Bobcat Aerospace
Software Requirements Document

Software Name: Craft Designer
Author: Gentry Atkinson
Version: 1.0.0

1. Introduction
Product Purpose: this product is being developed for the purpose of facilitating space vehicle design. This software will be able to take input from an engineer or designer and generate simple statistics about one vehicle which will be saved in a text file.

2. Standards
Hardware: this product will be capable of running on any platform with a C++ compiler and a means of text I/O.

Schedule: completion of this product is expected no later than 15 July.

Language: this project will be implemented using the C++ programming language

3. System Description

System Context: this will be a self-contained piece of software that will not rely on external platforms for its basic operation.

User Characteristics: users will be trained aerospace designers who are proficient with computer operation. This project will not be responsible for user training.

4. Functional Requirements

Input: the product will collect the following information from the user through a simple text interface:

- **Designer Name**: the first and last name of the vehicle's designer
- **Ship Name:** a one word name for the vehicle
- **Ship Mass:** a numerical value representing the mass of a vehicle in kilograms
- **Engine Thrust:** a numerical value representing the force exerted by the vehicle's engines in newtons.

Output: the product should write the following values to a text file:

- **Ship Name**
- **Ship Mass**
- **Acceleration:** calculated using (**Engine Thrust/Ship Mass**)
- **Impulse:** calculated as **g0 * ln(Acceleration),** where **g0** is the constant gravitation value 9.8 and **ln** is the natural log function.
- **Maximum Altitude:** calculated as **(100 + (Impulse*200))/10**, with the result being the kilometers above sea level that the vehicle can safely travel.
- **Designer Name**

Interface: the product should format the output as follows:

- "File written for [Ship Name]" should be printed on the **console** with [Ship Name] replaced by the actual Ship Name.
- The program's output should be written to a **file** named [Ship Name].txt with [Ship Name] replaced by the actual Ship Name.
- The file output should be formatted as follows:

```
##########[Ship Name]##########      //10 #s before and after
                                     //blank line
Ship mass:          [Ship Mass]kg    //Values in neat columns
Acceleration:       [Acceleration]m/s2 //
Impulse:            [Impulse]Ns/kg   //
Max Altitude:       [Max Alt.]km     //
                                     //blank line
File saved at:      [time stamp]     //
Designed by:        [Designer Name]  //
```

  - All [Values] should be replace with correct values
  - Comments should not appear in the file.
  - Every item of output should have the correct unit as shown.

5. Non-functional Requirements

- The program should be submitted as a .cpp file with the name [Author's Last Name]_assignment1.cpp, with [Author's Last Name] replaced by the author's last name.
- The first three lines of the program should be the author's name, the date that the assignment was written, and the collaborators who helped on the project.
- The author is expected to follow the Style Guideline.
- Numbers should be formatted to one decimal place of accuracy.

6. Sample Test Cases:

Input
```
Robert Goddard
Zoomer1
1000
10000
```

Console
```
File written for Zoomer1
```

In file Zoomer1.txt
```
##########Zoomer1##########

Ship Mass:          1000kg
Acceleration:       10.0m/s2
Impulse:            22.6Ns/kg
Max Altitude:       461.3km

File saved at:      1620926125
Deigned By:         Robert Goddard
```

Input
```
Sergei Korolev
Boko
200
250
```

Console
```
File written for Boko
```

In file Boko.txt
```
##########Boko##########

Ship Mass:          200kg
Acceleration:       1.2m/s2
Impulse:            2.2Ns/kg
Max Altitude:       53.7km

File saved at:      1620956317
Designed by:        Sergei Korolev
```

7. Guidance

- The <**iomanip**> library can be used to set the precision of decimal numbers using the **fixed** and **setprecision** keywords.
- The **log** function from the <**cmath**> library can be used to calculate a natural log (ln).
- The function **time(NULL)** from **<ctime>** can be used to produce a timestamp formatted in computer time. Printing a timestamp on the screen would look like:
    - **cout << time(NULL) << endl;**