2. Using one line of code, count how ma... ⌵ Points: 0/0

2. Using one line of code, count how many families are represented in the file `Pacifici2013_data.csv`. Begin with `cut`, mind the spaces, and use as few options as possible. Paste your code below:

✓	cut -d ";" -f3 Pacifici2013_data.csv | tail -n +2 | sort | uniq | wc -l	**2 responses**	0	/ 0 pts

yes

✗	53	**1 response**	0	/ 0 pts

Paste your code below:

✗	cut - d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | sort | uniq | wc -l	**1 response**	0	/ 0 pts

yes

✗	cut -d ";" -f 1 Pacifici2013_data.csv	**1 response**	0	/ 0 pts

good job with the cut option -d, but families aren't in col 1

✗	cut -d ";" -f 2 Pacifici2013_data.csv | tail -n +2 | sort | uniq -c	**1 response**	0	/ 0 pts

very close but uniq -c does not answer question and families aren't in col 2

✗	cut -d ";" -f 2 Pacifici2013_data.csv | tail -n+2 | sort | uniq -c | wc -l	**1 response**	0	/ 0 pts

very close but families are not in col 2

✕ cut -d ";" -f 2 Pacifici2013_data.csv | wc -l **1 response** 0 / 0 pts

close, but this doesn't count the uniq families and col 2 does not contain families

✕ cut -d ";" -f 3 Pacifici2013_data.csv **1 response** 0 / 0 pts

this is a good start, keep going

✕ cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | sort |\uniq -c | wc -l **1 response** 0 / 0 pts

very close, no `\` before uniq -c

✕ cut -d ";" -f 3 Pacifici2013_data.csv | sort | uniq | grep -c idae **1 response** 0 / 0 pts

very very close use wc -l rather than grep -c there is a header row that needs to be removed, or else your result will be 1 too high

✕ cut -d ";" -f 3 Pacifici2013_data.csv | sort | uniq | wc -l **1 response** 0 / 0 pts

very close, but you have to remove the header row

✕ cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | sort | uniq | wc -l **1 response** 0 / 0 pts

yes

✕ cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | sort | uniq | wc **1 response** 0 / 0 pts yes

× `cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | sort | uniq | wc -1` 1 response 0 / 0 pts

very close, but it's `wc -l` that a -L lowercase

× `cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | sort | uniq | wc -l` 1 response 0 / 0 pts

yes, great job!

× `cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | uniq` 1 response 0 / 0 pts

you're on the right track, always use `sort` before `uniq` and then you need to count up the number of lines

× `cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | uniq | wc -l` 2 responses 0 / 0 pts

might work but get into habit of sorting before uniq

× `cut -d ";" -f 3 Pacifici2013_data.csv | tail -n +2 | wc -l` 2 responses 0 / 0 pts

close, but this counts all records rather than number of families

× `cut -d ";" -f 3 Pacifici2013_data.csv | uniq | wc -l` 1 response 0 / 0 pts

close, but this will count the header row. also, should always sort before uniq

× `cut -d ";" -f 5 Pacifici2013_data.csv | tail -n +2 | wc -l` 1 response 0 / 0 pts

close, but this will count the header row. also, families don't occur in col 5

× cut -d ";" -f2-6 ../data/Pacifici2013_data.csv | \ **1 response** 0 / 0 pts

good start, don't stop probably want to delete the \

× cut -d ";" -f3 Pacifici2013_data.csv | sort | uniq -c | wc -l **1 response** 0 / 0 pts

yes

× cut -d";" -f3 pacifici2013_data.csv | sort | uniq | wc -| **1 response** 0 / 0 pts

very close, but it's `wc -l` that's a lower case L not the pipe symbol `|`

× cut -f 3 Pacifici2013_data.csv | tail -n +2 | wc **1 response** 0 / 0 pts

this will count all rows use option with wc to narrow down output