# LECTURE 25: REVIEW

## POPULAR APPLICATIONS OF SCRIPTING LANGUAGES

- System administration
- Web development: server side, client side
  - Many web frameworks
- Text processing / manipulation
- Gluing programs and tools
- Job control and automation
- Graphical User Interfaces

## TOPIC 1: PYTHON PROGRAMMING

## PYTHON BASICS

- Data types
- Variables
- Operators
- Print statement
- Functions
- Modules and packages
- Strings, indexing and slicing

## RECURSIVE FUNCTIONS

- A function that calls itself is called a recursive function
- Any recursive function can be divided into two parts:
    - Base case(s): Where we handle the most basic case
    - Recursive case(s): Where we reduce the problem to a simpler problem of the same form
- Advantages?
    - Intuitive, elegant, appears in job interviews
- Disadvantages?
    - Inefficient as the problem size grows

## MAPPING, FILTERING, REDUCTION

- Mapping: One-to-one transformation through a function
  - [1,2,3,4,5] -> [1,4,9,16,25] (lambda x: x**2)
- Filtering:  Apply a condition, retain ones that satisfy the condition
  - [1,2,3,4,5] -> [2,4] (lambda x: x % 2 == 0)
- Reduction: Apply a binary function to each member of a list in a row
  - [1,2,3,4,5] -> 15 (lambda x,y: x + y)
  - (((1+2)+3)+4)+5=15
- Reserved keywords: filter, map, reduce

## MAPPING, FILTERING AND REDUCTION

```
>>> lst = [1,2,3,4,5]
>>> list(map(lambda x: x ** 2,lst))
[1, 4, 9, 16, 25]
>>> list(filter(lambda x:  x % 2 == 0,lst))
[2, 4]
>>> reduce(lambda x,y: x + y,lst)
15
>>> map(lambda x: 1 if x % 2 == 0 else 0,lst)
[0, 1, 0, 1, 0]
```

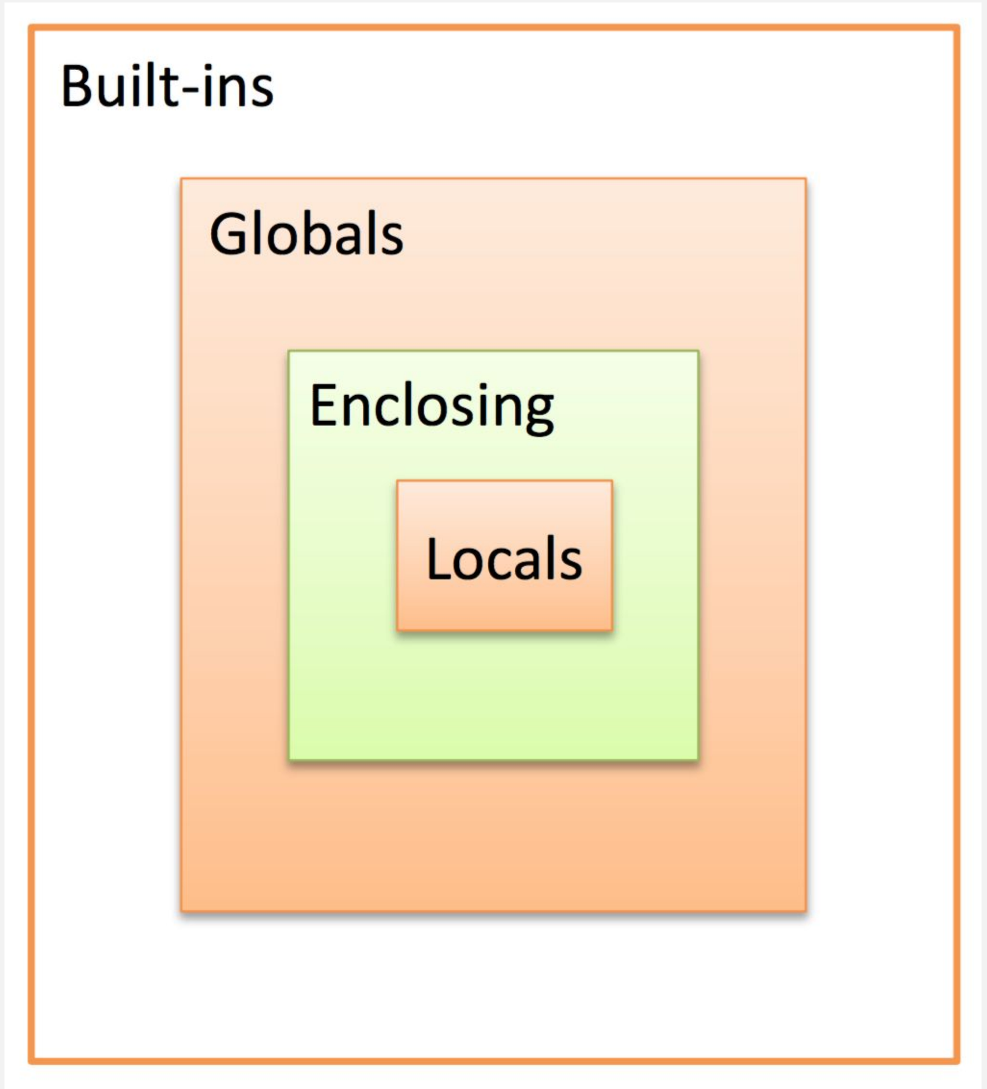
## PYTHON CLASSES

```
class BankAccount(object):
    def __init__(self):
        self.balance = 0

    def deposit(self,amount):
        self.balance += amount

    def withdraw(self,amount):
        self.balance -= amount

    def get_balance(self):
        return self.balance
```

