Exception Handling

Object Oriented System Design

Otago Polytechnic
Dunedin, New Zealand

Introduction

- Things can go wrong in our programs.
- Consider ex1.py in the example code on GitHub.
- You can anticpate ways to break it before you even run it.
- Run it anyway. Break it and see what happens.

Exceptions

- When something goes wrong in a Python program it *raises an exception*.
- If the exception isn't handled the program terminates.
- Sometimes this is unavoidable or even desirable, but we don't want this to happen all the time.

Guarding against exceptions

- When we're programming we can usually identify places in our code where exceptions are likely to occur.
- Consider ex2.py. We use an `if-elif-else` structure to guard against likely problems. Run it and see if you can get it to crash.[1]
- This version is far more robust, but there is a cost. There are only three lines of code that do what we really want mixed in with five lines of code that protect those three.

[1] I can.

Exception handling

- Python, like many languages, allows you to handle excetions in your code and avoid crashing.
- Consider ex3.py
- We still have more error handling code than “normal” code, but we have seperated the two so that the main code path is clear.

Exception handling - mixed approach

- It's best to use exception handling for cases that are truly *exceptional*.
- Consider ex4.py. We use a `while` to handle the more common user error. We use `try/except` to handle what we think is a less common error.

Practical exercise

1. Write a function called `oops` that explicitly raises an `IndexError` exception when called. Then write another function that calls oops inside a `try/except` statement to catch the error. What happens if you change oops to raise `KeyError` instead of `IndexError`?
2. Change the `oops` function you just wrote to raise an exception you define yourself, called `MyError`, and pass an extra data item along with the exception. Then, extend the `try` statement in the catcher function to catch this exception and its data in addition to `iIndexError`, and print the extra data item.
3. Modify your `MyError` exception class so that it inherits from `IndexError` instead of `Exception`. What happens when it is thrown?

Save your code examples in a directory called `exceptions` and push it to your GitHub repository.