Consider a variant of ARITH where integers can be between 0 and $2^{64}-1$ only, and the arithmetic operations yield an "overflow" error rather than results larger than $2^{64}-1$ (and otherwise behave as usual).

1. Revise the abstract syntax of the language.
   Note that the "overflow" error is not an expression of the language.

2. Revise the big-step operational semantics, using judgments of the form $e \Downarrow v$ where $e$ is an expression and $v$ is either an integer $n$ or the new result OVERFLOW.

3. Write an Objective Caml type for the expressions in this language, and a function that, given an expression $e$, returns $v$ such that $e \Downarrow v$.
   (Please try a few sample runs and turn in the results.)
   **Note:** It is acceptable that you write the code using a different bound on integers, e.g., 9 rather than $2^{64}-1$.

4. Write a contextual semantics for this variant of ARITH.
   In other words, define redexes, local reduction rules, contexts, and global reduction rules.

5. Please turn in **on paper** at start of class Thursday October 7.