## LEGB RULE

- Enclosing a lambda function

```
def a(x):
    f = lambda x: x + 3
    print(f(3))
    print(x)
>>> a(4)
6
4
```



## TOPIC 2: PERL PROGRAMMING

## WHAT IS PERL

- A high-level language–very high level
- A glue language–excellent for uniting different systems
- A scripting language–commands are usually placed in a text file, then interpreted by the perl interpreter
- Optimized for text-processing , I/O, and system tasks
- Incorporates syntax parts from bsh, csh, awk, sed, grep and C
- Open-source and free language-supported by a helpful international community

## PERL DATA TYPES

- Three main data types:
  - Scalars (Single valued variables)
  - Arrays (List of ordered scalars, indexed by number)
  - Hashes (Unordered scalars, indexed by strings)
- All variables are global unless otherwise declared
  - Declared in a local scope with a my qualifier (if not declared in any block, its scope is till the end of file)

## LIST OPERATORS AND FUNCTIONS

- Many useful operations or functions are built-in
  - sort
  - reverse
  - push/pop
  - unshift/shift
  - split/join
  - grep
  - map

## FLOW CONTROL

- More options than Python
- Conditional statements: if, unless
- Loop statements
  - while
  - until
  - for
  - foreach
- Modifiers

## FILES

- Access to files is similar to shell redirection
- Standard files
- Reading from files
- Writing to files
- File checks

## TOPIC III: TEXT PROCESSING

## REGULAR EXPRESSION

- A regular expression is a sequence of characters that defines a search pattern or patterns
  - It uses special characters (often called meta characters)and other characters to match text
  - Meta characters: ^$.|{}□()*+?\
- Some regex operators and functions in Perl
  - m/PATTERN/: the match operator
  - s/PATTERN/REPLACEMENT/: the substitution
  - =~,!~: binding (also called comparison) operators
- It works with many languages/tools

## CHARACTER CLASS RANGE AND NEGATION

- [A-Z]: find an upper-case letter
- A-Z: find an A followed by a hyphen followed by a Z
- [0-9.,]: find a digit or a full stop or a comma
- [0-9a-fA-F]: find a hexadecimal digit
- [a-zA-Z0-9\-]: find an alphanumeric character or a hyphen
- [1-31]: find a 1 or 2 or a 3
- [^a]: find any character other than an a
- [^a-zA-Z0-9]: find a non-alphanumeric character

## TOPIC IV: SCRIPTING FOR SYSTEM ADMINISTRATION

## WHAT IS SHELL

- Shell is a program
  - It takes commands issued from the keyboard as input, relays them to the OS for execution, and displays the output from the execution if there is any
- What's its first task?
  - Prints a command-line prompt e.g., #, $, >
- There are many shell programs
- Common ones: sh, bash, csh,  tcsh, ksh

