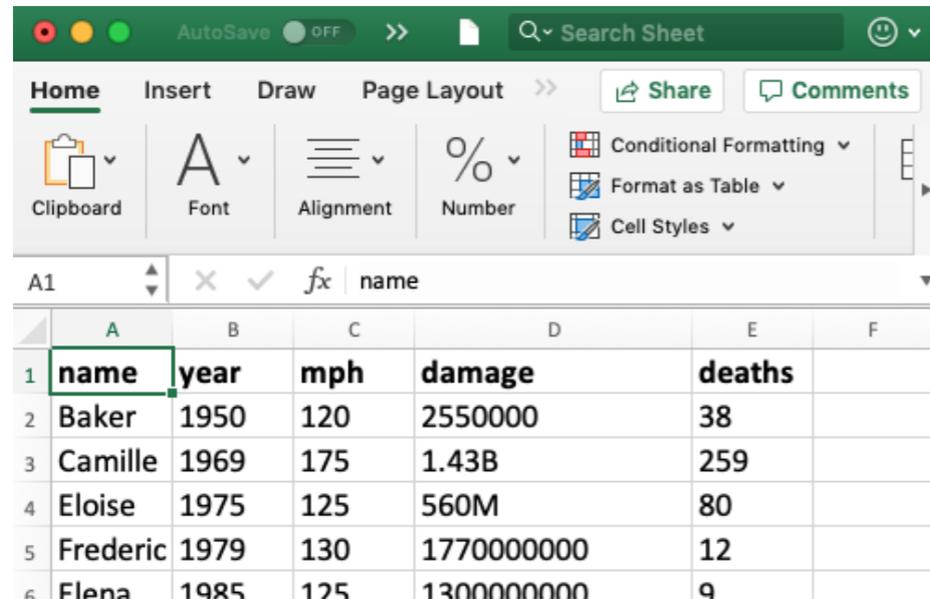
CSVs

Reading a CSV to a list of lists

Coding examples

Data Management

1. spreadsheet in Excel



The screenshot shows the Microsoft Excel interface. The ribbon is set to 'Home', and the 'Clipboard' group is active. The active cell is A1, containing the text 'name'. The spreadsheet data is as follows:

	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	



2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

Data Management

1. spreadsheet in Excel

name	year	mph	damage	deaths
Baker	1950	120	2550000	38
Camille	1969	175	1.43B	259
Eloise	1975	125	560M	80
Frederic	1979	130	1770000000	12
Elena	1985	125	1300000000	9

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program

list of lists

```
[
  ["name", "year", ...],
  ["Baker", "1950", ...],
  ...
]
```

Parsing Code

we'll come
back to this

Data Management

1. spreadsheet in Excel

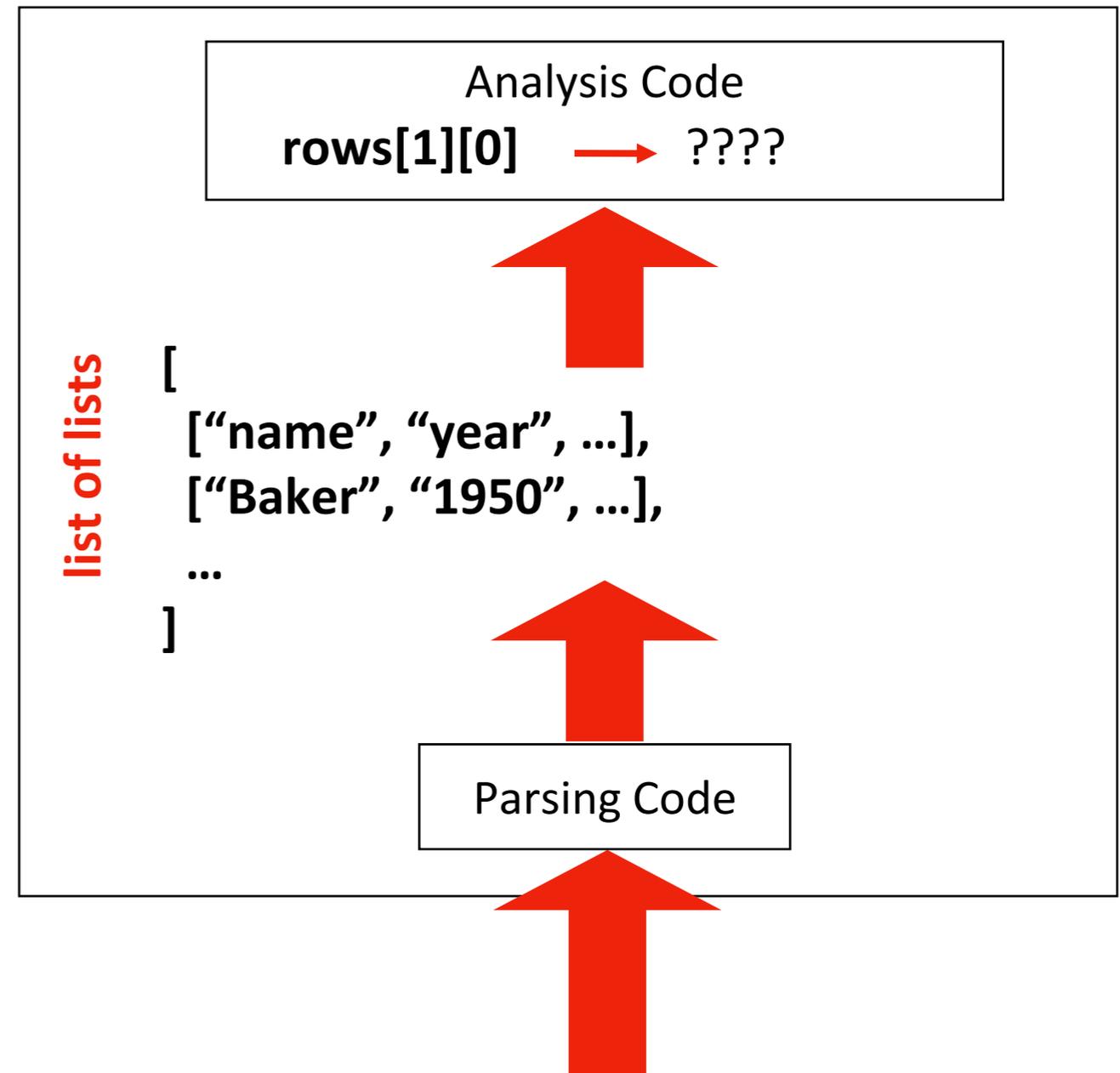
	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

