



CPT-182-83/R83 (23/SP) - Programming in C++

Exam 1 (40 Points)

Deadline: Thursday, February 23, by 07:59 AM (Code submitted on Canvas)

Project Overview

The **asteroid belt** is a torus-shaped region in the Solar System, located roughly between the orbits of the planets Jupiter and Mars (see Figure 1). It contains a great many solid, irregularly shaped bodies, of many sizes and masses. The National Aeronautics and Space Administration (NASA) is monitoring about 28,000 near-Earth asteroids that have some possibility to impact on Earth.

In this project, a celestial body (e.g., planet) is sent to the asteroid belt in order to destroy some asteroids. It will collide with each asteroid to destroy it.

Your task is to generate a solution to figure out what is the minimum mass of the celestial body in order to destroy all the asteroids. Your program will read in the masses of the asteroids from an input file, and write the result to an output file.

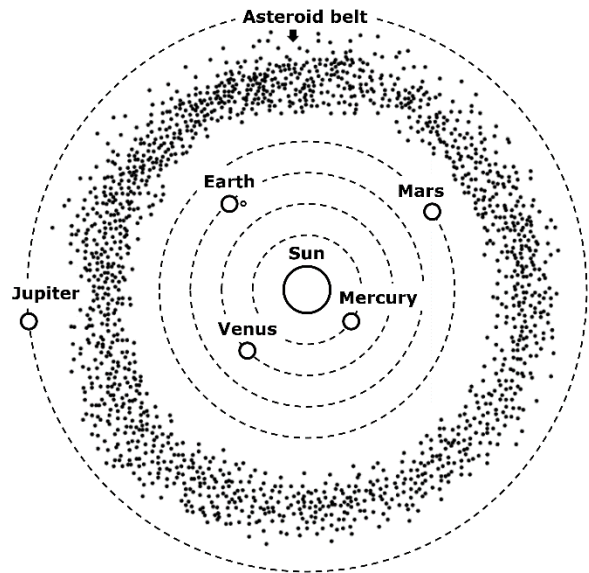


Figure 1. The asteroid belt is located between the orbits of the Jupiter and Mars (**not** to scale).

Rules of Destroying Asteroid

- The celestial body can collide with the asteroids in any order.
- In a collision, if the mass of the celestial body is greater than the mass of the asteroid, then the asteroid will be destroyed (succeeded) and the celestial body will gain the mass of the asteroid.
- In a collision, if the mass of the celestial body is smaller than the mass of the asteroid, then the celestial body will be destroyed (failed).
- In a collision, if the mass of the celestial body is equal to the mass of the asteroid, then both of the celestial body and the asteroid will be destroyed (failed).

The Input File

- The input file is a **plain text file** (filename: **masses.txt**).
- A list of positive integers is stored in the input file (separated by whitespaces).
- The integers are the masses of the asteroids the celestial body is going to destroy.
- There may be empty lines at the beginning, in the middle, and/or at the end of the input file, your program should smartly skip those empty lines.
- You **cannot** assume the number of masses stored in the input file. In other words, no matter how many asteroids are stored in the input file, your program should correct process all of them.
- Please refer to the **sample input files** (on Canvas) to better understand the input file format.

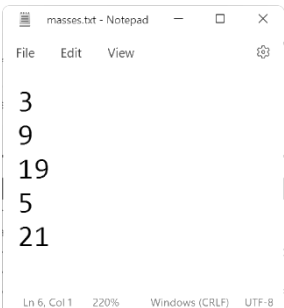
The Output File

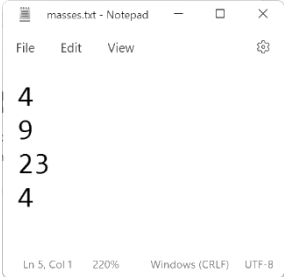
- The output file is a **plain text file** (filename: **result.txt**).
- Your program will write a single integer to the output file, which is the minimum mass of the celestial body required to destroy all the asteroids.
- Please refer to the **sample output files** (on Canvas) to better understand the output file format.

Other Development Notes

- You need to use a vector to store the masses read from the input file.
- (Hint) **Sorting** will be very helpful to solve the problem in this project.
- In this project, you can use either index or iterator to iterate through the vector.

Examples

Sample Input File	Result	Explanation
	4	<ul style="list-style-type: none">▪ One way to order the asteroids is [3, 5, 9, 19, 21].▪ Collide with the asteroid with a mass of 3. New mass: 7▪ Collide with the asteroid with a mass of 5. New mass: 12▪ Collide with the asteroid with a mass of 9. New mass: 21▪ Collide with the asteroid with a mass of 19. New mass: 40▪ Collide with the asteroid with a mass of 21. New mass: 61▪ All asteroids are destroyed and 4 is the minimum mass.

Sample Input File	Result	Explanation
	7	<ul style="list-style-type: none"> ▪ One way to order the asteroids is [4, 9, 4, 23]. ▪ Collide with the asteroid with a mass of 4. New mass: 11 ▪ Collide with the asteroid with a mass of 9. New mass: 20 ▪ Collide with the asteroid with a mass of 4. New mass: 24 ▪ Collide with the asteroid with a mass of 23. New mass: 47 ▪ All asteroids are destroyed and 7 is the minimum mass.

Sample Input and Output Files

Please see and download the sample input and output files on Canvas.

Project Submission

- Source code (any **.h** and **.cpp** files) should be submitted on Canvas.
- Submission details are available on Canvas.

Deadline: Thursday, February 23, by 07:59 AM (Code submitted on Canvas)