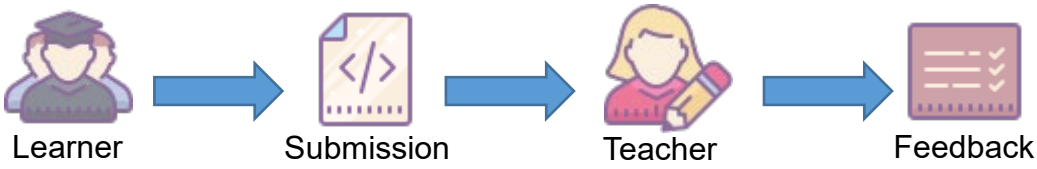
Pedal

Pedagogical Feedback Library for Python

https://pedal-edu.github.io



Feedback Is Hard!

According to a 2019 study by Denny et al, the #1 question among computing teachers is:

"How and when is it best to give students feedback on their code to improve learning?"

Do you need to…?

- **Check** more precise conditions and patterns than just input/output
- **Revise** a hundred lines of complicated, dependent unit tests?
- **Track** what feedback was triggered for your students?
- **Regrade** your students' submissions with a changed grading script?

Pedal Makes It Easier

Pedal has a wide set of features:

- Structurally identify student mistakes using Python Code
- Reusable collection of student misconceptions
- Unit testing functions in classic assertion style
- Sophisticated sandboxing and mocking
- Type checking and flow analysis

And is meant for teachers:

- Compatible with autograding platforms that support pure Python
- Successfully deployed in BlockPy, Jupyter, VPL, and more!
- Fully open-source, completely free

Get in touch!

Luke Gusukuma (lukesg08@vt.edu) and Cory Bart (acbart@udel.edu)

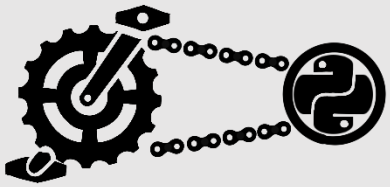
Join us on GitHub!

https://github.com/pedal-edu/pedal

Pedal

Pedagogical Feedback Library for Python

https://pedal-edu.github.io



No fuss installation

```
pip install pedal
```

Minimal boilerplate

```
from pedal import *
```

Rich feedback primitives with metadata

```
gently("You have the wrong output", label="wrong_output")
compliment("Great progress!", score="+10%")
suppress("runtime")
set_success()
```

Enhanced error messages through our flow and type analyzer

TypeError

unsupported operand type(s) for +: 'int' and 'str'

Incompatible types

You used an addition operation with a number and a string on line 2. But you can't do that with that operator. Make sure both sides of the operator are the right type.

Detect structural mistakes with declarative, wildcard searches

```
if find_matches("answer = 42"):
    explain("You may not simply embed the answer.")
if not find_matches("for _item_ in ___: _item_"):
    gently("You're not using the iteration variable")
prevent_operation("/")
ensure_ast("If")
```

Sophisticated sandboxing with contextualized errors

```
block_function('sum')
assert_equal(call('add_up', [1,2,3]), 6)
clear_output()
assert_output(run(inputs=['hello']), "Hello")
ensure_called_uniquely('add_up', at_least=3)
```

I ran:

```
add_up([1, 2, 3])
```

But your code had the following error on line 17...

All your classic assertions, plus so much more!

```
ensure_function("distance", parameters=(int, int), returns=int)
unit_test("distance", [ ((3, 6), 3), ((1, 7), 6) ])
ensure_documented_functions()
ensure_coverage(.75)
assert_plot("histogram", [1, 2, 3, 4])
```