Outline

specialize software

raising exceptions
specialize flexibly

If we decided to extend the features of Stack, what’s wrong with:

- modifying the existing Stack?
- cut-paste-modify Stack $\rightarrow$ MyStack?
- include Stack attribute in new classes
we subclass (extend) a superclass (base class) by:

- declaring that we’re extending it...

```python
class NewClass(OldClass):
...
```

- add methods and attributes to specialize

- other methods and attributes are searched for in superclass
override versus extend

3 ways

you may replace or modify old code

- subclass method with the same name replace superclass method

- access superclass method with
  `OldClass.method(self, ...)`

- `__init__` is a special case — careful
Return types are not appropriate in all cases.

There are no special values!

- What's wrong with IntStack returning a "special" integer for pop-on-empty?

- Push usually has return type None, but what if stuff happens?

- What if the calling code doesn't know what to do?
cause existing Exceptions:

- `int("seven")`  
  value error

- `a = 1/0`  
  div by zero

- `[1, 2][2]`  
  index
raise existing Exceptions:

- raise ValueError or...
  - raise ValueError('you can’t do that!')
roll your own Exceptions:

- class ExtremeException(Exception):
  pass

- raise ExtremeException

- raise ExtremeException('I really object')
roll your own Exceptions:

- class ExtremeException(Exception):
  pass

- raise ExtremeException

- raise ExtremeException('I really object')