How to think like a computer scientist (learning with Python). 

Reading assignment 1: Foreword, Preface, Chapter 1 (The way of the program).

Questions: How old is Python? What languages does it relate to? Give a very brief summary of what the Foreword and the Preface both are trying to say. Describe what you think is ahead of you in this book (be totally open and honest). What is a program (1.2)?

Reading Assignment 2: Chapter 2 (Variables, expressions and statements).

Questions: What’s the meaning of type(2.3)? What does print do? What’s the difference between 12 + 13 and ”12” + ”13”? How do you write a million in Python? What does print 1,000 do in Python? What’s the difference between 3/2 and 1.5?

Consider the following Python fragment:

\[
\begin{align*}
n & = 3 \\
m & = 5 \\
n & = n + m \\
m & = n - m \\
n & = n - m
\end{align*}
\]

What values do we have in the two variables at the end? Can you summarize the meaning of the last three lines, in general? What kind of statements are the five statements above?

Give a short account of what a legal variable name should look like. What is an operator? What is an operand? What do we mean by rules of precedence? What’s the difference between 2/3 * 9 and 9 * 2/3? What do + and * mean in a string context? How do you calculate 1.5 squared (it’s 2.25) and how do you calculate the square root of 2 in Python? Give a one line summary of section 2.9 in the text. Why would you use comments in your programs and how do you mark them (syntactically)?

Reading Assignment 3: Chapter 3 (Functions)

Questions: What does type conversion have to do with function calls? Is 1 the same as 1.0 in Python? If the variable a contains a 1 and the variable b contains a 2 write the shortest expression that evaluates to 0.5 using a, b but no literals or additional variables.

---

1 http://www.cs.indiana.edu/classes/a202-dger/fall2006/a201-text/thinkCSpy.pdf
2 Recall that most answers can be found in the text as you read it. I am putting this list of questions as I turn the pages of the book myself, and where I see a sentence that is particularly relevant, or when I see an example whose importance is I think significant, I immediately add a question about it here, in this material. So reading the text with this list of questions next to you is the approach I recommend.
The logarithm in base $10$ of a number $n$ is the $x$ in the equation $10^x = n$. Write a program that checks this equation for a number of your choosing.

Write Python expressions that verify the following identities:

a) $10^{\log_{10} x} = x$ for $x > 0$, and

b) $\sqrt{3^2 + 4^2} = 5$ (calculate both sides and print them, then look at them to compare).

What is a function in Python? Can you define your own functions in Python? What do we mean by list of parameters in the context of this chapter? Define a function called `bart` which prints the ASCII art below every time it’s called:

```
    |\\\\\|
    |      |
    | (o)(o)
    |    _)
    | ,___|
    /____\
```

Write another function, called `three`, which produces the ASCII art shown above three times, one below the other. How short can you make a program that prints one hundred Bart Simpsons?

Define flow of execution. What does it have to do with functions? Write a function called `next` that takes one argument (an integer) and prints the number that follows after it. Call this `next` with the following arguments: 1, 2, 3, 2+1, 3-1, -2, and report the results.

```python
def next(x):
    print(x + 1)
next(1)
next(2)
next(3)
next(2 + 1)
next(3 - 1)
next(-2)
```

What happens if you call `add` with two string arguments?

What is a stack diagram? Does it resemble the history of pages visited by a web browser? Draw the stack diagram of the following call:

```
add(add(3, 4), add(7, add(5, 6))).
```

Can you quickly evaluate the call? What is a Python traceback?

---

3 It is acceptable for `three` to rely on `bart` defined earlier.
4 Can you do it in less than 30 lines? Why or why not?
5 Hint: add one to the argument, then print the result.
6 Try to anticipate the results before you run the program.
7 In what way it does and in what ways it doesn’t?
8 You might be asked to draw stack diagrams often, later on in this review.
9 Trying to do it too quickly is sometimes a bit of a problem.
Consider the following Python program:

```python
def fun(x, y):
    return x * y  # [2]

a = fun(2, 3)  # [1]
b = fun("2", 3)

print a, b
```

What does it evaluate to? Replace the last statement `print a, b` with `print a + b` and explain the traceback. What’s wrong? Now eliminate the line marked [1] and change line [2] to read `return x + y`. Run the program and explain the traceback.

Consider the following definition:

```python
def fun(n, m):
    return m - n
```

Evaluate the following expressions (perhaps using stack diagrams):

a) `fun(fun(1, 2), 3)`
b) `fun(fun(1, 2), fun(3, fun(fun(4, fun(5, 6)), 7)))`
c) `fun(fun(1, 2), fun(3, fun(fun(4, fun(5, 6)), fun(7, 8))))`

What happens if in the definition of `fun` above we replace `return` by `print`\(^{10}\)?

Considering the following definitions:

```python
def alpha(x, y):
    return x + beta(y, x)
def beta(x, y):
    return y - x  # [1]
```

What does `alpha(2, 3)` evaluate\(^{11}\) to?

How does the answer change if the line marked [1] is changed to `return x - y`?

Consider the following definition:

```python
def fun(x):
    a = x + 1
    print a
    fun(a)
```

Using stack diagrams can you anticipate the result of calling `fun(-10)`?

---

\(^{10}\) Yes, you get the same as `add(add(3, 4), add(7, add(5, 6)))` in the page before. Why?

\(^{11}\) Use stack diagrams to calculate the answer, which is 1.
Reading Assignment 4: Chapter 4 (Conditionals and recursion).