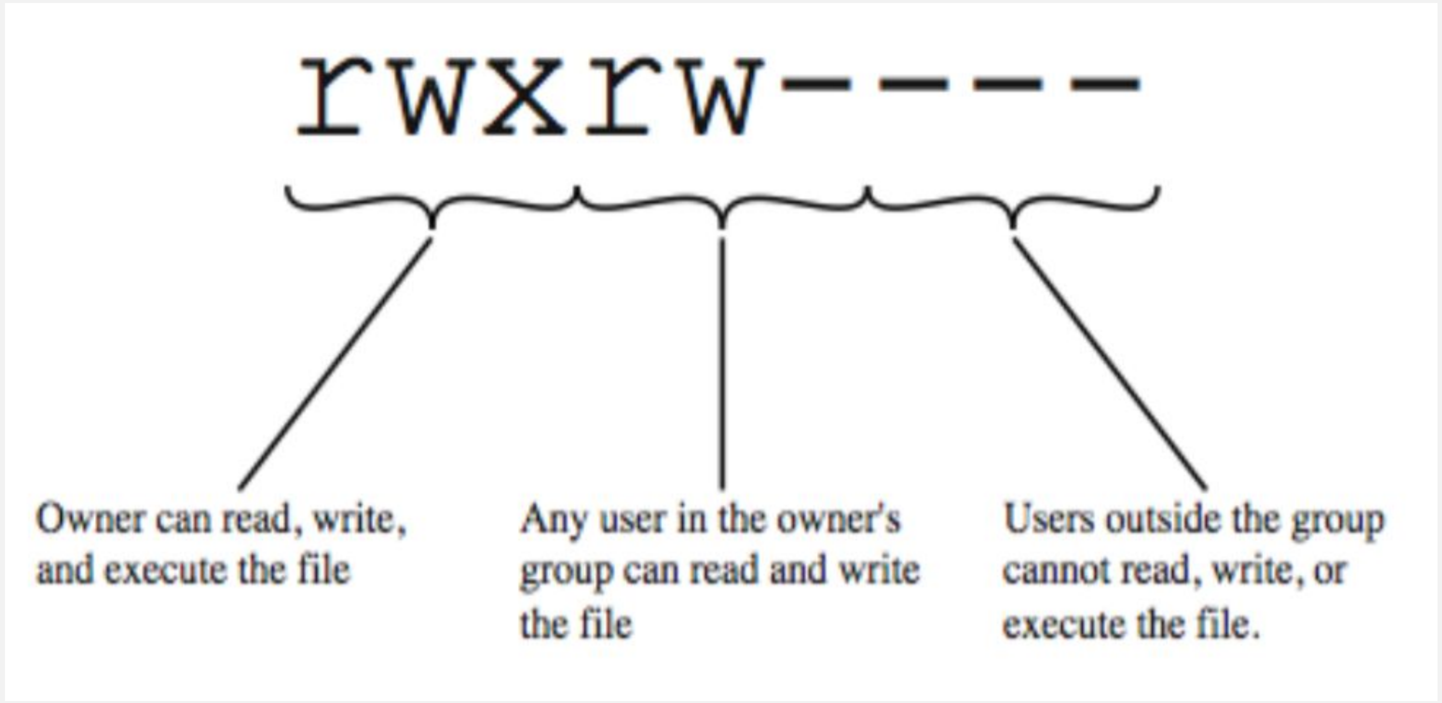
## WHAT DOES THESE COMMANDS DO?

- pwd: show current directory I am in
- ls: list content of my current directory
- less: see content of file: “datafile.txt”
- mkdir: create a directory called
- cd: go into a directory
- vi create/edit a file
- cp copy a file to another file

## SOME OTHER SHELL COMMANDS

- rm: remove file(s) (DANGEROUS with –R flag)
- man rm: display manual page
- scp: securely copy files
- mutt: read emails
- cal: calendar
- wc: word, character, line counts
- gzip: gnu zip
- gunzip: gnu unzip
- yppasswd to change password

## PERMISSION BITS


rwxrw----
Owner can read, write, and execute the file
Any user in the owner's group can read and write the file
Users outside the group cannot read, write, or execute the file.

- 3 sets of permission
  - owner, group owner, and others

## PIPES

- The shell allows one to use standard output of one process as the standard input to another process.
- General form command A | command B
- ps –ef | grep sshd
- ls | sort