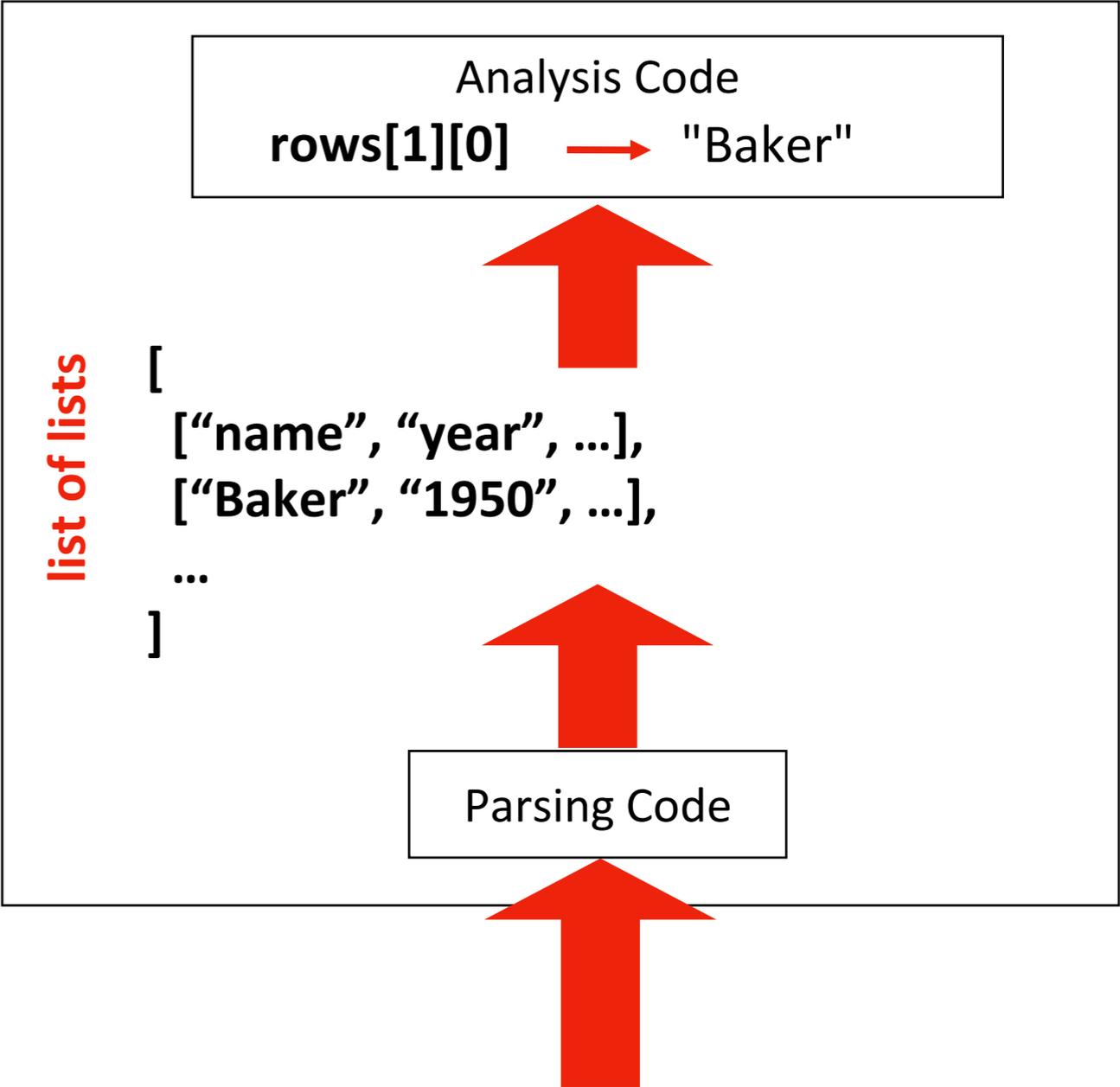
	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