Questions: Evaluate \((a \div b) \times b + a \mod b\) where \(a\) and \(b\) are integer variables. How can you determine if an integer number is odd/even? Write a program that determines the coins that a cashier will need to return an amount of money originally expressed in cents (e.g., a change of 123 cents will be returned as 4 quarters, 2 dimes, 0 nickels and 3 cents). What is \(3 \div 4\)? What is \(3 \mod 4\)? How do you calculate the last digit of any given number?

What is a boolean expression? Is \(2 + 3\) a boolean expression? What value does it have? Is \(2 > 3\) a boolean expression? What value does it have? If \(x\) is \(3\) and \(x \leq y\) is True what values can \(y\) have? How do you read the expression \(a \neq b\)? What does it mean? Does it have a value? If so, what is it? Is \(x = y\) a boolean expression?

Explain the following results:

```python
>>> x = 1
>>> y = 2
>>> x and y
2
>>> x or y
1
>>> 
```

Simplify the following expressions:

- \(a\) and \(!a\)
- \(a\) or \(!a\)
- \(x \geq 6\) and \(x < 7\)
- \(a == False\)
- \(a == True\)
- \(a\) or \(True\)
- \(a\) and \(True\)
- \(a\) and \(!a\)
- \(a\) or \(!a\)
- \(!False\) and \(!False\) or \(True\)
- \(!False\) and \(!True\) or \(False\)
- \(!False\) and \(!False\) or \(True\)
- \(!False\) and \(!True\) or \(False\)

Is \(!a\) and \(b\)\) the same as \(!a\) or \(!b\)? Justify your answer.

What are conditional statements and what can they do for us? Can you find the largest of two numbers without using an if statement? What if the two numbers are integers?

Describe the concepts of block and body. What is indentation and how is it related to block and body? Explain the use of the pass statement. Illustrate it with an example.

---

12 Do you need to know the values of \(a\) and \(b\)? Why or why not?
13 If not, what kind of expression is it?
14 You can use +, -, /, and abs(...) that is: addition, subtraction, division and absolute value.
What do we mean by alternative execution?

Consider the following function:

```python
def what(n):
    if n == 0:
        return 0
    else:
        return n + what(n-1)
```

What does it do? What’s the result returned by `what(3)`? What does `what(-3)` evaluate to? How can you change what so it works for negative arguments as well?

Define chained conditionals and write a program using a chained conditional to solve the following problem: Write a program (called Eight) that translates a number between 0 and 4 into the closest letter grade. For example, the number 2.8 (which might have been the average of several grades) would be converted to B-. Break ties in favor of the better grade; for example 2.85 should be a B.

What is a nested conditional?

Consider the following code:

```python
if x > 3:
    if x <= 5:
        y = 1
    elif x != 6:
        y = 2
    else:
        y = 3
else:
    y = 4
```

If `y` has the value 2 after executing the above program fragment, then what do you know about the initial value of `x`?

Consider the following two fragments:

```python
if x == 5:
    x = x + 1
else:
    x = 8
```

```python
if x == 5:
    x = x + 1
if x != 5:
    x = 8
```

Are the two fragments logically equivalent? Why or why not?

Simplify `a <= b and b <= c` in Python.

---

15 Explain what we mean by “branches in the flow of execution”.
16 A, B, C, D have the values 4, 3, 2, 1. A plus adds 0.3, a minus takes 0.3 away. F is 0. A+ is a 4.
Consider the following Python code:

```python
def what(n):
    if n == 0:
        result = 0
    else:
        print n                       # [1]
        result = n + what(n-1)        # [2]
    return result
```

What’s the result of calling `what(10)`? Now swap the statements marked [1] and [2]:

```python
def what(n):
    if n == 0:
        result = 0
    else:
        result = n + what(n-1)        # [2]
        print n                       # [1]
    return result
```

What’s the result of calling `what(10)` now?

What is recursion? Write a recursive function that plays “guess the number” with the user. Use `raw_input` for user input and keep playing until the user guesses the number. Can you keep track of and report the total number of guesses at the end? Can you program the game where you have a total of ten tries at most? Here’s a start:

```python
def game(____):
    ____ = raw_input("Enter guess: ")
    if ____ == ____:
        print "You guessed it. "
    else:
        if ____ < ____:
            print "Try higher"
        else:
            print "Try lower. "
        ____(____)
```

Did you use stack diagrams to answer the two questions about `what` on this page?

Define a function called `sum` with no arguments that asks the user (using `raw_input`) for two integers and prints their sum.

**Reading Assignment 5: Chapter 5 (Fruitful functions).**

**Questions:** What does `abs` do? Write a function called `distance` which receives four arguments `(x0, y0, x1, y1)` representing the coordinates of two points in the plane and calculates and returns the distance\(^\text{17}\) between them. Define and use the function

\(^{17}\) Which is the square root of the sum of the squared differences between the corresponding coordinates.
compare described on page 48. Write a function that calculates the area of a triangle using Heron’s formula\(^\text{18}\):

In geometry, Heron’s formula (also called Hero’s formula) states that the area of a triangle whose sides have lengths \(a\), \(b\) and \(c\) is

\[
\text{area} = \sqrt{s(s-a)(s-b)(s-c)}
\]

where \(s\) is the triangle’s semiperimeter:

\[
s = \frac{a + b + c}{2}
\]

(see also square root). Heron’s formula can also be written

\[
\text{area} = \frac{\sqrt{(a + b + c)(a + b - c)(b + c - a)(c + a - b)}}{4}.
\]

What is composition? Use composition to implement a function that calculates the area of a triangle using Heron’s formula but starting from the coordinates of the three vertices.

