This is your alternate Homework One for A202/A598 Spring 2007.

Text: How to think like a computer scientist (learning with Python[1]).

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:

```
n = 3
m = 5
n = n + m
m = n - m
n = n - m
```

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[2]? 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. 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.

---

[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.

Write Python expressions that verify the following identities:

a) $10^{\log x} = x$ 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)
 C      _)
 | ,___|
 |   /
/____\
/      \
```

Write another function, called `three`, which produces the ASCII art shown above three times, one below the other[3]. How short can you make a program that prints one hundred Bart Simpsons[4]?

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[5]. Call this `next` with the following arguments: `1`, `2`, `3`, `2+1`, `3-1`, `-2`, and report[6] the results. Call `next` again on `x`, `y`, `z` where `x` is `2`, `y` is `3` and `z` is `x-y`. Write a function called `add` that receives two arguments (integers), adds them up in a local variable `x` and prints `x`. Is `x` available outside `add`? What happens if you call `add` with two string arguments?

What is a stack diagram? Does it resemble[7] the history of pages visited by a web browser? Draw[8] the stack diagram of the following call:

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

Can you quickly[9] evaluate the call? What is a Python `traceback`?

Consider the following Python program:

---

[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.

```
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:

```
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:

```
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:

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

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

Turn in a Word document with all the questions answered completely.

---

[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`.