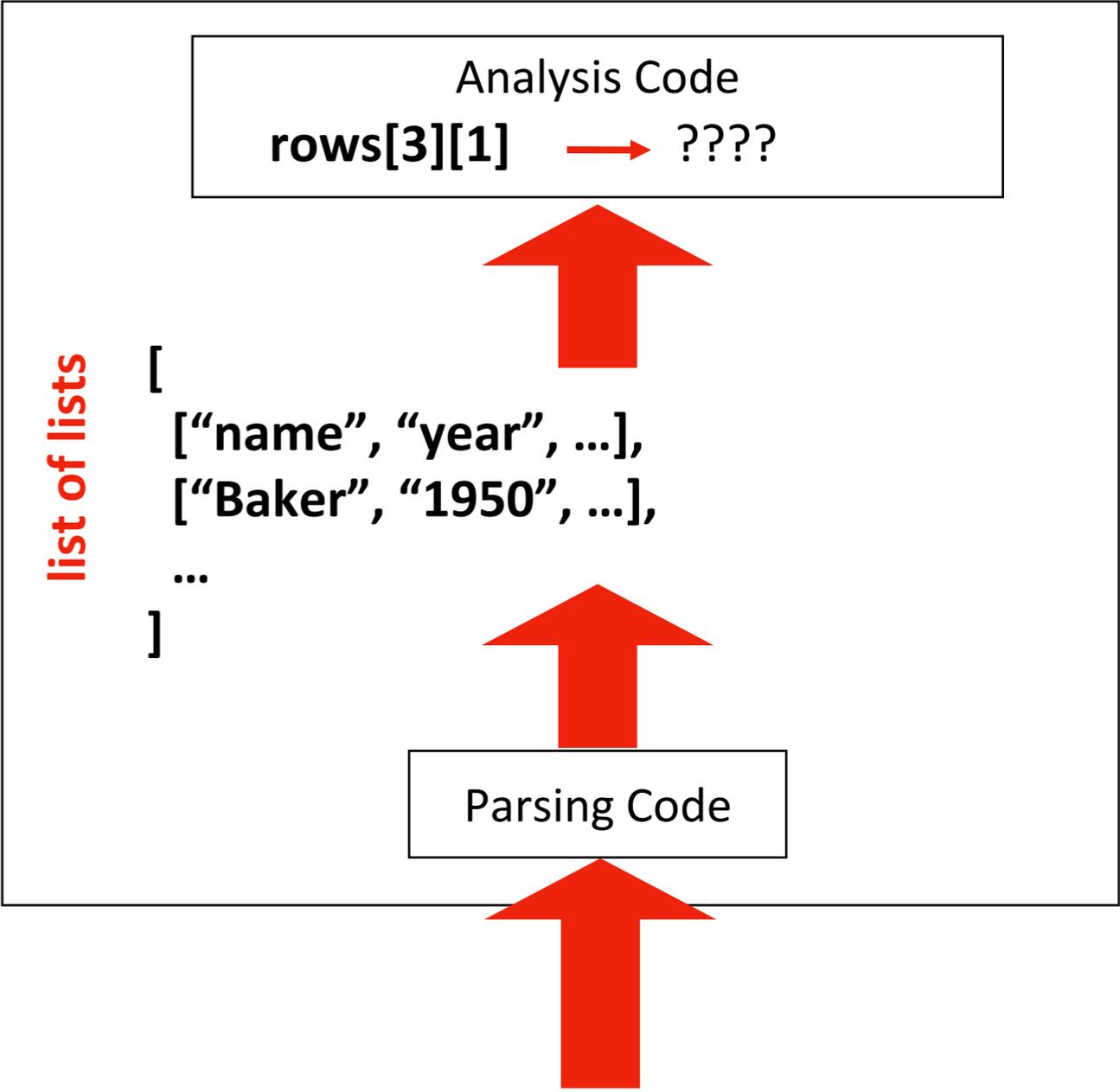
name	year	mph	damage	deaths
Baker	1950	120	2550000	38
Camille	1969	175	1.43B	259
Eloise	1975	125	560M	80
Frederic	1979	130	1770000000	12
Elena	1985	125	1300000000	9

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

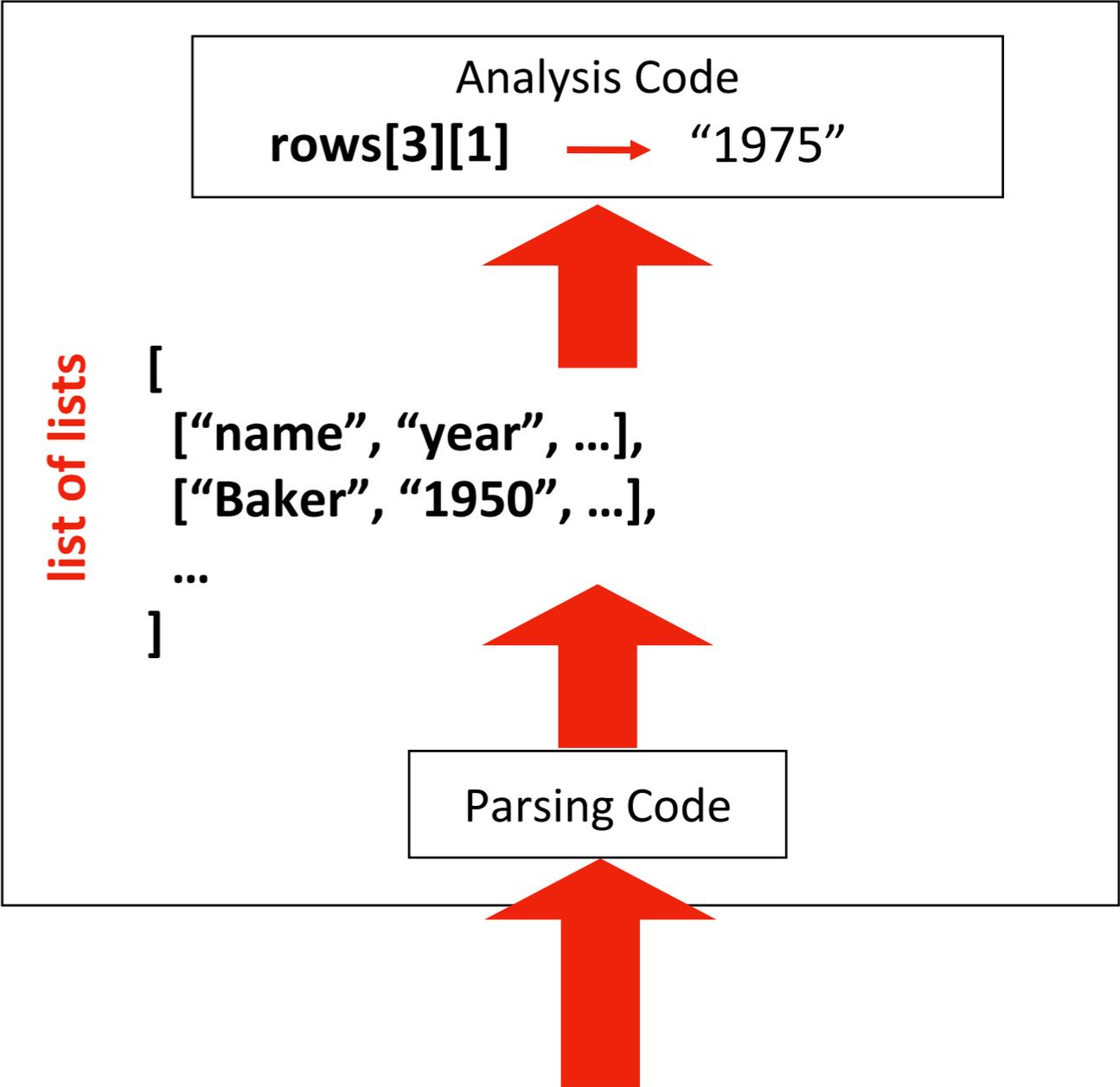
1	name	year	mph	damage	deaths
2	Baker	1950	120	2550000	38
3	Camille	1969	175	1.43B	259
4	Eloise	1975	125	560M	80
5	Frederic	1979	130	1770000000	12
6	Elena	1985	125	1300000000	9