## REDIRECTION

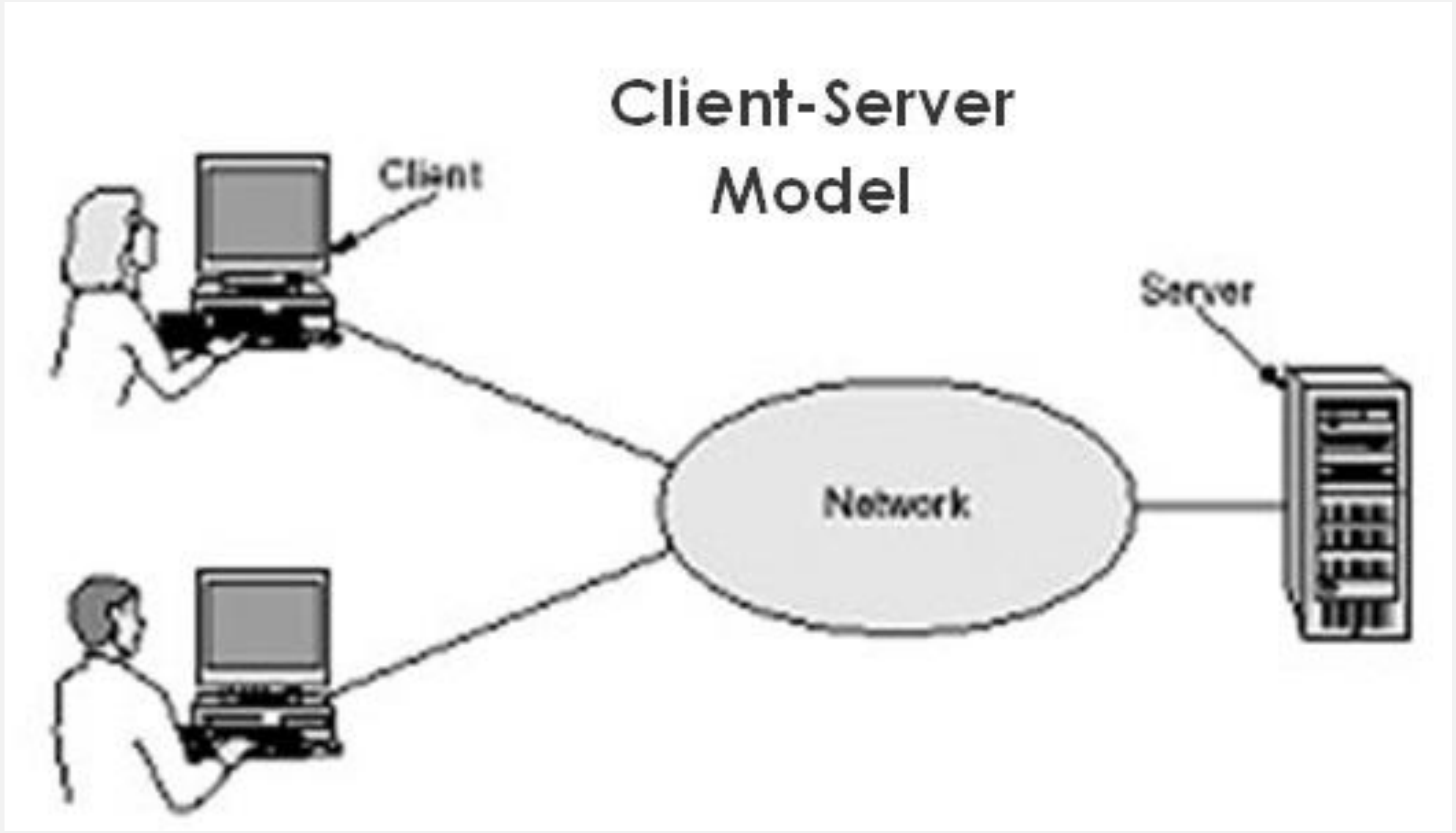
- Among the most useful facilities that the shell provides are the shell redirection operators
- Every process has at least 3 communication channels to use: STDIN, STDOUT, STDERR
  - Each can connect to a terminal window, a file, a network connection or another process’ channel
  - A process is any program in execution. The program can be the shell, an application, or a program that you write

## TOPIC V: WEB PROGRAMMING

## A CLIENT/SERVER APPLICATION

- Consists of several parts
  - Client program: the code that runs on the client machine; interacts with the user and the server
  - Server program: the code that runs on the server machine; accepts and serves client requests
  - Application protocol: types/order/format of the messages transferred between client and server programs
- Today’s client/server applications commonly run on the internet

## A CLIENT/SERVER APPLICATION



## FLASK

- Flask is a micro web framework
- Supports cookies
- Extensions available
- Lots of 3rd party libraries
- Install: pip(or pip3) install flask

