

**Using conditional statements and input-failure loop**

**Effect of Gravity**

It is finally time to write a complete program. This project will use many of the C++ features that were illustrated in the source code you were given for the first and second programming assignments. The resulting program will read names and weights of people from an input file. It will also read in a planet name. The program will then write out to a simple output file the weights of the people on that particular planet based on the gravity on that planet. This project is based on Programming Problem 2 from the textbook.

**Sample input data**

Here is a sample input file, named `planetdata.txt`, for the program. The first two lines are for descriptive purposes and do not contain any input data. Each of the following lines in the input file contains exactly three fields. The first field is the name of a cartoon character, the second field is his/her weight and the third field is the name of a planet. The three fields are separated by a single tab (\t) character.

Name	Weight	Planet
=====	=====	=====
Bugs Bunny	65	Pluto
Donald Duck	89	Venus
Mickey Mouse	90	Earth
Roadrunner	76	Mercury
Minnie Mouse	80	Venus
Goofy	99	Pluto
Wile E. Coyote	78	Neptune
Elmer Fudd	98	Uranus
Yosemite Sam	45	Mars
Marvin Martian	100	Jupiter
Jiminy Cricket	10	Jupiter
Uncle Scrooge	120	Moon
Foghorn Leghorn	45	Mercury
Daffy Duck	98	Venus

There can be a varying number of comic characters specified in the input file. You should not design your program for a specific number of input data lines. There is guaranteed to be at least one character in the input file. The input values are guaranteed to be syntactically and logically correct. This means that there will be a tab character between the three input fields.

**Calculations:**

The new weight of the comic character on any given planet is calculated based on the table shown below. For example, if somebody weighed 100 lbs, his/her weight on Neptune will be  $100 \times 1.1794 = 117.94$  lbs.

Planet Name	Gravitation Factor
Mercury	0.4155
Venus	0.8975
Earth	1.0000
Moon	0.1660
Mars	0.3507
Jupiter	2.5374
Saturn	1.0677
Uranus	0.8947
Neptune	1.1794
Pluto	0.0899

Given the data in the input file and the table above, the results should be written out to an output file.

**Sample output:**

Here is a sample output file, named `weights.txt`, which corresponds to the input data given above:

```
Programmer: Mir Farooq Ali
CS 1044 Summer II 2004 Project 3

Name           New weight    Planet
=====
Bugs Bunny     5.8435       Pluto
Donald Duck   79.8775      Venus
Mickey Mouse  90.0000      Earth
Roadrunner    31.5780      Mercury
Minnie Mouse  71.8000      Venus
Goofy          8.9001       Pluto
Wile E. Coyote 91.9932      Neptune
Elmer Fudd    87.6806      Uranus
Yosemite Sam  15.7815      Mars
Marvin Martian 253.7400     Jupiter
Jiminy Cricket 25.3740      Jupiter
Uncle Scrooge  19.9200      Moon
Foghorn Leghorn 18.6975     Mercury
Daffy Duck     87.9550      Venus
=====
Sum of weights = 889.1409
```

The first two lines identify the programmer and project, as usual. Then there is a blank line, followed by a descriptive header and a separating line. Then, there is a line for each comic character that lists its name, new weight and the planet for which the weight has been calculated. Finally, the sum of the new weights is printed out after a separator line.

If you have read the description of how the Curator scores your program in the *Student Guide*, you know that is important that you use the same spelling and capitalization for all the labels shown above. The horizontal spacing does not effect scoring unless you combine things that should be separate or separate things that should be combined. You are free to experiment with the horizontal layout but you should try to align the columns neatly. Note that you must insert blank lines and divider lines as shown.

**Evaluation:**

Everything that was said in the specification for the earlier about testing still applies here. Do not waste submissions to the Curator in testing your program! There is no point in submitting your program until you have verified that it produces correct results on the sample data files that are provided. If you waste all of your submissions because you have not tested your program adequately then you will receive a low score on this assignment. You will not be given extra submissions.

Your submitted program will be assigned a score, out of 100, based upon the runtime testing performed by the Curator System. We will also be evaluating your submission of this program for documentation style and a few good coding practices. This will result in a deduction (ideally zero) that will be applied to your score from the Curator to yield your final score for this project.

Read the *Programming Standards* page on the CS 1044 website for general guidelines. You should comment your code in the same manner as the code given for the first two programming assignments. In particular:

- You should have a header comment identifying yourself, and describing what the program does.
- Every constant and variable you declare should have a comment explaining its logical significance in the program.
- Every major block of code should have a comment describing its purpose.
- Adopt a consistent indentation style and stick to it.

Your implementation must also meet the following requirements:

- Choose descriptive identifiers when you declare a variable or constant. Avoid choosing identifiers that are entirely lower-case.
- Use named constants instead of literal constants when the constant has logical significance.
- Use C++ streams for input and output, not C-style constructs.
- Use C++ string variables to hold character data, not C-style character pointers or arrays.
- **Note: you are explicitly forbidden to write any user-defined functions for this program. This will make the program somewhat repetitive, and physically longer than the alternative. To some extent, that's the reason for this restriction.**

Understand that the list of requirements here is not a complete repetition of the *Programming Standards* page on the course website. It is possible that requirements listed there will be applied, even if they are not listed here.

### Submitting your program:

You will submit this assignment to the Curator System (read the *Student Guide*), and it will be graded automatically. Instructions for submitting, and a description of how the grading is done, are contained in the *Student Guide*.

You will be allowed up to five submissions for this assignment. Use them wisely. Test your program thoroughly before submitting it. Make sure that your program produces correct results for every sample input file posted on the course website. If you do not get a perfect score, analyze the problem carefully and test your fix with the input file created on the curator page, before submitting again. The highest score you achieve will be counted.

The *Student Guide* and the submission link can be found at:

<http://eags01.cs.vt.edu:8080/CuratorS04/index.jsp>

### Pledge:

Each of your program submissions must be pledged to conform to the Honor Code requirements for this course. Specifically, you **must** include the following pledge statement in the header comment for your program:

```
// On my honor:  
//  
// - I have not discussed the C++ language code in my program with  
//   anyone other than my instructor or the teaching assistants  
//   assigned to this course.  
//  
// - I have not used C++ language code obtained from another student,  
//   or any other unauthorized source, either modified or unmodified.  
//  
// - If any C++ language code or documentation used in my program  
//   was obtained from another source, such as a text book or course  
//   notes, that has been clearly noted with a proper citation in  
//   the comments of my program.  
//  
// - I have not designed this program in such a way as to defeat or  
//   interfere with the normal operation of the Curator System.  
//  
// <Student Name>
```

**Failure to include this pledge in a submission is a violation of the Honor Code.**