Exceptions and Unit Testing

CSCI 1030U - Intro to Computer Science
@IntroCS

Randy J. Fortier
@randy_fortier

Outline

- Exceptions
  - Catching exceptions
  - Throwing exceptions
  - Custom exceptions
- Testing
  - Unit testing

Exceptions

Exceptions

- Exceptions are run-time errors
  - Exceptions are often raised when performing input and output (e.g. socket communication)
  - You can catch these errors, to prevent it stopping your program
  - You can define your own exceptions for your program’s needs

Catching Exceptions

- Example:

```
try:
    # do something that could raise an exception
except SomeError as err:
    print('Error:  ', err)
```

Raising Exceptions

- Example:

```
class MyCustomError(Exception):
    pass

...
raise MyCustomError('Error message.')
...
```

Programming Exercise 08b.1

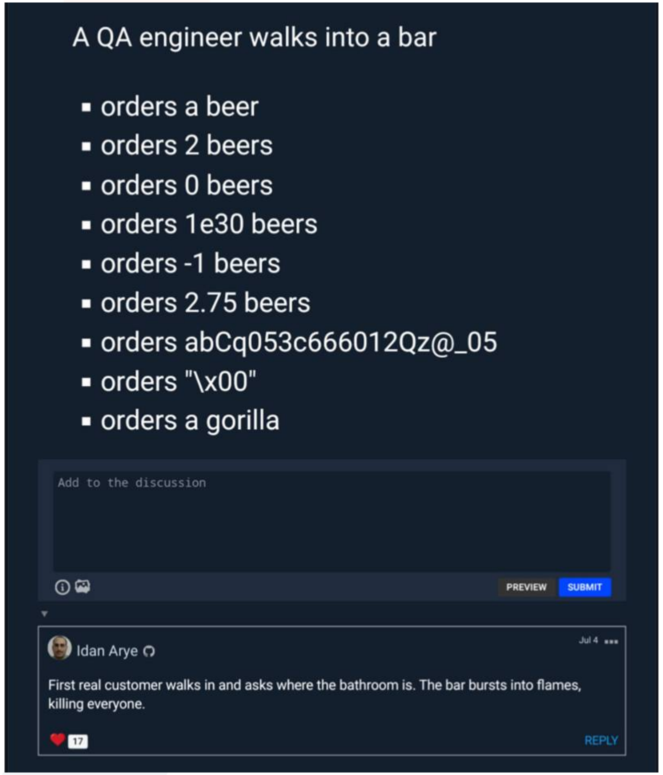
- Write some code to output `1/n` for all `n` in the list `[5,4,3,2,1,0]`
  - Be sure to catch the exception that will be generated for `1/0`

Programming Challenge 08b.1

- Write some code to ask the user their age, and raise a custom exception if they are not yet 18 years old

Testing

Quality Assurance


A QA engineer walks into a bar

- orders a beer
- orders 2 beers
- orders 0 beers
- orders 1e30 beers
- orders -1 beers
- orders 2.75 beers
- orders abCq053c666012Qz@_05
- orders "\x00"
- orders a gorilla

First real customer walks in and asks where the bathroom is. The bar bursts into flames, killing everyone.

Quality Assurance

- Functional tests:
  - Unit Testing
  - Integration Testing
  - System Testing
- Non-functional tests:
  - Performance/load/stress Testing
  - Security Testing
  - Usability Testing
  - Localization Testing

Unit Tests

- Either classes or functions could be considered a unit (of modularity)
  - Unit tests isolate one of these units and test them thoroughly
  - Unit test frameworks, such as *unittest* for Python, are designed to make this easy to do
    - A test is a method which makes one or more *assertions*, which generate a report (and often terminate the program) when they fail
    - Most unit testing frameworks (e.g. JUnit, NUnit, Boost.Test) work in a very similar way

How to Write Good Unit Tests

- How much of your program is measured by a metric called *code coverage*
  - Code coverage of 100% means that every aspect of your code is being tested
- How do you increase your code coverage?
  - Use a range of inputs that will cause your program's different execution paths to be followed
    - e.g. check both True and False for conditionals

Unit Testing in Python

- A unittest test class inherits from `unittest.TestCase`
- Each method in that class starts with `test_` and is considered a separate test

```
class Pet_Test(unittest.TestCase):
    def test_speak(self):
        pet1 = Pet('Cat', 'Whiskers')
        self.assertEqual(pet1.speak(), 'Meow!')

        pet2 = Pet('Dog', 'Spike')
        self.assertTrue(pet2.speak() == 'Woof!')
```

https://docs.python.org/3/library/unittest.html

Programming Exercise 08b.2

- Write a test class to test the following class:

```
class Student:
    def __init__(self, gpa, name):
        self.gpa = gpa
        self.name = name

    def set_mark(self, course, mark):
        self.marks.append(mark)

    def get_average(self):
        sum = 0
        for mark in self.marks:
            sum += mark
        return sum / len(self.marks)
```

Wrap-up

- Exceptions
  - Catching exceptions
  - Throwing exceptions
  - Custom exceptions
- Testing
  - Unit testing

Coming Up

- Algorithms
  - Algorithm analysis
  - Insertion sort
  - Binary search