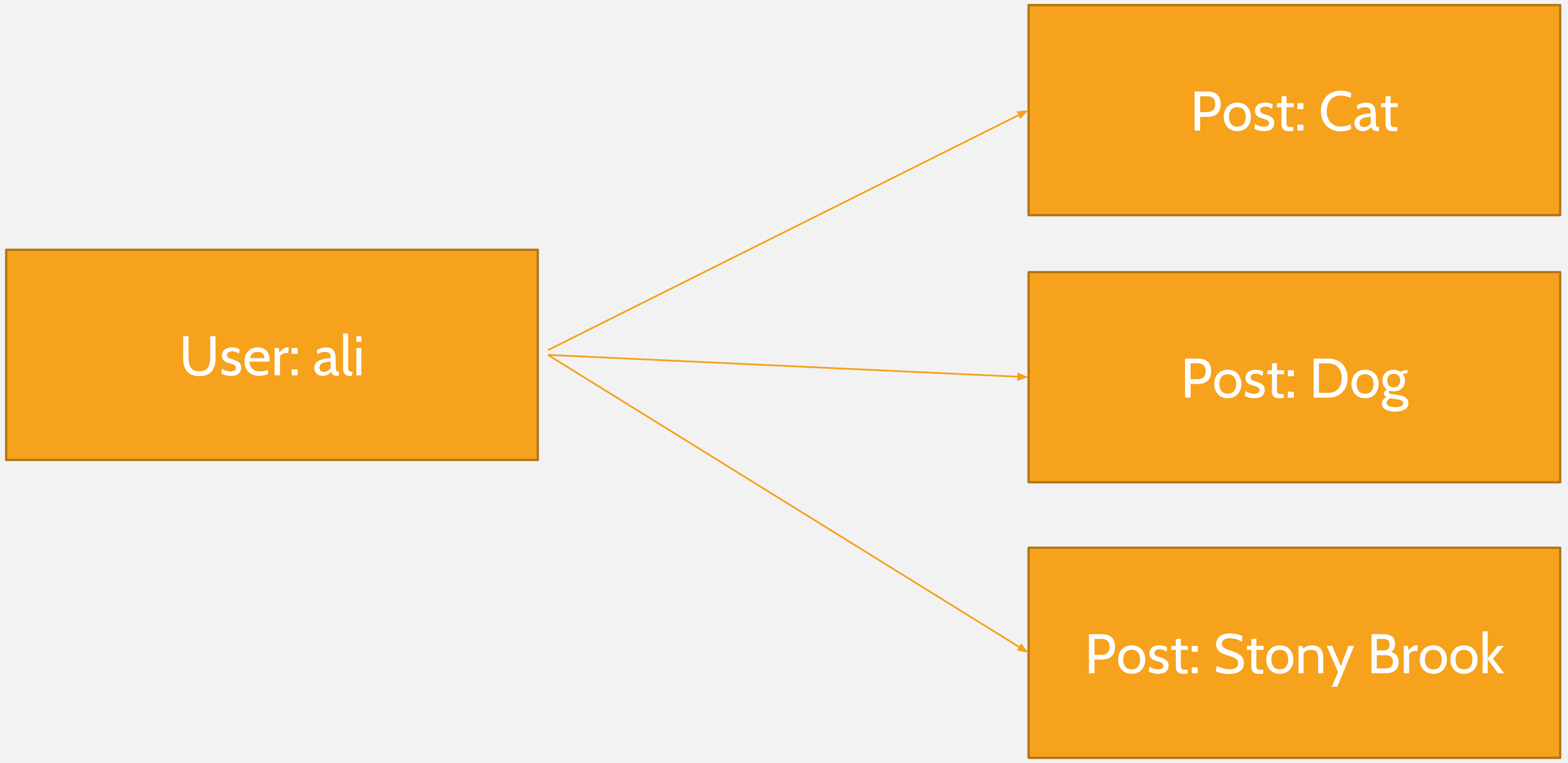
## WIKI APPLICATION

- What is a Wiki?
- Implementation
  - Displaying wiki pages
  - Submit pages
  - Index page listing wiki pages

## APP STRUCTURE

- app/: for application files
  - app/forms.py: Submission form
  - app/models.py: File for models(Data and data logic)
  - app/routes.py: Route decorators(To bind a function to a URL)
  - app/__init__.py: Initialize everything
  - app/templates: Template files
- config.py
- wiki.py: To start Flask application (Shell variables etc.)

