Aircraft Weight and Balance

In this exercise you will write a simple program to perform a weight and balance calculation for a small aircraft.

An aircraft will not fly correctly if it is over its maximum gross weight, or if the center of gravity (CG) is not within a prescribed range. It is the responsibility of the pilot to determine before each flight that the aircraft weight and center of gravity are within acceptable limits.

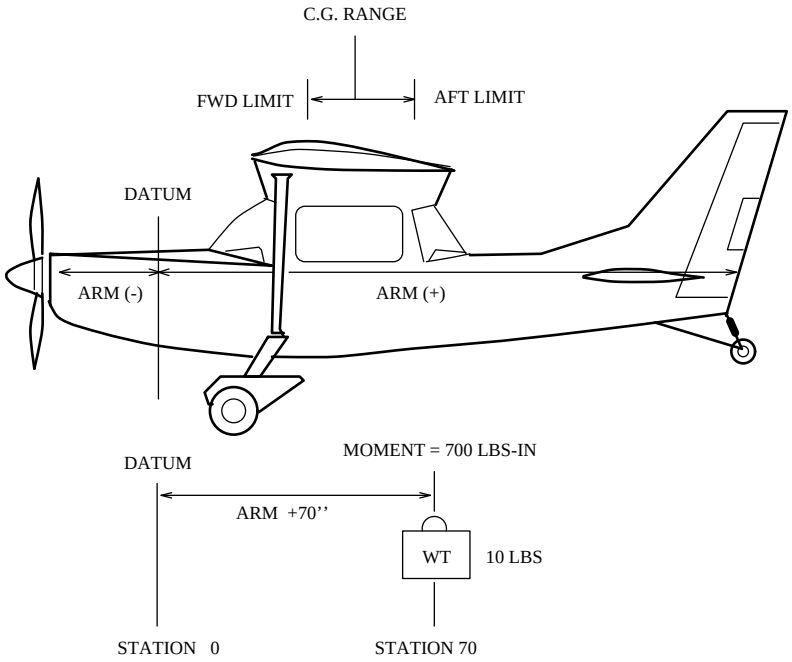

Figure 1: Weight and balance illustrated.

The gross (total) weight is simple to compute: add the empty weight of the aircraft, the weight of the pilot and passengers, the weight of any baggage, and the weight of the fuel. If w_i is the ith item out of these four things then the total weight W is

$$W = \sum_{i=1}^{4} w_i$$

The center of gravity is almost as simple to calculate. Each of the items mentioned above has a "moment," which is the product of the weight of the item and it's "arm," which is

the distance fore (-) or aft (+) of the object from some reference point (often the firewall between the engine and the cabin). If d_i is the distance of the ith object (or "station") from the reference point, then the position of the center of gravity, d_{cg}, is the total moment divided by the total weight,

$$d_{cg} = \frac{1}{W} \sum_{i=1}^{4} (w_i \times d_i) \ .$$

1. Problem

Write a Fortran program which will read the necessary data for a small plane (as specified below) and compute and display both the gross weight and the position of the center of gravity.

It may help you to know that your program does not have to be as complicated as the `AVGVAR` program. Just write the simplest program you can, which reads in the necessary data and prints the answers. It may help you to organize your program into three parts: (1) read in the data, (2) perform the calculations, and (3) print the answers.

1.1. Input

Read the station arm and weight (in that order, as a pair of numbers!) for each of:

1. aircraft
2. pilot and passenger
3. baggage
4. fuel

Be sure to read the data in the order given here. You may assume that the station arm is measured in inches and the weight is measured in pounds.

Each time you read in a pair of data items you should immediately print out the values read (that is, you should "echo" the values read). This is useful to confirm that the values you intended to enter were read correctly, and it also insures that the values will be printed if your program is run as a "batch" job, or if you feed data to your program using "I/O redirection" in Unix.

1.2. Output

Print the gross weight of the loaded aircraft and the position of the center of gravity.

Your output will look better if your program begins by explaining what it is doing, and if you arrange and label your results so that they are easy to read.

