Secure Coding (CNT 4419)
Assignment II

Objective: To become acquainted with, and use, a C safe-math library.
Due Date: Monday, November 9, 2026 at 3pm. No late submissions will be accepted.

Assignment Description
Complete this assignment by yourself. While doing this assignment you will need to run a C compiler. One option is to use an online C compiler such as https://paiza.io/en/languages/online-c-compiler. If you use Paiza, you may optionally create an account to save your code. Paiza has an Input tab for providing input to the program.

This assignment asks you to use the Safe Math library. An example program (*Main.c*) using this library can be found at https://paiza.io/projects/hqKdfhnoRkdSNH2GTpOOSA. This example includes sample input in the Input tab. You may fork this code (that is, add a copy to your account) using the menu next to the Run button.

If you are not using Paiza, begin by downloading the *safe-math.h* file from the repository at https://github.com/nemequ/portable-snippets/tree/master/safe-math. Then copy-paste the code shown in *Main.c* at the Paiza link above into your own *Main.c* file.

As part of the Safe Math library, you will be using the `psnip_safe_char_mul`, `psnip_safe_int_mul`, and `psnip_safe_long_mul` functions, which are declared and implemented in the *safe-math.h* file.

Below is brief documentation for the library functions you will need to use. This documentation appeared in the project's *readme* in GitHub. Note: `psnip_safe_bool`s can be used as regular bools.

- **psnip_safe_bool psnip_safe_char_mul (char* res, char a, char b)**
  Attempts to multiply chars `a` and `b`, and stores the results in the address `res`. If the operation can be completed without overflowing or underflowing, a `psnip_safe_bool` value is returned that evaluates to true. Otherwise, a `psnip_safe_bool` that evaluates to false is returned.

- **psnip_safe_bool psnip_safe_int_mul (int* res, int a, int b)**
  Attempts to multiply integers `a` and `b`, and stores the results in the address `res`. If the operation can be completed without overflowing or underflowing, a `psnip_safe_bool` value is returned that evaluates to true. Otherwise, a `psnip_safe_bool` that evaluates to false is returned.

- **psnip_safe_bool psnip_safe_long_mul (long* res, long a, long b)**
  Attempts to multiply longs `a` and `b`, and stores the results in the address `res`. If the operation can be completed without overflowing or underflowing, a `psnip_safe_bool` value is returned that evaluates to true. Otherwise, a `psnip_safe_bool` that evaluates to false is returned.

Use these functions to implement a program that inputs two strings S1 and S2 from stdin, converts S1 and S2 to longs (set to 1 if the conversion fails), multiplies S1 and S2 using the smallest single type S, and outputs the result. The eligible types S are char, int, and long, where char is "smaller" than int, and int is "smaller" than long. The two possible program outputs are of the form:

1. "`S1 and S2 can be multiplied as type S, yielding S3.`" (where `S` is the smallest single type that is large enough for storing S1, S2, and S3)
2. "`S1 and S2 cannot be multiplied as longs.`"

Substitute the actual values for S1, S2, S3, and S in these outputs.

You will want to use the constants `CHAR_MIN`, `CHAR_MAX`, `INT_MIN`, `INT_MAX`, `LONG_MIN`, and `LONG_MAX`, which are found in the standard C-header file *limits.h*. For this assignment, you should only need the *Main.c* and *safe-math.h* files, with your *Main.c* including only the *safe-math.h* and standard-library headers (e.g., *stdio.h*, *limits.h*, *errno.h*, and *stdlib.h*).

Submit to Canvas your completed *Main.c* file.

Notes:

- For full credit, your code must compile and execute without warnings or errors (on Paiza).
- Your code will be graded in part on readability, simplicity, and efficiency.