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

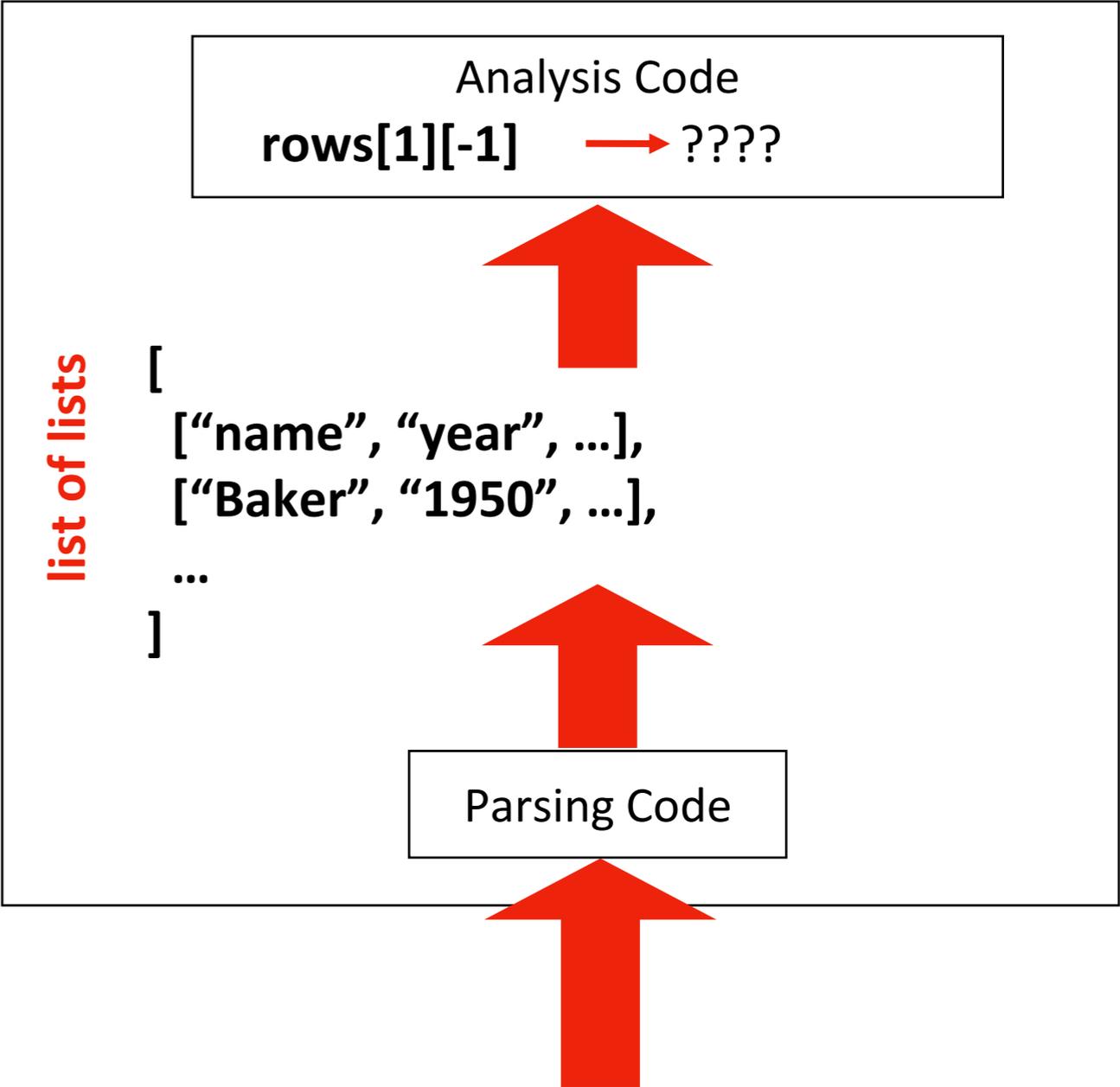
	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

