

```

1  #include <stdio.h>
2
3  // DEFINING STRUCT
4  struct MyData
5  {
6      int i;
7      float f;
8      double d;
9  };
10
11 int main(void)
12 {
13     //function prototypes
14     void ChangeValues(struct MyData *);
15
16     //variable declarations
17     struct MyData *pData = NULL;
18
19     //code
20     printf("\n\n");
21
22     pData = (struct MyData *)malloc(sizeof(struct MyData));
23     if (pData == NULL)
24     {
25         printf("FAILED TO ALLOCATE MEMORY TO 'stunct MyData' !!! EXITTING NOW ... ↗
26             \n\n");
27         exit(0);
28     }
29     else
30         printf("SUCCESSFULLY ALLOCATED MEMORY TO 'stunct MyData' !!!\n\n");
31
32     //Assigning Data Values To The Data Members Of 'struct MyData'
33     pData->i = 30;
34     pData->f = 11.45f;
35     pData->d = 1.2995;
36
37     //Displaying Values Of The Data Members Of 'struct MyData'
38     printf("\n\n");
39     printf("DATA MEMBERS OF 'struct MyData' ARE : \n\n");
40     printf("i = %d\n", pData->i);
41     printf("f = %f\n", pData->f);
42     printf("d = %lf\n", pData->d);
43
44     ChangeValues(pData);
45
46     //Displaying Values Of The Data Members Of 'struct MyData'
47     printf("\n\n");
48     printf("DATA MEMBERS OF 'struct MyData' ARE : \n\n");
49     printf("i = %d\n", pData->i);
50     printf("f = %f\n", pData->f);
51     printf("d = %lf\n", pData->d);

```

```
52
53     if (pData)
54     {
55         free(pData);
56         pData = NULL;
57         printf("MEMORY ALLOCATED TO 'struct MyData' HAS BEEN SUCCESSFULLY FREED !!!\n\n");
58     }
59
60     return(0);
61 }
62
63 void ChangeValues(struct MyData *pParam_Data)
64 {
65     //code
66
67     pParam_Data->i = 9;
68     pParam_Data->f = 8.2f;
69     pParam_Data->d = 6.1998;
70
71     // CAN ALSO DO THIS AS ...
72     /*
73     (*pParam_Data).i = 9;
74     (*pParam_Data).f = 8.2f;
75     (*pParam_Data).d = 6.1998;
76     */
77 }
78
79
80
```