

CEG 2170

Programming Assignment 2

Purpose of Programming Assignment 2

The purpose of this assignment is to gain experience with functions and conditional (if/else) statements. You will once again implement various formulas in this program, asking the user for input into those formulas, and outputting the results to the user. Except this time, you will use functions to perform all of the calculations and use if/else statements to make conditional decisions in your program. See the sample output for the expected results of your program.

General Requirements:

1. The main function should be exactly this:

```
int main() {  
    implementPart1();  
    implementPart2();  
    return 0;  
}
```

2. Create constants for the gravity constant, PI, and five different constants for the density values for the materials used in Part 2. You must also create a constant for an error state for use in Part 2.

a. Example: `#define ERROR -1`

3. You are required to show units in Part 1 and 2 for both input and output.

Part 1 – Computing a Projectile’s Flight

In Part 1, you are dealing with a projectile that is fired at a target. The projectile is fired with an initial velocity (v) and a given elevation angle (θ). Your program will use the projectile’s initial velocity, elevation angle, and distance to the target (d) to compute the duration of a projectile’s flight time (t) and its height above the ground (h) when it reaches the target, as follows. Note that g is the gravitational constant which should be assumed to be 32.17 ft/s² and should be treated as a constant.

$$\textbf{Formula \#1} \quad t = \frac{d}{v \cos \theta} \qquad \textbf{Formula \#2} \quad h = v t \sin \theta - \frac{g t^2}{2}$$

Note that the user will enter the angle in *degrees*, but you’ll have to convert angle to *radians* since the trigonometric functions in C require arguments in radians.

Part 1 – Requirements

1. Create three functions as follows:
 - a. `implementPart1` is called from `main` and does the majority of the work for Part 1.
 - b. `computeDuration` is called from `implementPart1` to compute the duration of the projectile’s flight time.
 - c. `computeHeight` is called from `implementPart1` to compute the height above the ground for the projectile.
2. In `implementPart1`, prompt the user for the initial velocity (f/s), elevation angle (degrees), and distance to the target (ft). Remember to convert the elevation angle to radians before using the

angle in your computations. Call the `computeDuration` and `computeHeight` functions to compute their respective values. Print the computed flight duration and height of the flight to the screen. Note that certain inputs may result in a *negative height*. Use an if/else statement to determine the validity of the height value (i.e., greater than or equal to zero) and either print an error message or the two computed values. See the sample output below for test data for each case.

3. Create a function named `computeDuration` to compute the duration of the projectile's flight time using Formula #1. This function accepts the distance, velocity, and angle (in radians) as its only parameters. This function returns the computed duration.
4. Create a function named `computeHeight` to compute the projectile's height above the ground using Formula #2. This function accepts the velocity, duration, and angle (in radians) as its only parameters. This function returns the computed height.

Part 2 – Shipping Weight

You work for a hardware company that manufactures circular, flat washers. To estimate shipping costs, your company needs a program that computes the weight of a specified quantity of flat washers. Note that a circular flat washer resembles a donut, that is, a circle with a circular cut out in the center. Each washer also has a certain thickness. If the outer diameter of the washer is d_2 and the diameter of the inner cut out is d_1 then the washer's rim area is computed as shown in Formula #3.



$$\textbf{Formula \#3} \quad \text{rim area} = \pi \left(\frac{d_2}{2} \right)^2 - \pi \left(\frac{d_1}{2} \right)^2$$

The washer's volume is the rim area multiplied by the thickness of the washer. The weight of the washer can then be calculated as the volume multiplied the density of the material from which the washer is manufactured. The company manufactures washers from the materials shown below in Table 1, which also shows the density for each material.

Table 1 - Washer Material Densities

Material	Density (gm/cm ³)
Nylon	1.15
Aluminum	2.70
Titanium	4.50
Steel	7.87
Brass	8.75

Part 2 – Requirements

Use the physical characteristics of a washer and the number of washers to be shipped to calculate the weight of the shipment.

1. Create three functions as follows:
 - a. `implementPart2` is called from `main` and does the majority of the work for Part 2.

- b. `computeRimArea` is called from `implementPart2` to compute a washer's rim area.
 - c. `computeVolume` is called from `implementPart2` to compute a washer's volume.
 - d. `computeWeight` is called from `implementPart2` to compute the weight of a single washer.
2. In `implementPart2`, prompt the user for the washer's outer rim diameter (cm), inner rim diameter (cm), thickness (cm), and the quantity of washers to be shipped. Call the `computeRimArea`, `computeVolume`, and `computeWeight` functions to compute their respective values. Print the computed weight of the total quantity of washers in grams if the weight was successfully computed; print an error message if the weight computation failed (i.e., the weight is equal to the error condition). See the sample output below for test data for each case.
3. Create a function named `computeRimArea` to compute a washer's rim area using Formula #3. This function accepts inner and outer diameters as its only parameters. This function returns the area.
4. Create a function named `computeVolume` to compute a washer's volume as the washer's rim area multiplied by its thickness. This function accepts the rim area and thickness as its only parameters. This function returns the volume.
5. Create a function named `computeWeight` to compute a washer's weight as the washer's volume multiplied by the density of material of which the washer is made. This function must provide the user with a list of materials and then compute the washer's weight based on the density of the washer's material – use the values specified in Table 1 for the prompts and computations. This function accepts the volume as its only parameter. This function returns either the computed weight for a valid material, or -1 (i.e., the ERROR condition) if an invalid material is specified. You must use constants for the density values and the ERROR condition.

