## 6.100L Recitation 7 - 28 October, 2022

**Reminders:**

- MQ7 next Monday 10/31
- PS3 due next Wednesday 11/2

**Lecture 13: Exceptions & Assertions**

Exception Handling

Exceptions occur when the syntax is correct, but the code performs some operation that isn't allowed. We can handle them in multiple ways. Below are a few options:

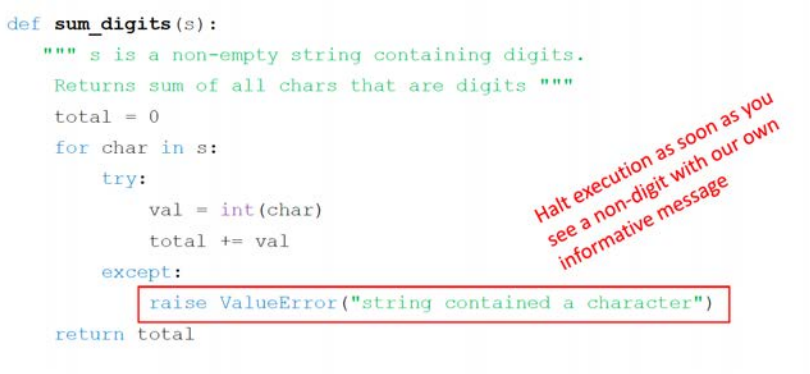
1. **`try/except:`**
    - Use this to handle an exception (i.e. prevent the program crashing)
    - If you don't specify a specific exception, then it handles ALL exceptions that occur in the try block.
    - If you specify a specific exception, the except clause handles only errors of that type.
    - This is optional, but can include a message after the error is thrown:
      `except ZeroDivisionError("Cannot divide by zero")`

```
try:
    a = int(input("Tell me one number: "))
    b = int(input("Tell me another number: "))
    print("a/b = ", a/b)
    print("a+b = ", a+b)
except ValueError:
    print("Could not convert to a number.")
except ZeroDivisionError:
    print("Can't divide by zero")
    print("a/b = infinity")
    print("a+b =", a+b)
except:
    print("Something went very wrong.")
```

only execute if these errors come up

for all other errors

2. **`raise:`**
    - used when you want an exception to occur
    - e.g: `raise ValueError("string contained a character")`




3. **`assert:`**
    - good defensive programming technique, execution halted when expected condition not met.
    - `assert <Boolean condition>`
    - `assert <Boolean condition>, <assertion message>`

## Lecture 14: Dictionaries

- Example dictionary: my_dict = `{'key1': 'value1', 'key2': 2}`
- A dictionary is another data structure that maps keys to values
- Keys:
    - must be immutable
    - must be unique
    - Ordering is not guaranteed
    - `my_dict.keys() # returns the all keys of a dictionary`
- Values:
    - Don't need to be immutable or unique
    - `my_dict['key1'] # returns 'value1'`
    - `my_dict['key2'] # returns 2`
    - `my_dict.values() # returns the all values of a dictionary`

- Iterating over a dictionary iterates over the keys
- Using the keyword `in` tests for membership amongst keys
- always check `in dict`, not `in dict.keys()` for efficiency reasons
- `dict.items() # returns the key, value pairs of a dictionary`

# 6.100L Introduction to CS and Programming Using Python

Fall 2022