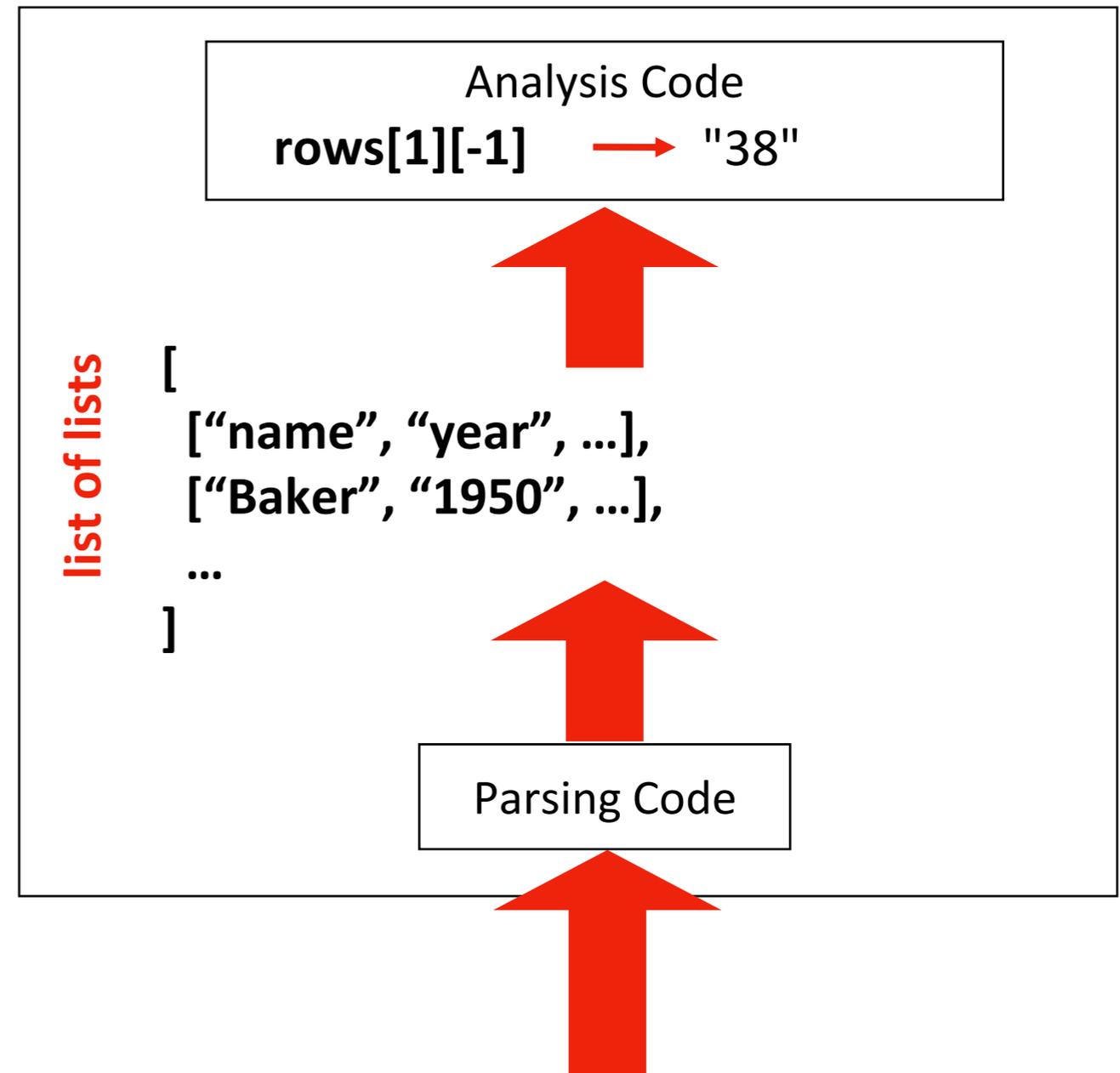
	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

