```c
#include <stdio.h>

#define MAX_STRING_LENGTH 512

int main(void)
{
    //function prototype
    void MyStrcat(char *, char *);
    int MyStrlen(char *);

    //variable declarations
    char *chArray_One = NULL, *chArray_Two = NULL; // A Character Array Is A String

    //code

    // *** STRING INPUT ***
    printf("\n\n");
    chArray_One = (char *)malloc(MAX_STRING_LENGTH * sizeof(char));
    if (chArray_One == NULL)
    {
        printf("MEMORY ALLOCATION TO FIRST STRING FAILED !!! EXITTING NOW...\n\n");
        exit(0);
    }

    printf("Enter First String : \n\n");
    gets_s(chArray_One, MAX_STRING_LENGTH);

    printf("\n\n");
    chArray_Two = (char *)malloc(MAX_STRING_LENGTH * sizeof(char));
    if (chArray_Two == NULL)
    {
        printf("MEMORY ALLOCATION TO SEOND STRING FAILED !!! EXITTING NOW...\n\n");
        exit(0);
    }

    printf("Enter Second String : \n\n");
    gets_s(chArray_Two, MAX_STRING_LENGTH);

    // *** STRING CONCAT ***
    printf("\n\n");
    printf("****** BEFORE CONCATENATION ******");
    printf("\n\n");
    printf("The Original First String Entered By You (i.e : 'chArray_One[]') Is : \n\n");
    printf("%s\n", chArray_One);

    printf("\n\n");
    printf("The Original Second String Entered By You (i.e : 'chArray_Two[]') Is : \n\n");
    printf("%s\n", chArray_Two);

    MyStrcat(chArray_One, chArray_Two);

```

```
52      printf("\n\n");
53      printf("****** AFTER CONCATENATION ******");
54      printf("\n\n");
55      printf("'chArray_One[]' Is : \n\n");
56      printf("%s\n", chArray_One);
57
58      printf("\n\n");
59      printf("'chArray_Two[]' Is : \n\n");
60      printf("%s\n", chArray_Two);
61
62      if (chArray_Two)
63      {
64          free(chArray_Two);
65          chArray_Two = NULL;
66          printf("\n\n");
67          printf("MEMORY ALLOCATED TO SECOND STRING HAS BEEN SUCCESSFULLY
              FREED !!!\n\n");
68      }
69
70      if (chArray_One)
71      {
72          free(chArray_One);
73          chArray_One = NULL;
74          printf("\n\n");
75          printf("MEMORY ALLOCATED TO FIRST STRING HAS BEEN SUCCESSFULLY
              FREED !!!\n\n");
76      }
77
78      return(0);
79  }
80
81  void MyStrcat(char *str_destination, char *str_source)
82  {
83      //function prototype
84      int MyStrlen(char *);
85
86      //variable declarations
87      int iStringLength_Source = 0, iStringLength_Destination = 0;
88      int i, j;
89
90      //code
91      iStringLength_Source = MyStrlen(str_source);
92      iStringLength_Destination = MyStrlen(str_destination);
93
94      // ARRAY INDICES BEGIN FROM 0, HENCE, LAST VALID INDEX OF ARRAY WILL
          ALWAYS BE (LENGTH - 1)
95      // SO, CONCATENATION MUST BEGIN FROM INDEX NUMBER EQUAL TO LENGTH OF THE
          ARRAY 'str_destination'
96      // WE NEED TO PUT THE CHARACTER WHICH IS AT FIRST INDEX OF 'str_source' TO
           THE (LAST INDEX + 1) OF 'str_destination'
97      for (i = iStringLength_Destination, j = 0; j < iStringLength_Source; i++,
          j++)
98      {
99          *(str_destination + i) = *(str_source + j);
100     }
101
```

```
102     *(str_destination + i) = '\0';
103 }
104
105 int MyStrlen(char *str)
106 {
107     //variable declarations
108     int j;
109     int string_length = 0;
110
111     //code
112     // *** DETERMINING EXACT LENGTH OF THE STRING, BY DETECTING THE FIRST
        OCCURENCE OF NULL-TERMINATING CHARACTER ( \0 ) ***
113     for (j = 0; j < MAX_STRING_LENGTH; j++)
114     {
115         if (str[j] == '\0')
116             break;
117         else
118             string_length++;
119     }
120     return(string_length);
121 }
122
```