## ONE TO MANY RELATIONSHIP

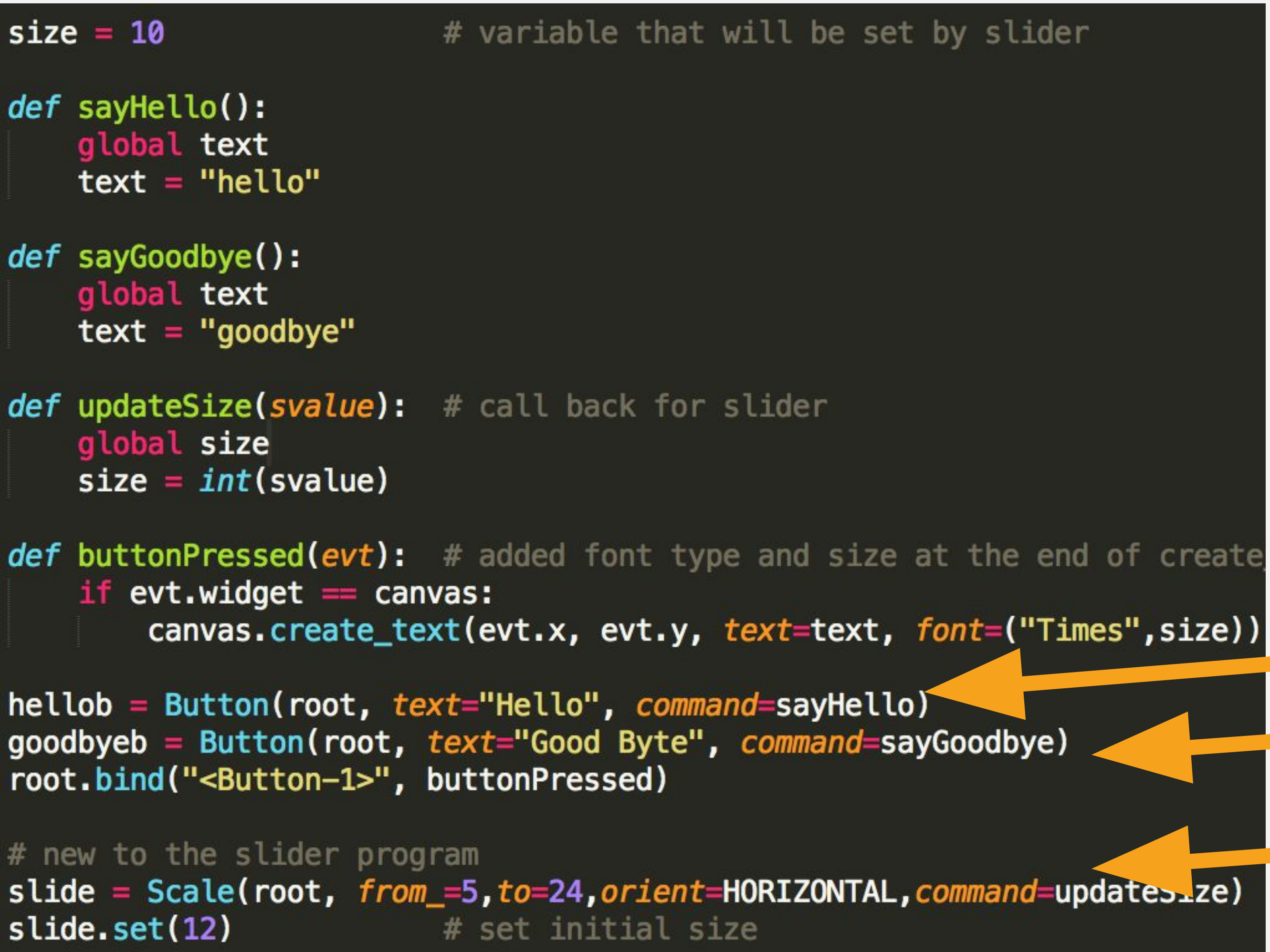


## TOPIC VI: GUI PROGRAMMING

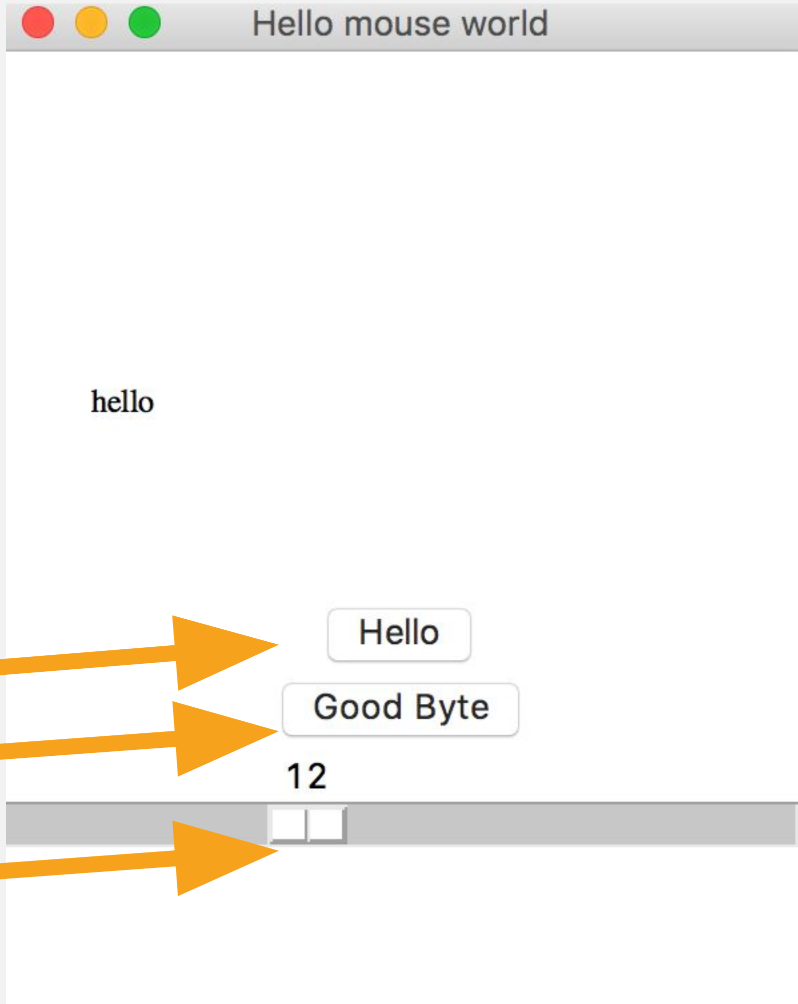
## GUI PROGRAMMING

- GUI: Graphical User Interface
- Type of interface that allows users to interact with computers using visual elements rather than text commands
  - Use windows, menus, buttons, text boxes, scrollbars
- Python uses a library called “Tkinter” to create GUI components
  - Alternatives? WxPython, PyQT, Kivy ….

