

# Poetry and Linting

# Poetry is a virtual environment management system

- interact with poetry on the command line

```
poetry shell
```

```
poetry install
```

```
pip -V
```

```
deactivate
```



# How to tell you are in a virtual environment

Look for (environment name) at the beginning of the command prompt

```
(square-py3.11) egraber ~/Documents/Teaching/S2024-CMPSC101/integer-  
squaring-starter/square % 🌲🌲🌲 pip -V  
pip 23.2.1 from /Users/egraber/Library/Caches/pypoetry/virtualenvs/s  
quare-lxZ0blbv-py3.11/lib/python3.11/site-packages/pip (python 3.11)  
(square-py3.11) egraber ~/Documents/Teaching/S2024-CMPSC101/integer-  
squaring-starter/square % 🌲🌲🌲 deactivate  
egraber ~/Documents/Teaching/S2024-CMPSC101/integer-squaring-starter  
/square % 🌲🌲🌲
```

# Common Issues

Poetry commands will not work unless the pyproject.toml file is accessible

```
(square-py3.11) egraber ~/Documents/Teaching/S2024-CMPSC101/i  
ninteger-squaring-starter %  ls  
README.md config square writing  
(square-py3.11) egraber ~/Documents/Teaching/S2024-CMPSC101/i  
ninteger-squaring-starter %  poetry install
```

**Poetry could not find a pyproject.toml file in /Users/egraber  
/Documents/Teaching/S2024-CMPSC101/integer-squaring-starter o  
r its parents**

```
(square-py3.11) egraber ~/Documents/Teaching/S2024-CMPSC101/i  
ninteger-squaring-starter %  
```

# Common Issues

You should be working with the virtual environment activated

You should never need to type `pip install ...`

Everything that is needed for all the projects will be available in the virtual environments

Everything that is needed ==  
Dependencies!

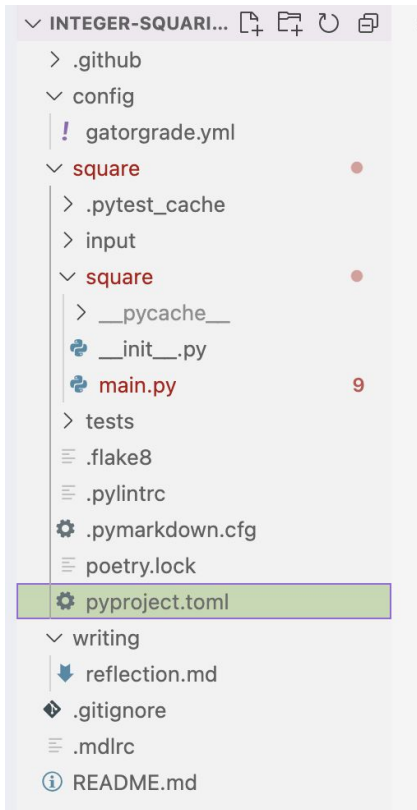
# Dependencies

Rich

Typer

many more...

# Find files in VS code with ctrl+p (windows) cmd+p (mac)



## The pyproject.toml file

- lists dependencies with minimum version
- creates convenient shortcut commands

```
15 [tool.poetry.dev-dependencies]
16 taskipy = "^1.8.1"
17 flake8 = "^3.9.2"
18 pydocstyle = "^6.1.1"
19 pylint = "^2.10.2"
20 mypy = "^0.910"
21 isort = "^5.10.1"
22 black = "^22.6.0"
23 pymarkdownlint = "^0.9.7"
24 pytest = "^7.2.1"
25
```

```
--
26 [tool.taskipy.tasks]
27 black = { cmd = "black square tests --check", help = "Run the black checks for source code format" }
28 fiximports = { cmd = "isort square tests", help = "Run isort to fix source code imports" }
29 fixformat = { cmd = "black square tests", help = "Run the black checks for source code format" }
30 isort = { cmd = "isort -c square tests", help = "Run the isort checks for source code" }
31 flake8 = { cmd = "flake8 square tests", help = "Run the flake8 checks for source code documentation" }
32 markdownlint = { cmd = "poetry run pymarkdown --config .pymarkdown.cfg scan ../writing/reflection.md ../README.md", help =
33 mypy = { cmd = "poetry run mypy square", help = "Run the mypy type checker for potential type errors" }
34 pydocstyle = { cmd = "pydocstyle square tests", help = "Run the pydocstyle checks for source code documentation" }
35 pylint = { cmd = "pylint square tests", help = "Run the pylint checks for source code documentation" }
36 test = { cmd = "pytest -x -s", help = "Run the pytest test suite" }
37 test-silent = { cmd = "pytest -x --show-capture=no", help = "Run the pytest test suite without showing output" }
38 all = "task black && task isort && task flake8 && task markdownlint && task mypy && task pydocstyle && task pylint && task
39 lint = "task black && task isort && task flake8 && task markdownlint && task mypy && task pydocstyle && task pylint"
40
```