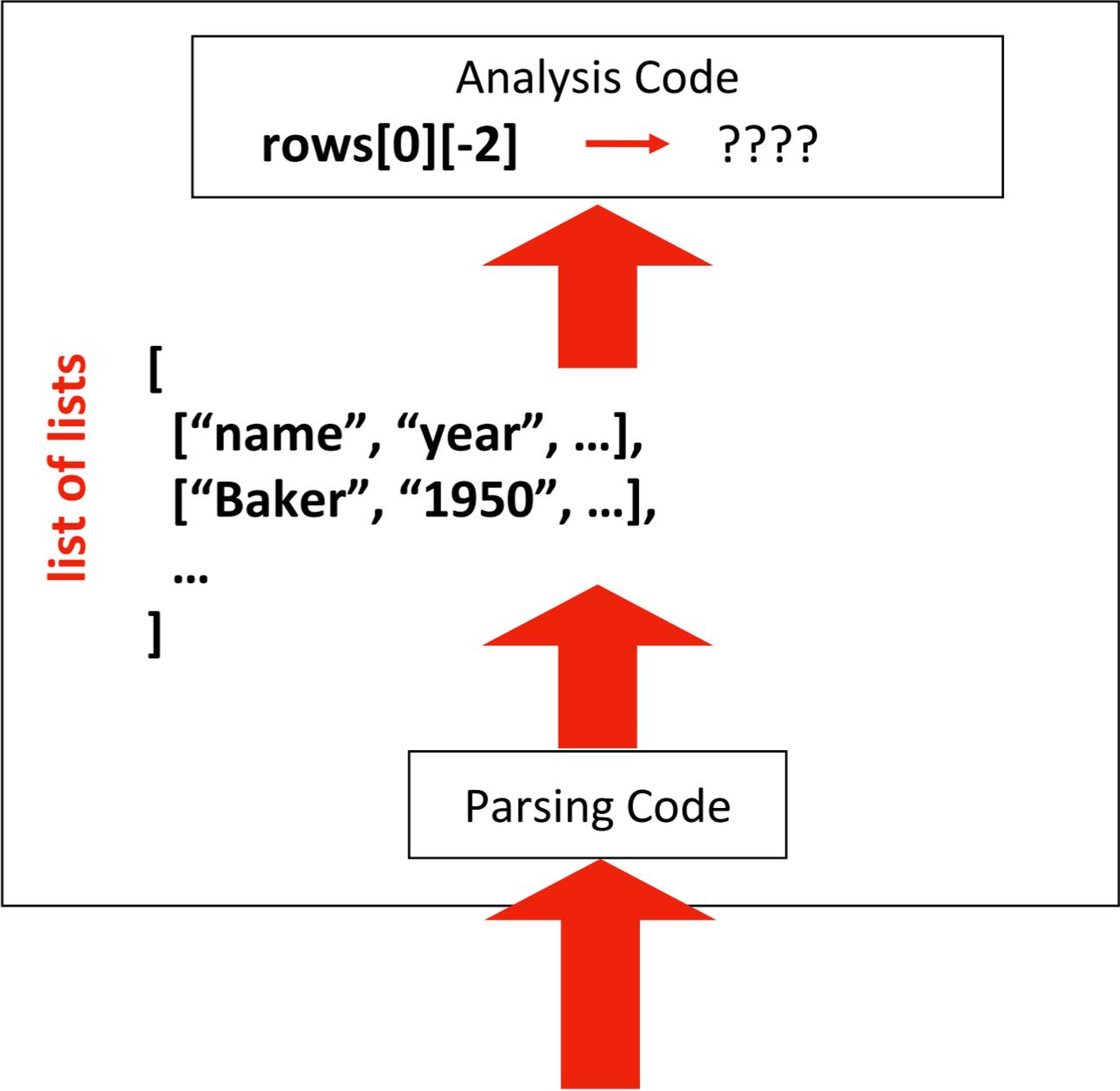
	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

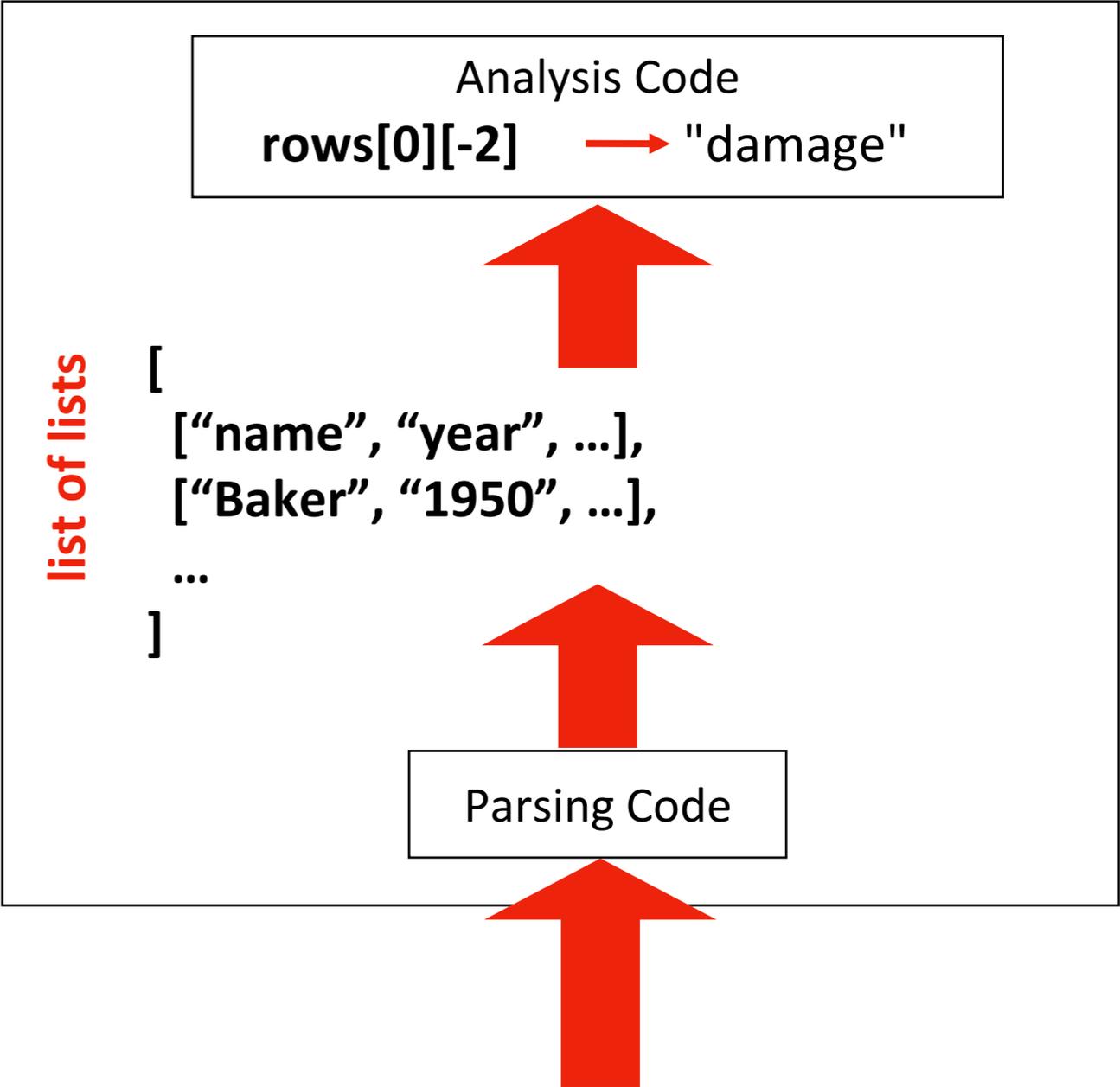
1	name	year	mph	damage	deaths
2	Baker	1950	120	2550000	38
3	Camille	1969	175	1.43B	259
4	Eloise	1975	125	560M	80
5	Frederic	1979	130	1770000000	12
6	Elena	1985	125	1300000000	9

