```c
#include <stdio.h>

#define MAX_STRING_LENGTH 512

int main(void)
{
    //function prototype
    void MyStrrev(char[], char[]);

    //variable declarations
    char chArray_Original[MAX_STRING_LENGTH], chArray_Reversed
      [MAX_STRING_LENGTH]; // A Character Array Is A String

    //code

    // *** STRING INPUT ***
    printf("\n\n");
    printf("Enter A String : \n\n");
    gets_s(chArray_Original, MAX_STRING_LENGTH);

    // *** STRING REVERSE ***
    MyStrrev(chArray_Reversed, chArray_Original);

    // *** STRING OUTPUT ***
    printf("\n\n");
    printf("The Original String Entered By You (i.e : 'chArray_Original[]') Is :
      \n\n");
    printf("%s\n", chArray_Original);

    printf("\n\n");
    printf("The Reversed String (i.e : 'chArray_Reversed[]') Is : \n\n");
    printf("%s\n", chArray_Reversed);

    return(0);
}

void MyStrrev(char str_destination[], char str_source[])
{
    //function prototype
    int MyStrlen(char[]);

    //variable declarations
    int iStringLength = 0;
    int i, j, len;

    //code
    iStringLength = MyStrlen(str_source);

    // ARRAY INDICES BEGIN FROM 0, HENCE, LAST INDEX WILL ALWAYS BE (LENGTH - 1)
    len = iStringLength - 1;

    // WE NEED TO PUT THE CHARACTER WHICH IS AT LAST INDEX OF 'str_source' TO THE
```

```
        FIRST INDEX OF 'str_destination'
51      // AND SECOND-LAST CHARACTER OF 'str_source' TO THE SECOND CHARACTER OF
          'str_destination' and so on...
52      for (i = 0, j = len; i < iStringLength, j >= 0; i++, j--)
53      {
54          str_destination[i] = str_source[j];
55      }
56
57      str_destination[i] = '\0';
58  }
59
60  int MyStrlen(char str[])
61  {
62      //variable declarations
63      int j;
64      int string_length = 0;
65
66      //code
67      // *** DETERMINING EXACT LENGTH OF THE STRING, BY DETECTING THE FIRST
          OCCURENCE OF NULL-TERMINATING CHARACTER ( \0 ) ***
68      for (j = 0; j < MAX_STRING_LENGTH; j++)
69      {
70          if (str[j] == '\0')
71              break;
72          else
73              string_length++;
74      }
75      return(string_length);
76  }
77
```