Write a function that receives three arguments representing the lengths of its sides and determines if the triangle is isosceles\(^\text{19}\) or not. Write a function called `inside` that receives information about a circle (two coordinates for the center and the value of the radius) and a point (two coordinates) and returns a boolean value indicating if the point is inside the circle or not.

Write a function that determines if two circles\(^\text{20}\) overlap. In this last case you have the freedom and responsibility of deciding what (and how many) arguments the function has.

**Reading Assignment 6: Chapter 6 (Iteration).**

**Questions:** If \(i\) is an integer variable, how do you increment\(^\text{21}\) it?

Take a look and review the material on page 56 then find/devise a non-recursive way to produce the first few Fibonacci numbers.

What is the output of the following code fragment:

```
  n = 10
  m = n
  n = 12
  print m
```

Does the change in \(n\) affect \(m\) at all? Why (or why not)?

\(^{18}\) http://en.wikipedia.org/wiki/Heron%27s_formula

\(^{19}\) Same question for equilateral, right, scalene.

\(^{20}\) Same question about rectangles.

\(^{21}\) What does that mean?
Briefly explain\(^{22}\) and evaluate (assume \(i\) holds an integer value):

- \(i = i + 1\)
- \(i == i + 1\)
- \(i == i\)
- \(i = i\)

What does the following code fragment print\(^{23}\)?

```python
print 1,
print 2, 3
print 4, 5, 6
print 7,
print "ha!"
```

What does it mean for equality to be commutative? How is assignment different from equality\(^{24}\)? Why is assignment non-commutative?

What is iteration? Is iteration the same as repetition? Python provides a `while` statement; what is its purpose? Write a function that takes one argument (call it \(n\)) and prints the integers between 1 and \(n\) in ascending order; write the function recursively, then using a `while` loop. Repeat this exercise but make the function print the numbers in descending order this time.

Consider the following Python code:

```python
i = 10
while i < 10:
    i = i - 1
print i
```

What does it print (or amount to)? Do you see any potential\(^{25}\) problem with it?

Now consider the following code:

```python
s = 0
i = 10
while i > 0:
    s = s + i
    i = i - 1
print s
```

What does it print (if anything)? What do you think it is computing?

\(^{22}\) Discuss differences, similarities of purpose and evaluation/processing.
\(^{23}\) Try to predict with great accuracy the output, spaces and everything.
\(^{24}\) Isn’t \(a = b\) essentially an assignment but also a statement of equality?
\(^{25}\) What if \(i\) starts at 9 instead of 10?
How do you find if a number divides another? What is an odd number? Write a function that takes one argument (a positive integer) and reports if the argument is odd or not. What is a prime number? Write a function that takes one argument (a positive integer) and reports if the argument is prime or not.

Can you calculate the sum of the first \( n \) numbers without using a loop or recursion?

Is the loop in the code below infinite? How do you know (for sure) before you run it?

```python
m = 3
n = 5
while n < 10:
    m = n - 1
    n = 2 * n - m
print n, m
```

Write a program that prints a table on two columns: on the left the integer temperatures between 0 and 100 (Fahrenheit) and in the right column the corresponding Celsius values. Make a similar table that helps converting miles into kilometers and pounds in kilograms.

How do you print a \texttt{tab} character? What is an escape character? How do you print a newline? How do you print three new lines using only one print statement? What do we mean by \texttt{cursor}? What base is the log function in Python using? How can you calculate the logarithm in base 2 of a number?

Write a one line program in Python that produces this output:

```
+--+
 | |
+--+
```

Solve the exercise at the end of section 6.3.

Write functions to print scalable letters \texttt{Z}, \texttt{M}, \texttt{E}, \texttt{L}, \texttt{C}, \texttt{F}, \texttt{W} and the digit \texttt{4} as square patterns of size \( n \) (where \( n \) is the only argument of the function, the size).

Here’s an eight by eight \texttt{Z}: remember the function only gets the size of the pattern:

```
*  *  *  *  *  *  *  *
  *  *  *  *  *  *  *  *
    *  *  *  *  *  *  *  *
      *  *  *  *  *  *  *  *
        *  *  *  *  *  *  *  *
```

---

26 What are tabs useful for?
27 Can you give an example?
Also write programs that generate each of these three patterns as scalable patterns:

```
* * * * * * * * * * * * * *
* * * * * * * * * * * * *
* * * * * * * * * * *
* * * * * * * * *
* * * * * * *
* * * * *
* * * *
* * *
* *
```

What do we mean by a loop variable? Give examples. How many loop variables are you using in your scalable patterns programs? Are you using encapsulation in your scalable patterns programs? Is it easier or harder to use encapsulation (also, is it better or worse)? Is it always easier/better/worse/harder to use encapsulation? Give an example of each.

Write a program that asks the user for two integer numbers and stores them in variables \( n \) and \( m \) and with \( n \leq m \) and prints all the prime numbers between \( n \) and \( m \). Are you using encapsulation and generalization in your coding?  

Write a program that prints a square table of random numbers (where the size provided).

What does the term development plan mean on page 66 in the text?

What is a local variable? Is a local variable like a parameter? In what ways is a local variable not like a parameter? Why do we need local variables at all in Python? What local variables have you seen in the code written so far in this material?

