1. Ex. 1: Add to the INTEXP interpreter the integer numbers (-1, -2, . . .) and the following operators:

   - - (subtraction), e.g., (-7 - -2),
   - / (integer division), e.g., (-9 / 3),
   - mod (remainder of an integer division), e.g., (-7 mod -2)

2. Ex. 2: Extend again the INTEXP interpreter to allow the use of identifiers (like you do in mathematical expressions). For instance,

   ```
   (a * (b + 5))
   ```

   is a mathematical expression with two identifiers a and b. We allow the user to employ all the identifiers in the set $\{a, b, \ldots, z\}^+$, i.e., all the lowercase strings over the english alphabet.

   A program in this new language consists of:

   - a finite sequence of identifier initializations, e.g., a = (5 + 4); b = 3; c = ((5 * a) + 5); ...;
   - a single mathematical expression with identifiers.

   For instance, the following is a valid program:

   ```
   base = 5;
   height = (base * 2);

   (base * height)
   ```

   We assume that all the used but non-initialized identifiers evaluate to 0. The output of the program is the value of the final expression.