```
#include <stdio.h>
 2
   #include <stdlib.h>
 3
 4 #define MAX STRING LENGTH 512
 5
 6 int main(void)
 7
 8
        //function declarations
 9
        char* ReplaceVowelsWithHashSymbol(char *);
10
        //variable declaration
11
        char string[MAX STRING LENGTH];
12
13
        char *replaced string = NULL;
14
       //code
15
        printf("\n\n");
16
        printf("Enter String : ");
17
18
        gets_s(string, MAX_STRING_LENGTH);
19
20
        replaced string = ReplaceVowelsWithHashSymbol(string);
21
        if (replaced_string == NULL)
22
        {
            printf("ReplaceVowelsWithHashSymbol() Function Has Failed !!! Exitiing >
23
               Now...\n\n");
24
            exit(0);
25
        }
26
        printf("\n\n");
27
28
        printf("Replaced String Is : \n\n");
29
        printf("%s\n\n", replaced_string);
30
31
        if (replaced string)
32
       {
33
            free(replaced string);
34
            replaced_string = NULL;
35
        }
36
37
        return(0);
38 }
39
40 char* ReplaceVowelsWithHashSymbol(char *s)
41 {
42
        //function prototype
43
        void MyStrcpy(char *, char *);
        int MyStrlen(char *);
44
45
46
        //varibale declarations
47
        char *new_string = NULL;
48
        int i;
49
50
        //code
        new string = (char *)malloc(MyStrlen(s) * sizeof(char));
51
52
        if (new string == NULL)
53
            printf("COULD NOT ALLOCATE MEMORY FOR NEW STRING !!!\n\n");
54
55
            return(NULL);
```

```
... unctions \verb|\| 02-Pointer As Return Value \verb|\| Pointer As Return Value.c
```

```
2
```

```
56
 57
 58
         MyStrcpy(new_string, s);
 59
         for (i = 0; i < MyStrlen(new_string); i++)</pre>
 60
 61
             switch (new_string[i])
 62
             case 'A':
 63
             case 'a':
 64
 65
             case 'E':
             case 'e':
 66
             case 'I':
 67
             case 'i':
 68
             case '0':
 69
             case 'o':
 70
             case 'U':
 71
 72
             case 'u':
 73
                 new_string[i] = '#';
 74
                 break;
 75
             default:
 76
                 break;
 77
             }
         }
 78
 79
 80
         return(new_string);
 81 }
 82
 83 void MyStrcpy(char *str_destination, char *str_source)
 84 {
         //function prototype
 85
 86
         int MyStrlen(char *);
 87
 88
         //variable declarations
 89
         int iStringLength = 0;
 90
         int j;
 91
 92
         //code
 93
         iStringLength = MyStrlen(str_source);
 94
         for (j = 0; j < iStringLength; j++)</pre>
 95
             *(str_destination + j) = *(str_source + j);
 96
 97
         *(str_destination + j) = '\0';
 98 }
 99
100 int MyStrlen(char *str)
101 {
102
         //variable declarations
103
         int j;
104
         int string_length = 0;
105
106
         //code
         // *** DETERMINING EXACT LENGTH OF THE STRING, BY DETECTING THE FIRST
107
           OCCURENCE OF NULL-TERMINATING CHARACTER ( \0 ) ***
108
         for (j = 0; j < MAX_STRING_LENGTH; j++)</pre>
109
         {
             if (str[j] == '\0')
110
```

```
...unctions\02-PointerAsReturnValue\PointerAsReturnValue.c
111 break;
```