Write stack diagrams to describe the action of the following code:

```python
def odd(num):
    if num == 0:
        return False
    else:
        below = num - 1
        return even(below)

def even(num):
    if num == 0:
        return True
    else:
        below = num - 1
        return odd(below)

print odd(4)
print even(4)
```

---

29 The circumstances are very suitable, in this case.

30 Can you use a variable before it’s initialized? How do parameters get initialized?
```python
print odd(5)
print even(5)
```

How could you fix the code so it works for negative inputs as well? Is below an example of a local variable in the sense discussed on page 67 in the text (same name, different functions)? Is there any interference? Why or why not? Is num a similar example? Is it also a local variable? What is it? What’s the difference, if any?

Write a program that calculates for every number between \( n \) and \( m \) (given) the number of times it takes the sequence function on page 61 to reach 1 (one). Change the program so it also prints a histogram, like this:

```python
>>> histogram(5, 14)
14 ****************** (17)
13 ********** (9)
12 ********** (9)
11 ****************** (14)
10 ***** (6)
 9 ****************** (19)
 8 *** (3)
 7 **************** (16)
 6 ***** (8)
>>> 
```

Solve the exercise on page 69 and explain what you did and how.

Summarize the benefits of functions and give an example for benefit

**Reading Assignment 7: Chapter 7 (Strings).**

**Questions:** Would you call integer values atomic?

Would you say that these values are in some sense indivisible (that you can’t take them apart)? How else could a value be? What does the book mean by compound type? What benefit is associated with such a type? What is the difference between the integer 123 and the string "123"? Don’t we hit the keyboard three times to type 123 (the number) just like we do for the string? What’s raw_input’s return value? Have you ever seen a number? Have you ever seen a string of characters? How do you convert a string to an integer? How do you obtain the last digit in a number? How do you get the last character in a string? How do you obtain the first character in a string? How do you obtain the first digit in a number? Is there any such thing as an empty string? Is it a string? Can you convert it to a number? Can you use it in an if statement, or in a loop as the condition? Explain your answers.

If \( a = \"blueberry\" \) what does \( a[2] \), \( a[1] \), \( a[0] \) evaluate to?

Same questions for \( a[-1] \) and \( a[-2] \). Any surprises?

---

31 The book also gives a list of benefits. Feel free to discuss them if you decide to rely on their list.

32 Comes from Greek atomos which means indivisible.

33 How do you calculate the length of a string?
What is an index? What is a[2-1]? What is a[len(a)-1]?

What is a[len(a)]? What does a[-len(a)] evaluate to?

Same questions for a[-len(a)-1] and a[-len(a)+1].

Define what we mean by traversal of a string. Write a function that receives a word (as a string, of course) and prints back the word without the vowels, like this:

```python
>>> steno("nectarine")
'ntcrn'
>>> steno("blueberry")
'blbrry'
```

Change then the code to surround the vowels with parens instead of not showing them.

```python
>>> steno("nectarine")
'n(e)ct(a)r(i)n(e)'
>>> steno("blueberry")
'b(l)u(e)(b)e(ry)'
```

Use a for ... in ... loop to produce the reverse of a given string.

Solve/finish the exercise listed at the bottom of page 74.

What is a string slice? If a = "blueberry" evaluate the following:

- a[2:3]
- a[2:]
- a[:3]
- a[:]
- a[-1:-3]
- a[:-1]
- a[:1]

Explain how the comparison operators ==, <, <=, >, >= and != work with strings.

What does the following code fragment produce and why:

```python
a = "readEval"
b = "read_eval"
c = "readeval"
print a < b, a < c, b < c
```

---

34 That is, obtain and print a[-1] + a[-2] + ... + a[0] where a is the given string.
35 Discuss if they work any different for numbers.
What does it mean for strings to be immutable?

Write code for a function (call it fun) that receives a non-empty word and replaces the first letter in it with an underscore, like this:

```python
>>> fun("blueberry")
'_lueberry'
>>> fun("glue")
'_lue'
>>> 
```

Is that how you implemented your steno function from the previous exercises?

Write the code for the modified find function mentioned in the exercise on page 76.

Write a function count which gets two arguments, a string and a character, and reports the number of times (perhaps zero) that the character appears in the string. Write another function contains which takes a string and a character and determines if the character appears in the string or not. Can you define contains in terms of count?

What is the benefit and purpose of the string module?

Briefly describe each of the following:

1. string.find(...)  
2. string.lowercase  
3. string.uppercase  
4. string.digits  
5. string.whitespace

What’s the difference between the in operator as presented in this section (7.10) and the one seen in the for ... in ... loop? Are there any similarities, too?

Where can you find the Python Library Reference?

Programs to write:
- write a program that plays Hangman with the user
- write a program that shuffles the letters in a word (word jumble)
- write a program to produce circular permutations of a given word

Reading Assignment 8: Chapter 8 (Lists).

Questions: A ____ is an ordered set of values, where each value is identified by an index.

---

36 List the URL, like so: http://docs.python.org/lib/  
37 Randomly. For example with input python it could produce poynth  
38 Give it the word apple and it produces: pplea, pleap, leapp, eappl, and apple.
Do strings match the definition you just read? How do they match it and how don’t they?

Both lists and strings are called ________.

Is a string the same as a list of characters?

Can we have a list of strings?

What is a nested list? Can we have nested strings?

What values can we have in a list? Do they all have to be the same type?[^39]

What does `range(...)` do?[^40]

How do you create the following lists:

- `[4, 5, 6]`
- `[-2, 1, 3]`
- `[-9, -8, -7, -6, -5]`
- `[-9, -10, -11, -12]`
- `[0, 1, 2]`

Is the empty string an empty sequence? Can lists be empty?

