**2. Str\_concat** Procedure Write a procedure named Str\_concat that concatenates a source string to the end of a target string. Sufficient space must exist in the target string to accommodate the new characters. Pass pointers to the source and target strings. Here is a sample call:

.data

targetStr BYTE "ABCDE",10 DUP(0)

sourceStr BYTE "FGH",0

.code

INVOKE Str\_concat, ADDR targetStr, ADDR sourceStr

**3. Str\_remove** Procedure Write a procedure named Str\_remove that removes n characters from a string. Pass a pointer to the position in the string where the characters are to be removed. Pass an integer specifying the number of characters to remove. The following code, for example, shows how to remove “xxxx” from target:

.data

target BYTE "abcxxxxdefghijklmop",0

.code

INVOKE Str\_remove, ADDR [target+3], 4

**5. Str\_nextWord** Procedure Write a procedure called Str\_nextWord that scans a string for the first occurrence of a certain delimiter character and replaces the delimiter with a null byte. There are two input parameters: a pointer to the string and the delimiter character. After the call, if the delimiter was found, the Zero flag is set and EAX contains the offset of the next character beyond the delimiter. Otherwise, the Zero flag is clear and EAX is undefined. The following example code passes the address of target and a comma as the delimiter:

.data

target BYTE "Johnson,Calvin",0

.code

INVOKE Str\_nextWord, ADDR target, ',' jnz notFound

In Figure 9-5, after calling Str\_nextWord, EAX points to the character following the position where the comma was found (and replaced).

