CIS 130 - Intro to Programming
Week 10 Lab - Wednesday, 03-28-07
Week 10 Lab Exercise

Purpose: get more practice with arrays/lists and file input/output

YOU MAY WORK IN PAIRS FOR THIS LAB. Make sure you understand all the concepts!

NAME(S) _____

CHECK YOUR ANSWERS with at least one other student/pair of students. (This does NOT include the person you worked with, if you worked in a pair.) Discuss any differences in your answers until you are both satisfied with what the answers should be (and you may change your answers as a result).

Who you checked your answers with: _____

Answer the following in the space provided:

1. There is a file named **numbers.txt** in the current working directory. You want to open it for reading, storing the file returned in a variable **infile**. Write a Python statement below that will accomplish this.

_____

2. You want to read a line of this file into a variable **next_line**. Write a Python statement below that will accomplish this.

_____

3. You want to close this file (you are finished reading from it). Write a Python statement below that will accomplish this.

_____

4. But, **next_line** contains a lovely value - we want to write it to another file, **stuff_read.txt**. Write a Python statement below that will open this file for **writing**, deleting any existing contents and storing the file returned in a variable **outfile**.

_____

5. Now write the value **next_line** to this file.

_____

6. And, write a Python statement that will now close this file.

_____

7. Several other things get done, and now you want to add something to **stuff_read.txt** --- BUT you don't want to delete its current contents. Write a Python statement below that will open this file for **appending**, NOT deleting the existing contents and storing the file returned in a variable **apfile**.

_____

8. You want to write **27** to this file, but also a newline character. Write a Python statement below that will do so.

_____

9. And, write a Python statement that will now close this file.

_____

10. Now, on to arrays/lists! Write a Python statement that will set the variable **listy** to be a list containing the values 1, 3, 15, and 8.

_____

11. You decide you want to add 23 to list **listy**. Write a Python statement that will do so.

_____

12. You find that you need to change the value **3** in this list to instead be **13**. Write a Python statement that will do so.

_____

13. Three more items are added to **listy** (so that now it contains **8** items in all). Write a Python statement that will print JUST the last element of the list **listy**.

_____

14. And, write a Python statement that will print the entire list **listy**.

_____

15. You can probably imagine that it is occasionally useful to read numbers from a file into a list.

Write a function **read_nums** that takes a string as argument, the name of a file in the current directory expected to contain one integer per line. The function opens the file and reads its contents into a list, and then **returns** a list of the numbers in this file as its result.

(Be careful -- the list should be a list of **numbers**, not a list of strings. How can you obtain the number equivalent to a number-formatted string (read from the file)? Note that it turns out that this means will also happily eliminate that pesky newline from the end of the string read...)

In the interests of lab time, I will just run your **read_nums** to see if it works. Create a file in the current working directory named **test.txt** that contains the numbers **5, 8, 2, 7**, **1**, **4**, and then you will show me the results of the following when it is your turn:

read_nums("test.txt") == [5, 8, 2, 7, 1, 4]

16. Remember **line_of_X** from HW #5? (If not, you can remind yourself by looking at the "Some solutions" section of the public course web page.)

Write a function **bar_chart** that expects a list of numbers as its argument; it should use appropriately use **line_of_X** to print a line of X's as long as each value in that list. For example, **bar_chart([1, 8, 5, 3])** would cause the following to be printed to the screen:

```
X
X X X X X X X X
X X X X X
X X X
```

When it is your turn, you will show me the results of the following call: **bar_chart([1, 8, 5, 3])**

17. Now do you see how you can combine **read_nums** and **bar_chart** to print to the screen a bar chart of all of the numbers from a given file?

Write a function **file_chart** that expects as its argument the name of a file in the current directory containing one value per line, that then prints to the screen a bar chart corresponding to the values in that file. The body of **file_chart** should use **read_nums** and **bar_chart**, and if its body is longer than 2-3 lines then you are over-complicating it...

When it is your turn, you will show me the results of the following call: **file_chart("test.txt")**
Its results should look like:

```
X X X X X
X X X X X X X X
X X
X X X X X X X
X
X X X X
```

18. Finally, you have a client who is terrified of pico, but willing to use the Python interpreter. (Go figure!)

He/She wants a Python function **create_file** that will accept the name of a file as its argument, and then prompt him/her for lines of input, which should then be written to that file (existing file contents from before this call can be cheerfully deleted). It should keep prompting until he/she enters the line **"stop"**.

When it is your turn, I will watch you run **create_file("try.txt")**, type in a few lines and then stop when prompted, and then look at the contents of "try.txt".

NOW write your name(s) on the **NEXT:** list on the board. (You write your name on this list if you have questions along the way, as well as when you are done; I'll then work my way down the list.) You need to complete the above and have it checked by the end of lab.