

Midterm 1

Preparation 2

Lists, Strings, and Files

Strings

- Strings are immutable
 - Strings cannot be changed, but we can build new strings from old ones
 - Because building strings as strings is expensive, we should get used to building strings as lists, and then convert in one step to strings

String Modification Pattern

- A function that takes a string and then returns the same string with all instances of 'a' preceded and followed by an underscore
 - Example:
 - `'Amazonas Territory'` becomes
 - `'_A_m_a_zon_a_s Territory'`

String Modification Pattern

- Solution:

```
def undera(string):  
    result = []  
    for letter in string:  
        if letter == 'a':  
            result.append('_a_')  
        elif letter == 'A':  
            result.append('_A_')  
        else:  
            result.append(letter)  
    return "".join(result)
```

Define an empty list to contain the components of the return value

Go through the argument string character by character

Build the result component by component

Use the `"".join()` in order to return a string instead of a list

String Processing

- We can make use of the numerous methods for programming tasks involving strings and list
 - The keyword “in” can be used to test membership
 - Test whether an “at” character is in a string:

```
def test_email(string):  
    return '@' in string
```

- By the way, this is not a very exhaustive test for a string being an email address

String Processing

- Testing whether a string is an IPv4 (Internet Protocol version 4) address
 - IPv4 addresses consists of four fields separated by periods
 - Each field has to be a number between 0 and 256.

String Processing

- Testing whether a string is an IPv4 address
 - First we break the string apart around periods to obtain the four fields
 - If we do not get four fields, it is not an IPv4 address
 - We then test whether the four fields are integers. If not, its not an IPv4 address
 - We then test whether the numbers are smaller than 0 or larger than 256. If it is, then it is not an IPv4 address
 - Now the string has passed all tests and we can certify that it is an IPv4 address

String Processing

- We develop the program step by step

```
def test_ipv4(string):  
    # Here we test for all reasons that  
    # the string is not an IP address  
    return True
```

- We check for reasons to return False
- If we cannot find a reason, we return True

String Processing

- We develop the program step by step

```
def test_ipv4(string):  
    fields = string.split('.')  
    if len(fields) != 4:  
        return False  
    # more tests  
    return True
```

- We use strip with '.' as the separator
- We then check for the number of fields

String Processing

- We develop the program step by step

```
def test_ipv4(string):  
    fields = string.split('.')  
    if len(fields) != 4:  
        return False  
    for field in fields:  
        if not field.isdigit():  
            return False  
    return True
```

- We tests whether all fields consist of digits

String Processing

- We develop the program step by step

```
def test_ipv4(string):  
    fields = string.split('.')  
    if len(fields) != 4:  
        return False  
    for field in fields:  
        if not field.isdigit():  
            return False  
        else:  
            number = int(field)  
            if number < 0 or number > 256:  
                return False  
    return True
```

- If all fields are digits, we convert the field to an integer and check whether it is between 0 and 256.

Files

- To access files, we need to open and close the files.
- The standard way is the with-construct that automatically closes the file
- We open the file in read (default) or write mode
- There is also a distinction between text (default) and binary.
- To open a file for reading, we just use
 - `with open("/Users/thomasschwarz/Documents/test.txt") as infi:`

Files

- Here we process a text file that contains potential IPv4 addresses

```
def process_file(filename):  
    with open(filename) as infi:  
        for line in infi:  
            string = line.strip()  
            print("{:20s} {:6s}".format(  
                string, str(test_ipv4(string))))
```

- It is important to strip the line.

```
1,9,0,0  
1.0.0.0  
234.23.a67.5.23  
25.31.109.23  
356.21.2.5  
0.3.a5.6.7
```