# pyproject.toml != poetry.lock

poetry.lock is derived from pyproject.toml

poetry creates the poetry.lock file as it figures out specific versions of everything that are compatible.

- if you see any errors that poetry.lock is incompatible with your system, you can delete it and run `poetry install` again



# Back to convenient shortcuts

poetry run task ...

```
26 [tool.taskipy tasks]
27 black = { cmd = "black square tests --check", help = "Run the black checks for source code format" }
28 fiximports = { cmd = "isort square tests", help = "Run isort to fix source code imports" }
29 fixformat = { cmd = "black square tests", help = "Run the black checks for source code format" }
30 isort = { cmd = "isort -c square tests", help = "Run the isort checks for source code" }
31 flake8 = { cmd = "flake8 square tests", help = "Run the flake8 checks for source code documentation" }
32 markdownlint = { cmd = "poetry run pymarkdown --config .pymarkdown.cfg scan ../writing/reflection.md ../README.md", help =
33 mpy = { cmd = "poetry run mypy square", help = "Run the mypy type checker for potential type errors" }
34 pydocstyle = { cmd = "pydocstyle square tests", help = "Run the pydocstyle checks for source code documentation" }
35 pylint = { cmd = "pylint square tests", help = "Run the pylint checks for source code documentation" }
36 test = { cmd = "pytest -x -s", help = "Run the pytest test suite" }
37 test-silent = { cmd = "pytest -x --show-capture=no", help = "Run the pytest test suite without showing output" }
38 all = "task black && task isort && task flake8 && task markdownlint && task mypy && task pydocstyle && task pylint && task
39 lint = "task black && task isort && task flake8 && task markdownlint && task mypy && task pydocstyle && task pylint"
```

- poetry run black path/to/file
- poetry run black .



# Try them all, see what happens!

poetry run task \_\_\_

```
26 [tool.taskipy.tasks]
27 black = { cmd = "black square tests --check", help = "Run the black checks for source code format" }
28 fiximports = { cmd = "isort square tests", help = "Run isort to fix source code imports" }
29 fixformat = { cmd = "black square tests", help = "Run the black checks for source code format" }
30 isort = { cmd = "isort -c square tests", help = "Run the isort checks for source code" }
31 flake8 = { cmd = "flake8 square tests", help = "Run the flake8 checks for source code documentation" }
32 markdownlint = { cmd = "poetry run pymarkdown --config .pymarkdown.cfg scan ../writing/reflection.md ../README.md", help =
33 mpy = { cmd = "poetry run mypy square", help = "Run the mypy type checker for potential type errors" }
34 pydocstyle = { cmd = "pydocstyle square tests", help = "Run the pydocstyle checks for source code documentation" }
35 pylint = { cmd = "pylint square tests", help = "Run the pylint checks for source code documentation" }
36 test = { cmd = "pytest -x -s", help = "Run the pytest test suite" }
37 test-silent = { cmd = "pytest -x --show-capture=no", help = "Run the pytest test suite without showing output" }
38 all = "task black && task isort && task flake8 && task markdownlint && task mypy && task pydocstyle && task pylint && task
39 lint = "task black && task isort && task flake8 && task markdownlint && task mypy && task pydocstyle && task pylint"
```

- poetry run \_\_\_ path/to/file
- poetry run \_\_\_ .