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



Data Management

1. spreadsheet in Excel

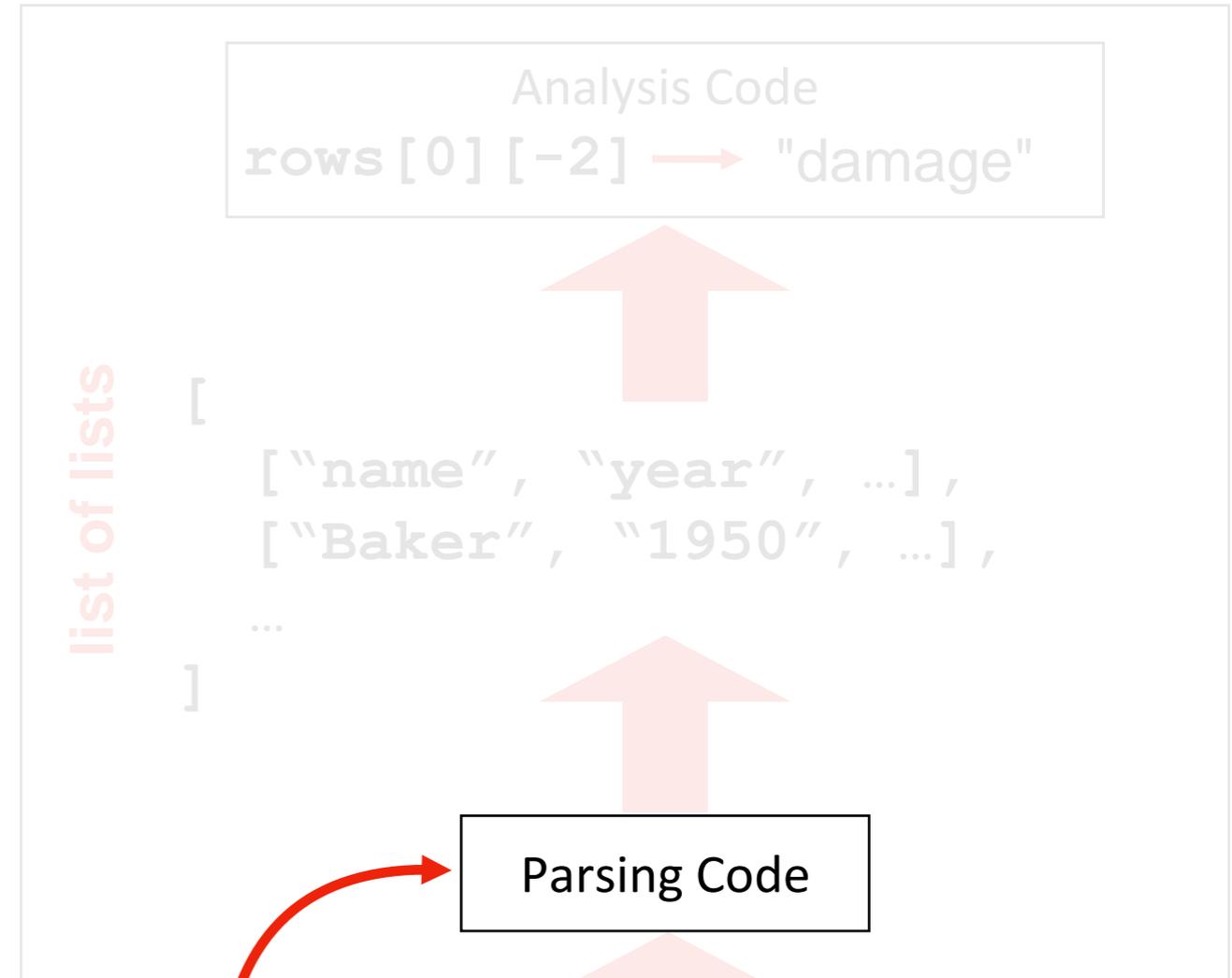
	A	B	C	D	E	F
1	name	year	mph	damage	deaths	
2	Baker	1950	120	2550000	38	
3	Camille	1969	175	1.43B	259	
4	Eloise	1975	125	560M	80	
5	Frederic	1979	130	1770000000	12	
6	Elena	1985	125	1300000000	9	

Save As
.CSV

2. CSV file saved somewhere

```
name,year,mph,damage,deaths
Baker,1950,120,2550000,38
Camille,1969,175,1.43B,259
Eloise,1975,125,560M,80
Frederic,1979,130,1770000000,12
```

3. Python Program



What does this look like?

