University of Washington
Department of Electrical Engineering

Name_____

Working with stdin and stdout, Variables and Data Types, and Control Flow

1. Let's continue with the travel problem. Using the library functions *printf()* and *scanf()*, prompt the user for the duration of the flight to Europe, in hours, as a floating point number. Based upon the number of air miles from Seattle to London, compute, then print out the estimated velocity of the aircraft as a floating point number. Using the velocity you computed and an estimated head wind of 89.6 miles per hour, compute, then print out the estimated duration of the flight as a floating point number.
   Use symbolic constants in your design rather than magic numbers as appropriate.
   Remember, the *scanf()* function delimits input by whitespace.

2. To help Bill in engineering stores stay within his budget as he is ordering new chairs and lab stools for the EEB 137 lab, write a program to determine whether a series of purchases will come in over or under budget.
   Chairs cost $435.00 each and lab stools $565.00 each, the shipping cost is $583.00.
   The program starts when Bill enters a budget and sales tax rate. As he is writing out the purchase order for Office Depot, he enters an input line containing the price of an item, the number purchased, and a discount rate. The program computes and prints the total cost for each item and adds the shipping cost.
   After Bill has finished putting the purchase order together, he enters the EOF character (crtl z on a PC, ctrl d in Linux). The program then prints the total cost of the purchase order and how it compares with the budget.
   If he is under budget, the program informs him and tells him the amount of money remaining and how many more chairs or lab stools he can purchase, if over, it indicates so, by how much and how many chairs or lab stools he will have to remove from his order, if spot on, it says *Yeah Bill Good Job – Go Get a Beer*.

3. The following program to compute the average of a collection of values produces incorrect results if the number of values is greater than INT_MAX, if any input value is greater than INT_MAX, or if the sum is greater than LONG_MAX. Rewrite the program to avoid these problems.

```
#include <stdio.h>

int main()
{
    int    next = 0;          // next input value
    long   sum = 0;           // running total
    int    n = 0;             // number of input values
    int    result = 0;        // did we read another value?
    double avg = 0.0;         // average of input values

    printf("Enter a series of numbers to be averaged\n");

    //  read input
    //  test for integer type entered

    while (1 ==(result = scanf("%i", &next)))
    {
        sum = sum + next;   //  running sum
        n = n + 1;          //  number of values entered
    }

    if (result != EOF)        // combination of ctrl and z keys on a PC or ctrl and d in Linux
    {
        printf("Warning: bad input after reading %i values\n", n);
    }

    if (0 == n)               //  check for no numbers entered
    {
        avg = 0.0;
    }
    else                      //  compute the average
    {
        avg = (double) sum / n;
        printf("Average of %i values is %f.\n", n, avg);
    }

    return 0;
}
```

4. Write a program that reads input data from the user, one character at a time, until the user enters the EOF (crtl z on a PC, ctrl d in Linux) character. As data is being read, count the number of words and punctuation characters that have been entered. When the user has finished, print out the total number of words and punctuation characters that have been entered.

   A word is any sequence of non-whitespace characters, except punctuation characters, separated by whitespace characters. Look at the character testing functions in the library ctype.h and in Chapter 5 of the text.

5. Modify the program in the previous problem to print each input word on a separate line as the data is being entered. Do not include the punctuation characters.