Write a function that receives a list and a value and counts the number of occurrences of the value in the given list. Provide two implementations: recursive and iterative.

If `a = [5, 4, 3, 2, 1, 0]` evaluate the following expressions:

- `a[0]`
- `a[-1]`
- `a[a[0]]`
- `a[a[-1]]`
- `a[a[a[a[2]+1]]]`

Can you change an element of a sequence? What if the sequence is a string?

Consider the following code: what is the value of `a` at the end?

```
a = [1, 2, 3]
a[2]=0
a[a[2]] = 5
a[1:2] = []
```

[^39]: Can you have an integer, a string, a list of integers and a list of strings in a list?

[^40]: Include all variants, with one, two and three arguments in your answer.
What is the difference between \( a[1:2] = [] \) and \( a[1] = [] \)? How would \( a \) change if we make this change in the last statement of the fragment presented?

What does \( a + b \) amounts to if \( a \) and \( b \) are lists?

Write a function that receives a list of integers and returns their sum\(^{41}\).

How do you determine the length of a list?

If \( a = [1, 2, [3, 4], 5] \) what is the length of \( a \)? How many elements does it contain? How many integers? What does \( \text{len}(a) \) return?

Same questions after we make \( a[1:2] = [[2, 3], 4] \).

What does \( a[\text{len}(a)] \) evaluate to\(^{42}\)?

Write a function that receives a list (that could contain nested lists) of integers and reports the total the number of integers at all levels\(^{43}\).

If \( a \) is \([1, 2, 3]\) is there a difference between \( \text{not} \ 1 \in a \) and \( 1 \not\in a \)?

Is every for loop expressible through an equivalent while loop? Why or why not, and what is your justification? Is the reverse true?

Write loops that produce the first 20 terms in the sequences:

- \( 1, -1, 1, -1, 1, -1... \)
- \( 1, 3, 5, 7, 9, 11, 13, ... \)
- \( 1, 0, 2, -1, 3, -4, 5, -6, ... \)

Is one type of loop usually more useful or convenient than the other?

Which one do you prefer? Why?

If \( a \) is \([1, 2, 3]\)

- what is the difference (if any) between \( a * 3 \) and \( [a, a, a] \)?
- is \( a * 3 \) equivalent to \( a + a + a \)?
- what is the meaning of \( a[1:1] = 9 \)?
- what’s the difference between \( a[1:2] = 4 \) and \( a[1:1] = 4 \)?

What’s \( a[1:1] \) if \( a \) is a string of at least two characters\(^{44}\)?

What’s the purpose of the del operator? Can you delete a slice?

---

\(^{41}\) What is list traversal?

\(^{42}\) Assume \( a \) is a list: how do you obtain its last element?

\(^{43}\) For example \([1, [2, 3, 5], 4]\) is a list of three elements with five integers inside.

\(^{44}\) What if the string is shorter?
Why is there so much space left on page 88?

What is a state diagram and where have we used it in this text?

What is the following Python interaction trying to communicate:

```python
>>> a = [1, 2, 3]
>>> b = [1, 2, 3]
>>> id(a)
10954592
>>> id(b)
12948944
>>> a[0] = 3
>>> a
[3, 2, 3]
>>> b
[1, 2, 3]
>>> id(a)
10954592
>>> id(b)
12948944
>>> a
[3, 2, 3]
>>> b
[1, 2, 3]
``` 

Same question for this one:

```python
>>> a = [1, 2, 3]
>>> b = [1, 2, 3]
>>> id(a)
12012840
>>> id(b)
12773376
>>> b = a
>>> id(b)
12012840
>>> a[0] = 3
>>> a
[3, 2, 3]
>>> b
[3, 2, 3]
``` 

Is there any difference between the two?

Now explain this:

```python
>>> a = 1
>>> b = 1
>>> id(a)
10901272
>>> id(b)
10901272
>>> b = 2
>>> id(b)
10901260
```
Do you find anything related in this last example:

```python
>>> a = [1, 2, 3]
>>> id(a)
10954592
>>> a[0] = 3
>>> id(a)
10954592
```  

Why is `id(a) == id(b)` while `id(x) != id(y)` in the example below:

```python
>>> a = 9
>>> b = 9
>>> id(a)
10901176
>>> id(b)
10901176
>>> id(a) == id(b)
True
>>> x = [1, 2, 3]
>>> y = [1, 2, 3]
>>> id(x) != id(y)
True
>>> id(x)
11428848
>>> id(y)
12943768
```  

What is aliasing? The book shows that aliasing sometimes does not happen for strings? Why do you think that is so? Comment on the code below:

```python
>>> a = "banana"
>>> b = "bana" + " na"
>>> id(a)
12929248
>>> id(b)
12930912
>>> c = "banana"
>>> id(c)
12929248
```  

What’s the simplest way to clone a list? What’s cloning?

Write a function that sorts an array of integers in ascending order.

Write a function that reverses an array of integers (in situ, or as a clone).
Write a function that receives two lists and creates a third, that contains all elements of the first followed by all elements of the second.

Using nested lists write a function that implements matrix addition.

Does the slice operator always produce a new list?

Write a function to generate a matrix of given size (rows and columns) full of random values (positive integers between -50 and 50). Work out a show function that receives a matrix and prints it nicely, with all the values properly aligned, like this:

```python
>>> show([[1, 2, 3], [101, 102, 19], [1, 12, 123]])
  1   2   3
101 102  19
  1  12 123
```

Note the function works even if the argument is not a matrix proper.

Write a program that reverses the order of words in a sentence.

