Team Assignment: Defend your code!

Part 1

Write a program in Java that does the following. Be sure and make clear to the user what is expected:

- prompts for and reads the user's first name, then last name -- each should be at most 50 characters -- decide what is legitimate for characters for first and last name
    - You must make it clear to the user what is expected for input (describe range, acceptable characters, and anything else you feel is important)
- prompts for and reads in two int values from the user (range of values are those of a 4 byte int)
    - Once again, make clear to user what is expected. Note that an int can be from -2,147,...,... to 2,147,...,...
- prompts for reads the name of an input file from the user
    - Make clear to user what is expected and will be accepted
- prompts for reads the name of an output file from the user
    - Same here
- prompts the user to enter a password, store the password, then ask the user to re-enter the password and verify that it is correct
    - password should be hashed using a salt and written to a file
    - to validate, grab hash from file and compare it to hash from second user entry for password
    - as long as passwords don't match/follow your password specifications, re-prompt the user
- opens the output file and
    - writes the user's name
    - writes the result of adding the two integer values (no overflow should occur)
    - writes the result of multiplying the two integer values (no overflow should occur),
    - writes the contents of the input file
    - Each thing written should be clearly labeled (e.g. First name, Last name, First Integer, Second Integer, Sum, Product, Input File Name, Input file contents)
    - NOTE: it is ok to echo output to the screen as you wish

Your program should do whatever you feel is necessary to ensure the above information is **properly** **obtained** (meaning you must get valid input from the user) and subsequently written to the output file -*without error*. Any error that does arise should be reported to an error log file.

Your program should keep running until it gets proper input from the user. You must mandate proper input is entered and must not allow your program to crash.

Part 2

Now do the same thing in Python (or a language of your team's choice if you feel Python is too much)!

NOTES

1. Make clear in your program output for parts 1 and 2 what you expect the user to enter (e.g. "you must entire filenames that are text files (end in .txt)")
   A. It should be clear to your instructor (or anyone else running your program) what to enter and what restrictions are in place on what to enter. This should be done for all requested input.
2. Document any shortcomings you are able to identify with your program (in required README.TXT - see below)
3. Clearly document what things you think you defended against in your code
4. Include your team name and member names at the top of your code
5. Make sure your code is clean and easy to read.
6. Your nefarious instructor will try and break your code / crash your program. Make sure he cannot do this!

TO TURN IN

Turn in a zip named with team member last names that contains:

- all your source code
  - one folder for Java solution
  - one folder for Python (or other language) solution
- output captures from testing each of your solutions (one output capture should be for Java, a second capture should be for Python - place those captures in the folders with the associated source code).
  - NOTE: Make sure your testing is thorough, you will be graded on this.
    - testing should include edge cases (mins and maxes), improper input, as well as general cases
  - captures should clearly show what was tested - annotate your captures as necessary for clarity
- Include a README.txt that includes team member names, shortcomings, any compilation instructions your code requires, and anything else you deem necessary.

Rubric

Defend Your Code Rubric

Criteria	Ratings	Pts

This criterion is linked to a Learning Outcome Output captures Captures show robust testing to ensure code works correctly	20 pts
This criterion is linked to a Learning Outcome First and Last name Properly reads in first and last name. Program makes clear what range and what characters are acceptable	10 pts
This criterion is linked to a Learning Outcome Integer values Obtains two integers in the range of a 4 byte integer and performs math as specified with no overflow. Program should make clear to user what is expected.	15 pts
This criterion is linked to a Learning Outcome Password mechanics Asks for a password and includes what is expected in terms of length and characters for that password. Password is salted and hashed in proper cryptographic fashion and save to a text file. Password is then verified by asking user to enter it again. Code makes sure a proper password is obtained before continuing	20 pts
This criterion is linked to a Learning Outcome Input and Output file Properly obtains an input and output file name, opens input file, opens output file, writes all required items to output file. Closes files when done.	20 pts
This criterion is linked to a Learning Outcome Errors/problems written to a log file	5 pts
This criterion is linked to a Learning Outcome Java specific issues	10 pts
This criterion is linked to a Learning Outcome Python specific issues	10 pts
This criterion is linked to a Learning Outcome Program ensures all input is valid Program assures all input from user is entered properly before moving to next input.	10 pts