Example Copied From Sweigart Ch 16

Code

```
import csv
exampleFile = open('example.csv')
exampleReader = csv.reader(exampleFile)
exampleData = list(exampleReader)
```

example.csv

```
4/5/2015 13:34,Apples,73
4/5/2015 3:41,Cherries,85
4/6/2015 12:46,Pears,14
4/8/2015 8:59,Oranges,52
4/10/2015 2:07,Apples,152
4/10/2015 18:10,Bananas,23
4/10/2015 2:40,Strawberries,98
```

Example Copied From Sweigart Ch 16

Code

```
import csv
exampleFile = open('example.csv')
exampleReader = csv.reader(exampleFile)
exampleData = list(exampleReader)
exampleData
```



**list of
lists**

```
[['4/5/2015 13:34', 'Apples', '73'], ['4/5/2015 3:41', 'Cherries', '85'],  
['4/6/2015 12:46', 'Pears', '14'], ['4/8/2015 8:59', 'Oranges', '52'],  
['4/10/2015 2:07', 'Apples', '152'], ['4/10/2015 18:10', 'Bananas', '23'],  
['4/10/2015 2:40', 'Strawberries', '98']]
```

Example Copied From Sweigart Ch 16

```
import csv
exampleFile = open('example.csv')
exampleReader = csv.reader(exampleFile)
exampleData = list(exampleReader)
exampleData
```

let's generalize this to a function
(don't need to know exactly how the code works, though we will eventually)

Example Copied From Sweigart Ch 16

```
import csv
exampleFile = open('example.csv')
exampleReader = csv.reader(exampleFile)
exampleData = list(exampleReader)
exampleData
```

input

output

let's generalize this to a function
(don't need to know exactly how the code works, though we will eventually)

Example Copied From Sweigart Ch 16

```
def process_csv():  
    import csv  
    exampleFile = open('example.csv')  
    exampleReader = csv.reader(exampleFile)  
    exampleData = list(exampleReader)  
    exampleData
```

1. move code to a function

Example Copied From Sweigart Ch 16

```
import csv

def process_csv():
    import csv
    exampleFile = open('example.csv')
    exampleReader = csv.reader(exampleFile)
    exampleData = list(exampleReader)
    exampleData
```

2. move out imports

Example Copied From Sweigart Ch 16

```
import csv

def process_csv():
    import csv
    exampleFile = open('example.csv')
    exampleReader = csv.reader(exampleFile)
    exampleData = list(exampleReader)
    return exampleData
```

3. return data to get it out of the function

Example Copied From Sweigart Ch 16

```
import csv

def process_csv():
    import csv
    exampleFile = open('example.csv')
    exampleReader = csv.reader(exampleFile)
    exampleData = list(exampleReader)
    return exampleData
```

4. generalize input

Example Copied From Sweigart Ch 16

```
import csv

def process_csv(filename):
    import csv
    exampleFile = open(filename)
    exampleReader = csv.reader(exampleFile)
    exampleData = list(exampleReader)
    return exampleData
```

4. generalize input

Example Copied From Sweigart Ch 16

```
import csv
```

```
# copied from https://automatetheboringstuff.com/2e/chapter16/  
def process_csv(filename):  
    import csv  
    exampleFile = open(filename)  
    exampleReader = csv.reader(exampleFile)  
    exampleData = list(exampleReader)  
    return exampleData
```

Reminder!
cite code
copied online

5. cite the code

Example Copied From Sweigart Ch 16

```
import csv

# inspired by https://automatetheboringstuff.com/2e/chapter16/
def process_csv(filename):
    example_file = open(filename, encoding="utf-8")
    example_reader = csv.reader(example_file)
    example_data = list(example_reader)
    example_file.close()
    return example_data
```

keep this handy for copy/paste

Today's Outline

Spreadsheets

CSVs

Reading a CSV to a list of lists

Coding examples

Example: Student Information Survey

Goal: find the average age of the students, for each lecture

Input:

- Student data (a CSV file)

Output:

- Average student age for a given lecture

Goal: column name, print that data for all hurricanes

Example:

LEC001: 18.5

LEC002: 18.2

LEC003: 18.6

...

Challenge: Hurricane Column Dump

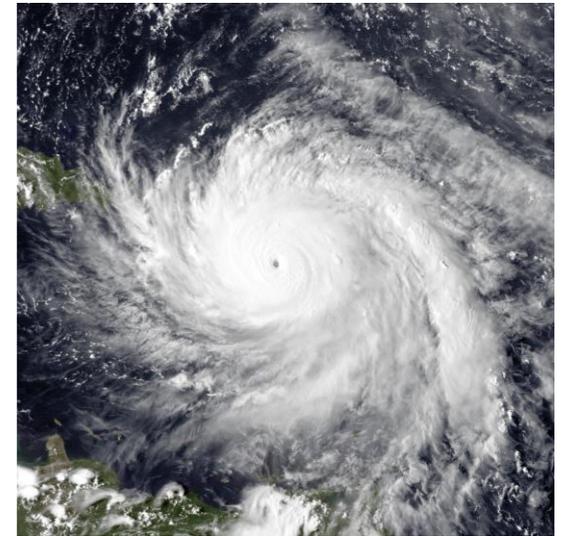
Goal: column name, print that data for all hurricanes

Input:

- column name (and a CSV file)

Output:

- data in given column, associated with name



Example:

Baker: 1950

Camille: 1969

Eloise: 1975

...

Challenge: Hurricanes per Year

Goal: column name, print that data for all hurricanes

Input:

- none typed (only a CSV file)

Output:

- the number of hurricanes in each year

Example:

1967: 23

1968: 29

2969: 15

...

