Pesky Pointers: There is a bug in the code.

To understand how changing len++ to ++len fixed the issue, we must first understand the behavior of prefix and postfix arithmetic operators.

- The *pre*-increment operator increment their operand by 1, and the value of the expression is the resulting incremented value.
- The *post*-increment operator increase the value of their operand by 1, but the value of the expression is the operand's value *prior* to the increment operation.

In order to compute where NULL character is '\0' ASCII 0 for array src, we want the value of integer len to be 5 since '\0' is at src[5] (formation of array src is {'c','s','2','3','!','\0'}). By executing while(src[len++]), the following iterations will occur

1. while(src[0]); // true, len is now 1
2. while(src[1]); // true, len is now 2
3. while(src[2]); // true, len is now 3
4. while(src[3]); // true, len is now 4
5. while(src[4]); // true, len is now 5
6. while(src[5]); // false, len is now 6

As we can see here, len keeps increasing even though the last iteration turns out to be false. Integer len is now 6 and not 5 as what we desired, that is the bug we are going to solve. Such that integer len will stop incrementing once it has reached src[5], we must increment the value in advance. By executing while(src[++len]), the following iterations will occur

1. while(src[1]); // true, len is now 1
2. while(src[2]); // true, len is now 2
3. while(src[3]); // true, len is now 3
4. while(src[4]); // true, len is now 4
5. while(src[5]); // false, len is now 5

Integer len now correctly represents 5, thus fixing the bug.