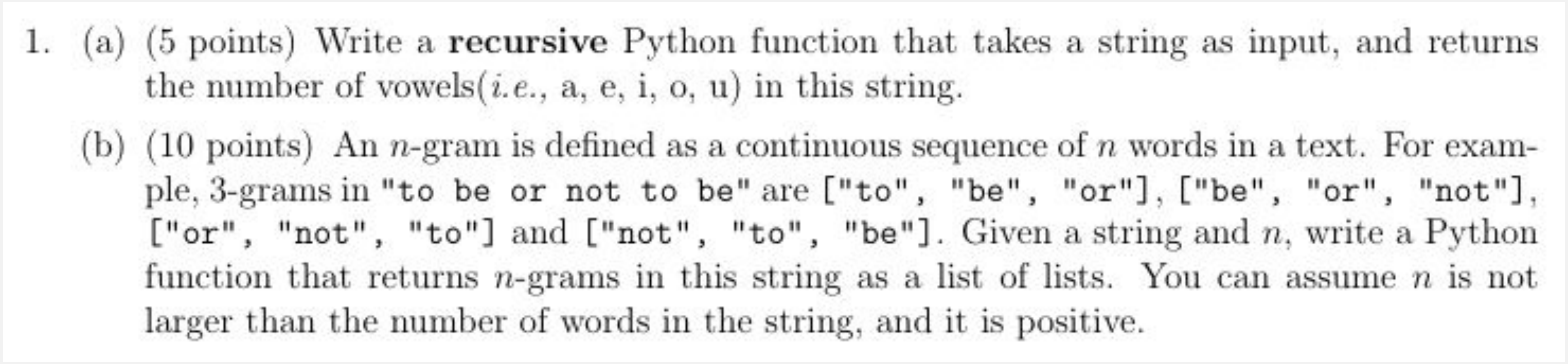
# SLIDER.PY



```
size = 10                  # variable that will be set by slider

def sayHello():
    global text
    text = "hello"

def sayGoodbye():
    global text
    text = "goodbye"

def updateSize(svalue):   # call back for slider
    global size
    size = int(svalue)

def buttonPressed(evt):   # added font type and size at the end of create
    if evt.widget == canvas:
        canvas.create_text(evt.x, evt.y, text=text, font=("Times",size))

hellob = Button(root, text="Hello", command=sayHello)
goodbyeb = Button(root, text="Good Byte", command=sayGoodbye)
root.bind("<Button-1>", buttonPressed)

# new to the slider program
slide = Scale(root, from_=5,to=24,orient=HORIZONTAL,command=updateSize)
slide.set(12)              # set initial size
```



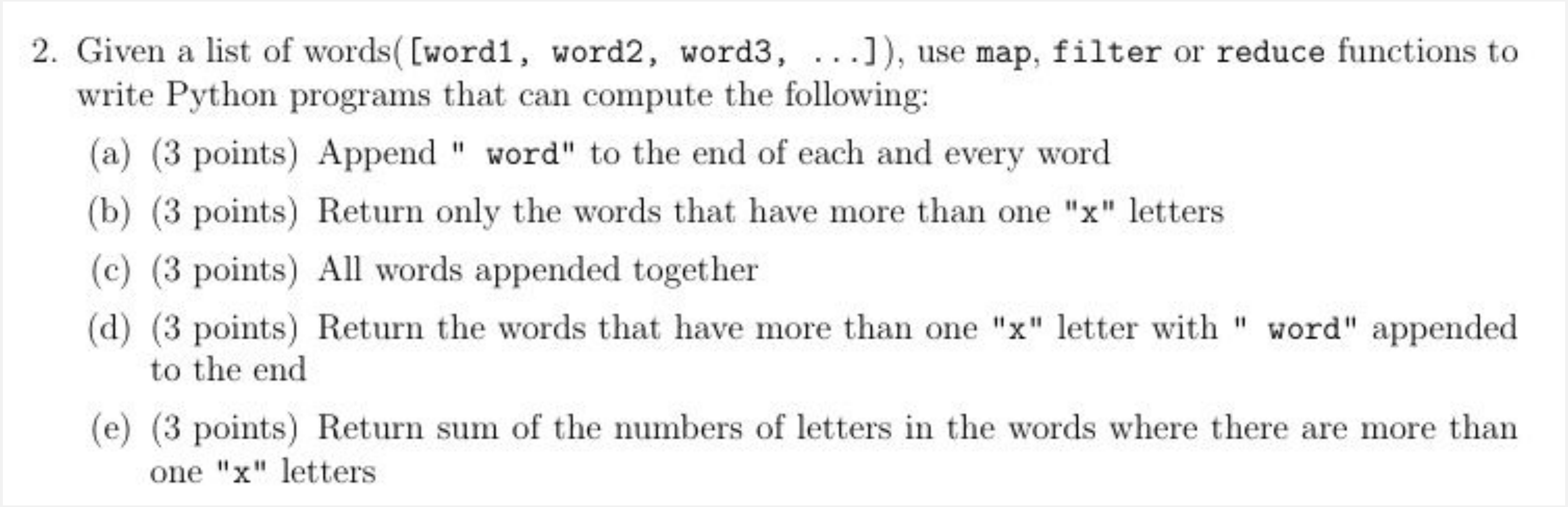
# WHAT TO EXPECT IN FINAL?



