CS 130R: Programming in Python

#12: Pickle and Exceptions
Reading: Chapter 7
I/O Binary files - Pickles

- Storing complex data in files
- Use pickle function to preserve the data (object, list, dictionary, tuple) and save it *in its entirety* to a file. The object can then be restored, or unpickled, later on.

```python
# import the pickle module
import pickle

# lets create a list to be pickled
picklelist = ['one', 2, 'three', 'four', 5, 'can you count?']

# now create a file
file = open('filename', 'w')

# now let's pickle picklelist
pickle.dump(picklelist, file)  # object to pickle, and file

# close the file, and your pickling is complete
file.close()
```
I/O Binary files - Pickles

- Reading complex data in files
- Use pickle.load() function to unpickle the data.

```python
# import the pickle module
import pickle

# now open a file for reading
unpicklefile = open('filename', 'r')

# now load the list that we pickled into a new object
unpickledlist = pickle.load(unpicklefile)

# close the file, just for safety
unpicklefile.close()

# Try out using the list
for item in unpickledlist:
    print(item)
```
I/O Binary files - Pickles

```python
import pickle
data = (1.4, 42)
f = open('data.pkl', 'wb')
pickle.dump(data, f)
f.close()
```

Read data.pkl in text format

ÄG?^fffffffK*Üq.

```python
import pickle
f = open('data.pkl', 'rb')
data = pickle.load(f)
print(data)
f.close()
```
Handling Exceptions

• When Python runs into an error, it stops the current program and displays an error message. (It raises an exception)

```python
>>> number = int("Hello")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'Hello'
```

```python
>>> n = int(input("Please enter a number: "))
Please enter a number: 23.5
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: '23.5'
```
Handling Exceptions

• An exception is an error that happens during the execution of a program.
• Different types of errors (syntactical) result in different types of exceptions.
• Error handling is generally resolved by saving the state of execution at the moment the error occurred and interrupting the normal flow of the program to execute a special function or piece of code, which is known as the exception handler.
• Depending on the kind of error ("division by zero", "file open error" and so on) which had occurred, the error handler can "fix" the problem and the program can be continued afterwards with the previously saved data.
try-except statement

```
try:
    statements
except [(Exception1, Exception2,...)]:
    statement

# example of error
n = int(input("Please enter a number: "))
Please enter a number: 23.5
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: '23.5'
```
Specifying an exception type

# handling error by specifying the exception type
try:
    n = int(input("Please enter a number: "))
except ValueError:
    #print executes only if error occurs
    print(“That was not an integer!”)
Please enter a number: 23.5
That was not an integer!
Specifying an exception type

```python
while True:
    try:
        n = input("Please enter an integer: ")
        n = int(n)
        break
    except ValueError:
        print("No valid integer! Please try again ...")
print("Great, you successfully entered an integer!")
```
Selected exception types

- IOError – I/O files – when they do not exist
- IndexError – for indices out of limits
- KeyError – when a dictionary key is not found
- NameError – when a name (variable, function) is not found
- SyntaxError – for syntax errors
- ValueError – inappropriate value
- ZeroDivisionError
import sys

try:
    f = open('integers.txt')
    s = f.readline()
    i = int(s.strip())
except IOError:
    print("I/O error")

except ValueError:
    print("No valid integer in line.")

except:
    print("Unexpected error:")

When to handle errors?
- Every time there is input from user,
- I/O to from files
Next lecture ...

- Importing modules