Write a program that reverses the order of the letters in the words in a sentence, leaving the order of the words in the sentence unchanged.

Combine the two programs above into one that reverses the order of the words in a sentence and the order of the letters in each word:

```
Echo> these are the days of miracle and wonder
these are the days of miracle and wonder
rednow dna elcarim fo syad eht era eseht
esesht era eht syad fo elcarim dna rednow
wonder and miracle of days the are these
Echo>
```

Answer the question on page 94.

Write a program that reads two times in military format (0900, 1730) and prints the number of hours and minutes between the two times. Here is a sample run. User input is in color.

```
Please enter the first time: 0900
Please enter the second time: 1730
8 hours 30 minutes
```
Extra credit if you can deal with the case that the first time is later than the second time:

```
Please enter the first time: 1730
Please enter the second time: 0900
15 hours 30 minutes
```

Write a program that reads a string (or receives one, if you write a function) and swaps the first with the last letter: for example, pooh would become hoop.

**Reading Assignment 9: Chapter 9 (Tuples).**

**Questions:** How do tuples resemble strings? How do they resemble lists? How do they resemble neither? Is it hard to create a tuple with a single element? Is it even possible? Are tuples sequences like lists and strings or not? Is slicing working for them in the same way? What does + mean for tuples?

What is the difference between

```
a, b = b - a, a - b
```

and

```
a = b - a
b = a - b
```

How can you swap the values of two variables in Python\(^{46}\)?

Explain what the following code does:

```
def swap(x, y):
    x, y = y, x

a = 3
b = 5
swap(a, b)
print a, b
```

How do you generate random numbers in Python?

Simulate a dice in Python.

Write a program to play the game of craps.

Solve the exercises on page 98.

\(^{46}\) With and without using a third variable.
Explain this code:

```python
>>> a = [[[0] * 3] * 3
>>> a
[[[0, 0, 0], [0, 0, 0], [0, 0, 0]],
>>> a[0][1] = 4
>>> a
[[[0, 4, 0], [0, 4, 0], [0, 4, 0]],

Is there anything wrong with it?

Describe and implement a test for the uniform distribution of the Python pseudorandom number generator. Use 8 buckets. Explain every line of your code.

Solve the exercise mentioned at the bottom of page 101.

Write a function that calculates the average of a list of numbers.
Write a function that receives a size, and two numbers (low and high) and returns a list of random numbers in the given range.

Write another function that calculates the standard deviation in a list of numbers received as input: the standard deviation is the average of the squared deviations to the mean.

Solve the exercise listed at the bottom of page 102.

Reading Assignment 10: Chapter 10 (Dictionaries).

Questions: What is a dictionary. Are dictionaries sequences? Is there any such thing as the empty dictionary? How can you add, delete, modify entries in a dictionary? In what circumstances are dictionaries useful? Define the terms key, and key-value pairs.

Is a dictionary suitable for storing username-password pairs? How would you do that using regular lists?

Look at the code below and explain what it illustrates:

```python
>>> hash = { 'one' : 1, 'two' : 2, 'three' : 3 }
>>> hash
{'three': 3, 'two': 2, 'one': 1}

How could you list the entries in this dictionary in the order you want?

---

47 Fix it if there’s anything wrong with it. Explain why it’s correct if there’s nothing wrong with it.
48 It’s up to you what numbers (floating-point or integers) you generate.
49 Individual deviations.
50 Such as in the order: one, two three.
Implement matrix multiplication using dictionaries.

Implement the following procedure to construct magic \( n \)-by-\( n \) squares:

- it works only if \( n \) is odd.
- Place a 1 in the middle of the bottom row.
- After \( k \) has been placed in the \((i, j)\) square, place \( k+1 \) into the square to the right and down, wrapping around the borders.
- However,
  1. if the square to the right and down has already been filled, or
  2. if you are in the lower right corner,

then you must move to the square straight up instead.

Here's the 5-by-5 square that you get if you follow this method:

\[
\begin{array}{ccccc}
11 & 18 & 25 & 2 & 9 \\
10 & 12 & 19 & 21 & 3 \\
4 & 6 & 13 & 20 & 22 \\
23 & 5 & 7 & 14 & 16 \\
17 & 24 & 1 & 8 & 15 \\
\end{array}
\]

Check that the square above is magic.

Calculate the 3-by-3, 7-by-7 and 13-by-13 magic squares.

How do you find how big a dictionary is?

What is the \textit{keys} method doing and how do you invoke\(^{51}\) it\(^{52}\)?

How do you find out if a dictionary entry exists or not?

Is any dictionary an object? What do you think that means?

How do you modify a dictionary and keep a copy of the original?

Is the \textit{get} method offering any break if the entry does not exist in the dictionary?

How can a dictionary be used to speed up the calculation of fibonacci numbers? If in the original solution we waste a lot of time and in the dictionary implementation we get some of that back, is there any tradeoff we make (like spending some other kind of resource, that is related to the use of dictionary, to not duplicate any calculations)? Is it possible that this tradeoff bring us back in the same starting situation?

\(^{51}\) What is that? What do we mean by \textit{invocation}?

\(^{52}\) Same question about \textit{values} and \textit{items}.
What does it mean for a value to be global?

How long can a long number be? Can you calculate $2^{23456}$? How many digits does the answer have? How long does it take for the calculation to conclude? How does Python resort to long integers (does it detect the need automatically, do you need to use a special syntax like in the case of strings and floats, etc.)?

Write a program that determines the frequency of occurrence of letters in a string.

Write a program that determines the frequency of occurrence of vowels in a string.