1. (a) (5 points) Write a **recursive** Python function that takes a string as input, and returns the number of vowels(*i.e.*, a, e, i, o, u) in this string.

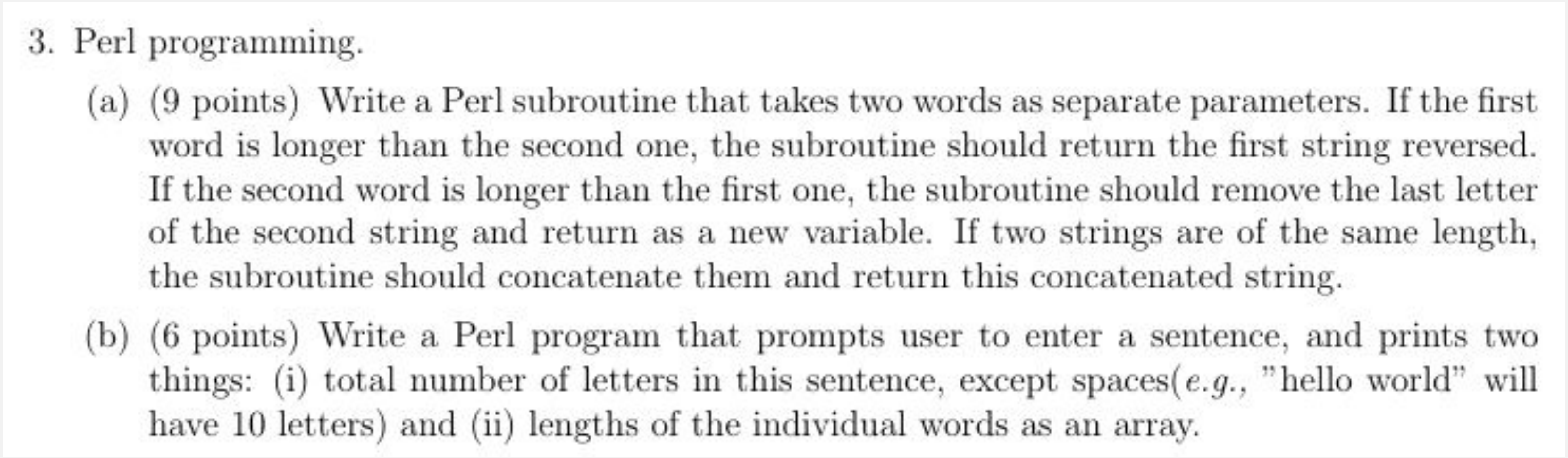
   (b) (10 points) An $n$-gram is defined as a continuous sequence of $n$ words in a text. For example, 3-grams in `"to be or not to be"` are `["to", "be", "or"]`, `["be", "or", "not"]`, `["or", "not", "to"]` and `["not", "to", "be"]`. Given a string and $n$, write a Python function that returns $n$-grams in this string as a list of lists. You can assume $n$ is not larger than the number of words in the string, and it is positive.

## WHAT TO EXPECT IN FINAL?



2. Given a list of words(`[word1, word2, word3, ...]`), use `map`, `filter` or `reduce` functions to write Python programs that can compute the following:
   (a) (3 points) Append `" word"` to the end of each and every word
   (b) (3 points) Return only the words that have more than one `"x"` letters
   (c) (3 points) All words appended together
   (d) (3 points) Return the words that have more than one `"x"` letter with `" word"` appended to the end
   (e) (3 points) Return sum of the numbers of letters in the words where there are more than one `"x"` letters

## WHAT TO EXPECT IN FINAL?



3. Perl programming.

   (a) (9 points) Write a Perl subroutine that takes two words as separate parameters. If the first word is longer than the second one, the subroutine should return the first string reversed. If the second word is longer than the first one, the subroutine should remove the last letter of the second string and return as a new variable. If two strings are of the same length, the subroutine should concatenate them and return this concatenated string.

   (b) (6 points) Write a Perl program that prompts user to enter a sentence, and prints two things: (i) total number of letters in this sentence, except spaces(*e.g.*, "hello world" will have 10 letters) and (ii) lengths of the individual words as an array.

## FIN!