2. Testing

To test your program you may use the following sample dataset, which is typical for a Cessna 150:

Item	Weight (lbs)	Arm (inches)
Aircraft	1107	34.15
Pilot and passenger	340	39.12
Baggage	10	63.77
Fuel	135	42.22

3. Counting Columns

In ancient times (the 1960's and 70's) Fortran programs were punched line by line onto computer cards, and then the deck of cards was fed into a card reader to load the program into the computer. Back then, the Fortran compiler was very picky about what could go in certain columns of the cards. Today's compilers are more lenient, but you still must follow some general guidelines and put things in appropriate columns.

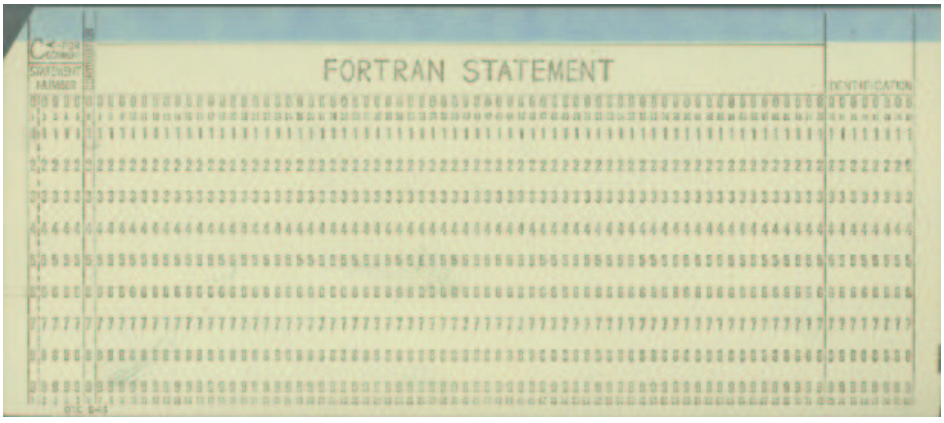

Figure 2.1: Computer punch card showing Fortran column assignments.

Strictly speaking, columns 1 through 5 of each line are reserved for line numbers, while Fortran statements must appear within columns 7 through 72. Anything after column 72 is ignored. If you must write a line longer than this, you can put any non-blank character in column 6 of the line following and continue on that line. (Any character will do, but "`&`" is recommended because it isn't used anywhere else in Fortran and because it means "and".) Lines which begin with "`C`" or "`*`" in column 1 are considered to be comments and are ignored. To summarize:

Column(s)	**Meaning**
1	Comment
1-5	Line numbers
6	Continuation
7-72	Fortran Statements
72-80	Ignored

In Fortran mode in emacs you can get a "Fortran Ruler" on your screen by pressing `^C-^R`. It will go away with the next key you press. Note also that you don't *have* to begin in column 7, you could begin in column 8, which is where you'll be if you simply press the TAB key once; So using TAB is an easy way to be sure you start in the right place.

Some Fortran compilers will consider any line which begins with "`&`" to be a continuation of the previous line. You might try a quick experiment to see if your compiler is one of these.

4. Style Notes – Comments

Blank lines in a Fortran source file are ignored, and any line which begins with a "C" in column 1 is considered to be a "comment" and is also ignored.

It is considered good style to include comments in your programs so that others may read and understand what your program does. After all, programs are read by people too, not just computers. In the least, at the beginning of your programs you should include comments which describe what the program does, who wrote it (your name and perhaps even your e-mail address), the date it was written (or finished, or begun), and where you are or who you work for (i.e. "Physics Department, Vassar College"). For the purposes of this course you should also explicitly state which exercise your program was written for.

It is also a good idea to leave blank lines in programs between separate blocks of code (think of the blocks as "paragraphs"), and to include comments at the beginning of major blocks to describe what is going on. These comments are for anybody who will read the program, which includes you. When you re-read a complicated program six months from now you will appreciate any hints you have left yourself describing what you have done. The level of detail in your comments is a matter of personal style, but not putting any comments in a program shows a distinct lack of good style.

It is also a good idea to begin every program with a `PROGRAM` statement, even though the Fortran compiler does not require it.

5. Reading

You should read in your book about the simplest forms of the `PRINT` and `READ` statements, and about arithmetic operations (or arithmetic computations) and the "assignment" statement. You should also read about the `PROGRAM` statement.