Part 2 – Sample Program Interaction

Success Cases

```

"C:\Users\Public\Documents\CEG 2170\Program Assignments\PA2\Hutchison_PA2\bin\Debug\Hutchison_PA2.exe"
Part 1:
Enter initial velocity (f/s): 800
Enter elevation angle (degrees): 17
Enter distance to the target (ft): 11000
The duration of flight is 14.38 seconds
The height of flight is 37.71 feet

Part 2:
Enter the outer diameter (cm): 2.4
Enter the inner diameter (cm): 1.2
Enter the washer thickness (cm): .1
Enter the quantity of washers: 1000
Enter the type of material for the washer:
N - Nylon
A - Aluminum
T - Titanium
S - Steel
B - Brass

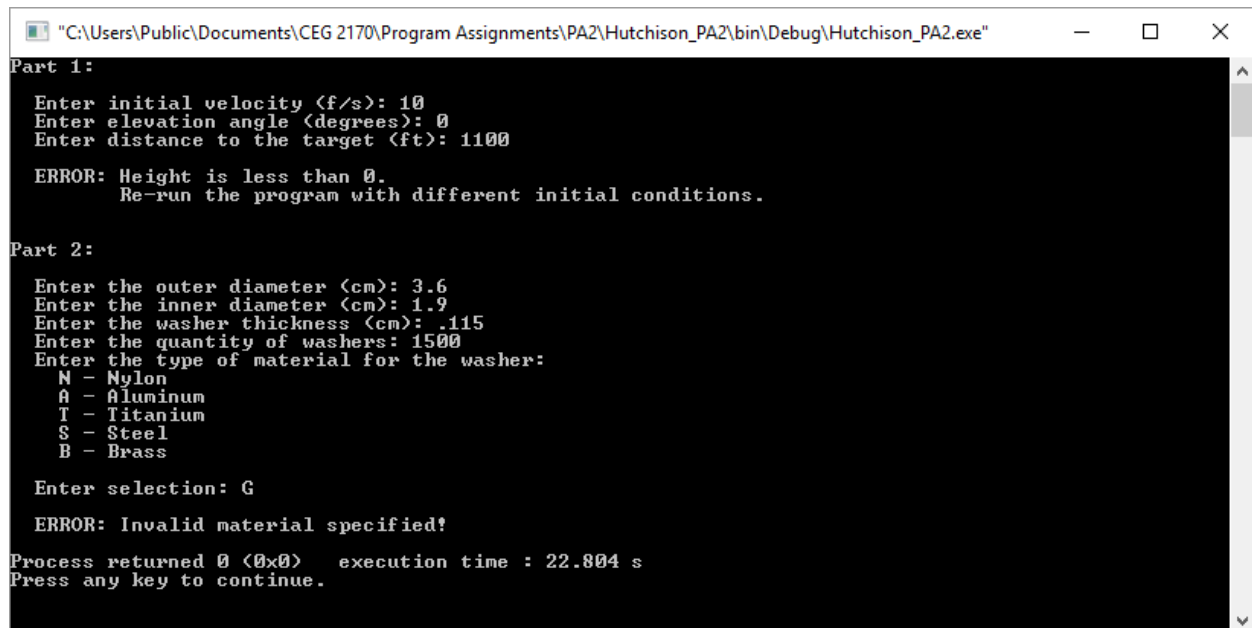
Enter selection: S

The weight of the washers is 2670.23 grams

Process returned 0 (0x0)   execution time : 21.176 s
Press any key to continue.

```

Error Cases



```
"C:\Users\Public\Documents\CEG 2170\Program Assignments\PA2\Hutchison_PA2\bin\Debug\Hutchison_PA2.exe"
Part 1:
Enter initial velocity (f/s): 10
Enter elevation angle (degrees): 0
Enter distance to the target (ft): 1100
ERROR: Height is less than 0.
        Re-run the program with different initial conditions.

Part 2:
Enter the outer diameter (cm): 3.6
Enter the inner diameter (cm): 1.9
Enter the washer thickness (cm): .115
Enter the quantity of washers: 1500
Enter the type of material for the washer:
N - Nylon
A - Aluminum
T - Titanium
S - Steel
B - Brass

Enter selection: G
ERROR: Invalid material specified!

Process returned 0 (0x0)   execution time : 22.804 s
Press any key to continue.
```

What to Turn In

Upload your zipped (compressed) project to the Dropbox on Pilot. This will be a single **main** function with the code required for Parts 1 and 2 above. Be sure your program follows the guidelines given on the Style Requirements document (provided on Pilot) with respect to commenting, variable naming, indenting, etc.

Grading Rubric

General documentation/style (12 points possible; add all that apply):
Complete header information at top of program (3 pts)
Correct indenting (3 points)
Correct use of white space; in general, code is single-spaced with a blank line between sections (such as between variable declarations and input; between input and calculations; between calculations and output). This applies to main and all other functions (3 pts)
Clean and organized presentation of your program input/output presented to the user (3 pts)
Part 1 (30 points)
Prompt for and accepts input from the user (6 pts)
Function calls to compute duration and height (8 pts)
Conditional check for the height value (8 pts)
Proper implementation of the computeDuration function (4 pts)
Proper implementation of the computeHeight function (4 pts)
Part 2 (40 points)
Prompt for and accept input from the user (8 pts)
Function calls to compute rim area, volume, and weight (6 pts)
Conditional check for value the weight value (6 pts)
Proper implementation of the computeRimArea function (4 pts)
Proper implementation of the computeVolume function (4 pts)
Proper implementation of the computeWeight function (12 pts)
Program and Main Function (18 points)
Include proper include statements (3 pt)
Meaningful variable names (5 pts)
Proper use of required constants (5 pts)
Proper specification of function prototypes (5 pts)