How can you list the entries in a dictionary in alphabetical order?

Give examples of `extend`, `reverse` and `append` in a list context.

Reading Assignment 11: Chapter 11 (Files and exceptions).

Questions: Write programs to calculate:

- the number of lines in a file
- the number of words in a file
- the number of characters in a file
- the average number of characters per line in a file
- the average number of words per line in a file
- the average number of characters per word in a file

Write a program that makes a copy of a file.

Briefly describe how you open a file for reading, how you open a file for writing, and give short but complete descriptions of the `read` and `write` functions. How can you exit a loop even when the loop’s test is `True`? How can you exit an infinite loop?

What’s the difference between `readline`, `readlines`, `read`, `write` and `writelines`?

Write a program that makes a copy of a file by filtering a certain set of words, listed in a dictionary; every time such a word appears it should be replaced in the filtered copy by a number of asterisks equal to the number of letters in the word.

Write a program that plays Hangman with the user; have the program choose the secret word randomly from a list of words in a given file.

Describe two ways in which you can write the integer 52 in a file. Define the terms: format operator, format string and format sequence. Are there any format sequences for integers, floats, strings? Do they have any default attributes?

Describe what options you have to specify the format of numbers in Python.
Write a program that reads a file containing votes: each line contains a name. Have the program print the list of candidates in descending order of the number of votes received.

Assume in the problem above each name is followed by a number between -10 and 10. Now make the program report the sum of the votes for each candidate, and list the candidates in that order (descending by sum of votes/points received).

Same as above only sort by average and format your output like on page 120.

Can Python I/O handle directories (as opposed to, or in addition to text files)?

What is pickling? Write a program that illustrates the notion\(^{53}\).

Write a program that stores a whole dictionary, then retrieves it.

What are exceptions?

What are they good for?

Write a program that reads numbers from the user, one per line, and keeps a running average. Make the program robust with respect to the user data entry: if the data entered cannot be parsed as a number have the program write a warning, without crashing, and giving the user a chance to re-enter the number.

**Reading Assignment 12: Chapter 12 (Classes and objects).**

**Questions:** Explain what happens in the code below:

```python
class Potato:
    pass

a = Potato()
a.x = 2

b = Potato()
b.y = 3

print a.x
print b.y
```

Are objects like dictionaries?

Are the two instances in the code above created equal?

Are they still equal by the time the code is finished? Is this good or bad (or irrelevant?).

\(^{53}\) Store a few values then retrieve them in that order.
Consider the following code:

```python
class Potato:
    pass

a = Potato()
a.x = 2

def fun(n):
    if n == 0:
        return n
    else:
        return n + fun(n-1)

a.fun = fun

print a.fun(10)
```

Explain what happens in it. Can you achieve the same using a dictionary?

```python
a = {}
a['x'] = 2

def fun(n):
    if n == 0:
        return n
    else:
        return n + fun(n-1)

a['fun'] = fun

print a['fun'](10)
```

Isn’t that ugly\(^{54}\)? Please explain. What is the solution?

Consider the following code and explain how it relates to the code above:

```python
class Potato:
    def __init__(self):
        self.x = 0
    def fun(self, n):
        if n == 0:
            return 0
        else:
            return n + self.fun(n-1)

a = Potato()
b = Potato()
c = Potato()

print a.fun(10), b.fun(10), c.fun(10)
```

\(^{54}\) It is in fact a combination of powerful and ugly. Powerful sometimes means beautiful. So there.
Define: instances, objects, instantiation. Can you implement the functionality of class Point using a dictionary instead? Is it better or worse?

```python
>>> a = {}
>>> id(a)
13590576
>>> a['one'] = "sdhf"
>>> a['two'] = "d9f6"
>>> id(a)
13590576
>>> a['three'] = "sdvc"
>>> a
{'three': 'sdvc', 'two': 'd9f6', 'one': 'sdhf'}
```

Can a dictionary be taught to report its `id`?

In other words can you implement this behavior with dictionaries?

```python
class One:
    def whoami(self):
        return id(self)

a = One()
b = One()
print id(a) == a.whoami()
print id(b) == b.whoami()
```

Is this close in spirit to your solution:

```python
>>> a = {}
>>> a['self'] = id(a)
>>> a
{'self': 13590576}
>>> id(a)
13590576
>>> 
```

Define a function that takes two points as arguments and reports the distance between them. Points are created and initialized outside, they each contain an x and a y coordinate.

Does shallow equality imply deep equality? Is the reverse true?

Define a function that takes two rectangle objects (whatever your design choice is for rectangles) and decides if they overlap or not. Same problem for circles.

Define a class Circle as a Point (for the center) and a float (the radius).

Define a class Line as two Points.

Define a class Triangle as being three Lines coming out of three Points.
Define a function that finds the midpoint\textsuperscript{55} of a segment\textsuperscript{56}.

Define a Fraction\textsuperscript{57} as a pair of numbers.

Define a class Robot that has a direction and can move forward. Then create such an object and simulate a random walk for it\textsuperscript{58}. When the random walk is over report the distance from the robot’s location and the location where the random walk started.

Write a function that can translate a circle, rectangle, triangle object.

Describe the difference between shallow and deep copying.

**Reading Assignment 13: Chapter 13 (Classes and functions).**

**Questions:** Solve the exercises on page 137 bottom, 138 top. Give an example\textsuperscript{59} of a pure function that adds two Fractions, producing a new one. Make sure that your function produces results in their lowest terms. Write a modifier function that changes a Fraction object by adding a second Fraction to it. Make sure the result is in its lowest terms.