# poetry run task black

```
(square-py3.11) egraber ~/Documents/Teaching/F2023-CMPSC101/Week02-python-foundation
integer-squaring-internal-emgraber/square % 🌲🌲🌲 poetry run task black
would reformat square/myreadfile.py
```

Oh no! 💣 💔 💣

**1 file** would be reformatted, 4 files would be left unchanged.

```
(square-py3.11) egraber ~/Documents/Teaching/F2023-CMPSC101/Week02-python-foundation
integer-squaring-internal-emgraber/square % 🌲🌲🌲 █
```

# poetry run black path/to/file

```
(square-py3.11) egraber ~/Documents/Teaching/F2023-CMPSC101/Week02-python-foundations/i  
ninteger-squaring-internal-emgraber/square % 🌳🌳🌳 poetry run black square/myreadfile.py
```

reformatted square/myreadfile.py

All done! ✨ 🍰 ✨

**1 file** reformatted.

```
(square-py3.11) egraber ~/Documents/Teaching/F2023-CMPSC101/Week02-python-foundations/i  
ninteger-squaring-internal-emgraber/square % 🌳🌳🌳 █
```

poetry run task isort

poetry run isort .

keep trying them!!!!

# Linting

Linters check for professional coding styles and practices

- unused variables, extra blank spaces, too many/few blank lines

Why?

- In English
- hi There,    my naMe naMe is::: Emily .

# Linting

Linters check for professional coding styles and practices

- unused variables, extra blank spaces, too many/few blank lines
- `gatorgrade --config config/gatorgrade`

- ✓ Pass the source code formatting checks run by black
- ✗ Pass the linting checks run by flake8
- ✓ Pass the import checks run by isort
- ✗ Pass the linting checks run by mypy
- ✗ Pass the linting checks run by pydocstyle
- ✗ Pass the linting checks run by pylint
- ✓ Pass the linting checks run by pymarkdown
- ✓ Pass all of the Pytest test cases
- ✓ Have at least a specific minimum of commits in repository

# DON'T GIVE UP

- pass/fail assignments
- graded assignments 1/6 is professional skills like linting

```
=====
ERROR collecting tests/test_square.py
/Users/egraber/Library/Caches/pypoetry/virtualenvs/square-lxZ0.../lib/python3.11/site-packa
mod = import_path(self.path, mode=importmode, root=self.path)
/Users/egraber/Library/Caches/pypoetry/virtualenvs/square-lxZ0.../lib/python3.11/site-packa
importlib.import_module(file_name)
/opt/homebrew/Cellar/python@3.11/3.11.5/Frameworks/Python.framework/Versions/3.11/lib/python3.11/i
return _bootstrap._gcd_import(name[level:], package, level)
importlib._bootstrap_000
=====
E
E
E
:
=====
summary
=====
tests/test_squa
!!!!!!!!!!!!!!!!!!!! after 1 failures !!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!! 1 error during collection !!!!!!!!!!!!!!!!!!!!!
=====
1 error in 0.07s =====
x Have at least a specific number of commits in repository
→ Found 3 commit(s) in the repository
```



# Strategies

- read the errors at the bottom!
- actually read them
- really
- yes!
- right click on the line numbers
- ignore files that are deep in your computer system
- get vs code extensions for markdown

✗ Pass the linting checks run by pymarkdown

→ ../README.md:13:1: MD013: Line length [Expected: 80, Actual: 114] (line-length)

../README.md:21:1: MD013: Line length [Expected: 80, Actual: 102] (line-length)

../README.md:70:1: MD013: Line length [Expected: 80, Actual: 152] (line-length)

✗ Pass all of the Pytest test cases

→ ===== test session starts =====

platform darwin -- Python 3.11.5, pytest-7.4.1, pluggy-1.3.0

rootdir: /Users/egraber/Documents/Teaching/S2024-CMPSC101/integer-squaring-starter/square

collected 0 items / 1 error