Lecture 13

1 Files

In every project:

- `python3 skater.py < in_file > out`
- `python3 moonlander.py < in_file > out`
- `python3 solver.py < in_file > out`

Here, the reading from and writing to files was handled by the terminal itself, not Python. But what if I want to open a file and read from it from inside Python? We can!

1.1 Reading Files

```
my_file = open("text_file.txt", 'r')  # 'r' is for reading
print(my_file.readline())
print(my_file.readline())
print(my_file.readline())
print(my_file.readline())
```

Draw picture with cursor, then modify to for loop!

```
my_file = open("text_file.txt", 'r')

for line in my_file:
    print(line)  # extra newlines
    # print(line.strip())
    # print(line.rstrip())
```

1.2 Writing to Files

What about writing to a file? We can do that too!

```
# must exist or causes error
in_file = open("text_file.txt", 'r')

# created if doesn't exist, contents erased if it does
out_file = open("new_file.txt", 'w')

line_num = 1
for line in in_file:
    print(line_num, line.rstrip())
    out_file.write("{:d}) {:s}".format(line_num, line))
    line_num += 1

in_file.close()
out_file.close()
```

2 Sorting

2.1 Sorting Numbers

```
# Suppose that I have a list of numbers
nums = [4, 6, -1, 8, 19]

# How might I sort it?  We'll get into more details of exactly "how"
# a little later, for now let's focus on how to get Python to do it
# for us

# sorted(list) will make a new list that contains the same values as
# the old one, but sorted
print(sorted(nums))
# and the original list is left unchanged
print(nums)

# Whereas, list.sort() returns None and sorts the list in place
nums.sort()
print(nums)  # so nums is now sorted

# We can sort in reverse (i.e., decscending order)!
nums.sort(reverse=True)
print(nums)
```

That's great! We can use this to sort a list of anything... almost. What if I want to sort a list of objects? How do I say what to sort based on? What if, for example, I wanted to sort a list of Pets.

2.2 Sorting Pets

```
from pet import Pet
# This is new.  We need it for attrgetter seen below.
from operator import attrgetter

pets = []
pets.append(Pet("Oliver", "Iguana", 12))
pets.append(Pet("Charlie", "Mongoose", 7))
pets.append(Pet("Spot", "Dog", 2))
pets.append(Pet("Casper", "Cat", 5))

print("\n-- Unsorted Pets --\n")
for pet in pets:
   print("{:7s} is {:2d} years old.".format(pet.name, pet.age))

# To sort by the attribute name
pets.sort(key=attrgetter('name'))

print("\n-- Sorted by name --\n")
for pet in pets:
   print("{:7s} is {:2d} years old.".format(pet.name, pet.age))
```

```
# Another way of sorting.  This uses a cool new thing called a lambda
# function!
pets.sort(key=lambda pet: pet.age)

print("\n-- Sorted by age --\n")
for pet in pets:
    print("{:7s} is {:2d} years old.".format(pet.name, pet.age))
```

3 Exceptions (Time Permitting)

What happens when I do the following?

```
value = int(input("Enter an integer: "))

print("You entered: {:d}.".format(value))
```

To ask a slightly different question, what could go wrong? What if I type in "five"? What happens then?

It crashes! More specifically, it raises an exception! More specifically, it raises a `ValueError` (this will be important in a moment). But what if I don't want the program to crash just because of a slight error? I can tell python to 'try' to do something and do something else of an exception is raised.

```
try:
    value = int(input("Enter an integer: "))

    # We only get here if there was not an error.
    print("You entered: {:d}.".format(value))
except ValueError:
    print("ERROR: That wasn't an integer.\n")
```

Let's modify the code so that I keep asking for a value until it's an integer.

```
is_valid_input = False

while not is_valid_input:
    try:
        value = int(input("Enter an integer: "))
        is_valid_input = True
    except ValueError:
        print("ERROR: Please enter an integer.\n")

print("You entered: {:d}.".format(value))
```

Here I'm saying to 'try' to do a possibly unsafe operation, and if a `ValueError` occurs, instead do something else.

Project 5

Intro Project 5