CS212 – Lab 3 – Implementing a stack with an array

A. Implement the following functions for manipulating a stack (without actually declaring the stack – you will do that later), based on an array of characters of size STACKSZ, where STACKSZ is a "*const int*" or a #define symbol with a value of 20 (for now). The name of the stack is your choice. Pick a name. It will be declared in step B.
    a. Push – adds one element to the top of the stack (accepts a character as input), returns an int:
        i. -1 if the stack is already full,
        ii. 0 if the item was successfully added
    b. Pop – removes the top element (if it exists) and returns:
        i. NULL if there are no elements (stack is empty)
        ii. The element itself if there was one
    c. Top – does NOT remove anything. Returns a char with the value:
        i. NULL if there are no elements (stack is empty)
        ii. The topmost element if there is one
    d. isFull – does NOT remove anything. Returns an int:
        i. 0 if there are no elements (the stack is empty)
        ii. 1 if the stack is full
        iii. -1 in all other cases
B. Using the above functions, write a program that:
    a. Declares the stack with a size of STACKSZ characters
    b. Uses your functions above to do the following:
        i. Accept input characters, 1 at a time until the symbol ^ is input
        ii. Push each character onto the stack
        iii. Output the stack (top to bottom) after all inputs have been received

Notes:

1. Your main program should have NO access to the stack except through the above functions. The stack array & value of the index to the top ARE global, so all your functions can get at them, but the main code must never look at or use them.
2. If the stack is full and more data keeps coming, your program should ignore the data and reply, "The stack is full. Enter the '^' character to stop.".
3. Repeat the above message if more input (other than "^" keeps coming).