Define functional programming style. What does the book have to say about it?

Discuss prototype development vs. planned development.

Explain why multiplying a digit n by 9 gives a two-digit number where the first digit is n-1 and the second one is 10-n (see comment on page 142, the section on algorithms).

Design an algorithm to calculate the square root of a positive number.

**Reading Assignment 14: Chapter 14 (Classes and methods).**

**Questions:** Explain what the following code does:

```python
class One:
    def fun(x):
        return x

a = One()
print a.fun()
```

What is (by convention) the first argument to an instance method?

Define a class of objects that implement the game of Nim.

---

\textsuperscript{55} That is, it returns a Point.
\textsuperscript{56} A segment is the same as a Line in this context.
\textsuperscript{57} Define a gcd function that finds the greatest common divisor of two numbers.
\textsuperscript{58} There’s more than one way to do it, so implement the most appealing of these.
\textsuperscript{59} That is, define.
Here’s how my game runs:

```python
>>> game = Nim(50)
>>> game.move()
How many? 24
The height is now: 26
The computer moves.
>>> game.move()
The computer chooses 7
The height is now: 19
The user moves.
>>> game.move()
How many? 9
The height is now: 10
The computer moves.
>>> game.move()
The computer chooses 3
The height is now: 7
The user moves.
>>> game.move()
How many? 1
The height is now: 6
The computer moves.
>>> game.move()
The computer chooses 2
The height is now: 4
The user moves.
>>> game.move()
How many? 2
The height is now: 2
The computer moves.
>>> game.move()
The computer chooses 1
The height is now: 1
The user moves.
>>> game.move()
user has lost the game.
New height for new game:24
>>> game.report()
The height is now: 24
The user moves.
```
5. Design and implement a class called Coin that represents a coin that can be flipped, showing either heads or tails. Create a driver class, called CoinFlip, whose main method flips a coin 100 times to determine how many times each side comes up, and test your class:

```python
Coin coin = new Coin();
for (int i = 0; i < 100; i++)
    coin.flip();
coin.report();
```

Use object-oriented programming to solve the following problems:

- Write a program to solve the Point problem. Define a class of objects called Point (in the two-dimensional plane). A Point is thus a pair of two coordinates (x and y). Every Point should be able to calculate its distance to any other Point once the second point is specified.
- Define a class of objects called Line. Every Line is a pair of two Points. A Point is a pair of two numbers (the Lines are also in the plane in these exercises). Points should be able to determine their distance to other Points (see above). Lines are created by passing two Points to the Line constructor. A Line object must be able to report its length, which is the distance between its two end points. Make length a method and write a test program in which you create a few Lines and ask them to report their lengths.
- Define a class of objects called Triangle. A Triangle should be a set of three Lines (which for the purpose of this problem should be a very adequate representation). However, a Triangle is created by specifying three Points (which are located in the plane as discussed above). Using Heron’s formula every Triangle should be able to calculate and report its area. (If the three Points are collinear the Triangle is extremely flat, its area is 0 (zero), and that should be acceptable.)
- Define a class of objects called Clock. An object of type Clock stores time (hours and minutes) in military time format, in two instance variables of type int. Objects of type Clock have two instance methods: report which is simply returning the time, and tick which advances the clock by one minute. The constructor for class Clock takes one argument, a String that represents the time the clock is originally set to. Write a test program too, that illustrates how your objects are working (tick the clock 10-20 minutes and show it).
- Define a class of objects called Player that could be used in a Paper Scissors Rock game. Such a Player object should have a method, called makeGuess that
could be used to generate (randomly) one of the following guesses: "paper", "rock", or "scissors". The guess made by this method should be stored in an instance variable as well (a String, called guess). Another method of class Player should be able to compare the choice of the player it belongs to with the choice of any other Player and determine if the first player's guess is stronger than the second player's guess. Call this method strongerThan and make it return true or false. A report method should return the guess instance variable for printing purposes.

- Design an Elevator class for objects of this type that go up and down in a building with 100 floors.
- Nobody bounces like Tigger! Years of research have finally revealed the special mechanism of Tigger's bouncy step. You are to design a Tigger class that implements this unique movement, which I will describe below. A Tigger always starts in a random point (with coordinates x and y). When it decides to bounce a Tigger changes its x and y by the following rule(s): x becomes the sum of the squares of its digits and y becomes the sum of the squares of its digits. Example: if x is 37, then one bounce turns x into 32 + 72 (= 58). Both x and y change by this rule. And the bounce goes on (as expected).

Describe optional arguments. Solve the first exercise on page 150.

Clearly identify the initialization methods in the list of problems provided above.

Define a class Fraction whose initialization method ensures that the fraction created is already in its lowest terms. Teach these fractions to add, subtract, multiply and divide. Add a __str__() function that prints Fractions nicely (-2 / 3 or some such thing).

Overload +, -, * and / for Fractions.

Define polymorphism. Give an example.

**Reading Assignment 15:** Chapter 15 (Sets of objects).

**Questions:** How do you define static variables? (page 159 now)

**Reading Assignment 16:** Chapter 16 (Inheritance).

**Questions:** What does

**Reading Assignment 17:** Chapter 17 (Linked Lists).

**Questions:** What does

**Reading Assignment 18:** Chapter 18 (Stacks).

**Questions:** What does
Reading Assignment 19: Chapter 19 (Queues).

Questions: What does

Reading Assignment 20: Chapter 20 (Trees).

Questions: What does

Appendix B.

Miscellaneous: