You are developing a new App where you can record the movement of a ball in a real-life shuffle board game and then convert it into movements of a blinking ball in a grid on the screen. Unfortunately the trajectory-capturing software records the movement of the real ball into a binary representation and the image-rendering library that you are using was written in JavaScript and expects a string of word commands. An important mobile device in the target market is based on a RISC-V microcontroller that uses RISC-V assembly, but there is no compiler available for this controller yet. Thus you will have to write RISC-V assembly routines to do this conversion. To make the task easier, you have divided the functionality that you need into two separate routines.

Question 4 (20 points): In this question you will write `concatenate`, a subroutine that receives two pointers to null-terminated strings: `a0` receives the `_to` pointer, which is the address of the string that will be grown by the concatenation; `a1` receives the `_from` pointer, which is the address of the string that is to be concatenated into the string pointed by `_to`. Figure 1 illustrates the effect of one call to the subroutine concatenate. You can assume that there is enough memory allocated for the `_to` string to enable the concatenation of the `_from` string without overwriting to other data structures in the program. You must follow all the subroutine invocation conventions of RISC-V.

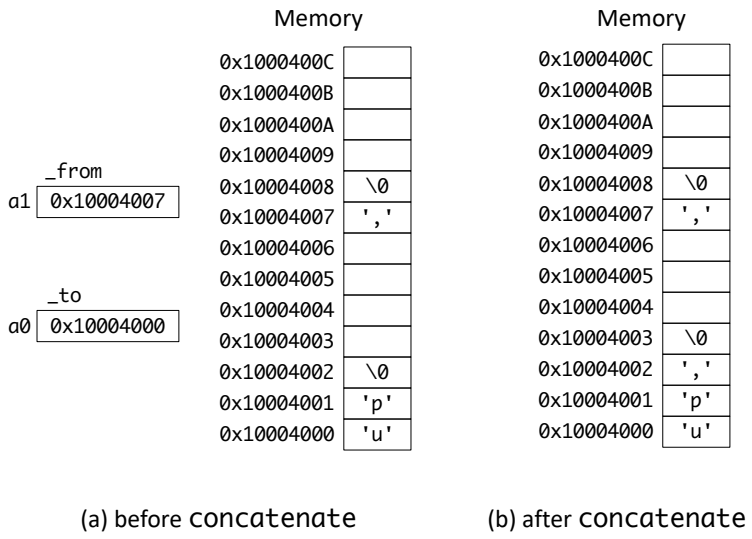

Figure 1: Example of execution of a concatenate function.

Code for `concatenate`