Mandatory

1. Refer the program file “memory_access_error.c”, read the description not and fix the reported error in this program

user72@trainux01: ~/Assignments

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_MODULE_LENGTH   10
#define FAILURE   1
#define SUCCESS   0
#define MAX_MODULE_NAME     20
#define FREE(x) \
    if (x)    \
    {       \
        free(x);    \
        x = NULL;    \
    }
typedef struct config
{
    char *module;
    int portcount;
} CONFIG;
int setconfig(CONFIG *cfg, int portcnt, char *mname)
{
    int ret = FAILURE;

    cfg->module = (char*)malloc(sizeof(char) * (strlen(mname) + 1));
    if (cfg->module)
    {
        strcpy(cfg->module, mname);
        cfg->portcount = portcnt;

        printf("\nsetconfig:  module:%s, portcount:%d", cfg->module, cfg->portcount);
        ret = SUCCESS;
    }
    return ret;
}
void dump_hex(char *buf, size_t sz)
{
    size_t i = 0;
    char *ptr = buf;

    printf("\nDump_hex:\n ");
    for (i = 0; i < sz; i++)
    {
        printf("%x ", *(ptr + i));
    }
    printf("\nEnd of Data, sz:%d, i:%d", (int)sz, (int)i);
}
int get_portcount(void *cfg)
{
```

user72@trainux01: ~/Assignments

```
45      printf("\nEnd of Data, sz:%d, i:%d", (int)sz, (int)i);
46  }
47  int get_portcount(void *cfg)
48  {
49      CONFIG *cfgp = (CONFIG*)cfg;
50      int portcnt = cfgp->portcount;
51
52      printf("\nget_portcount:port:%d", portcnt);
53
54      return portcnt;
55  }
56  char *get_module(void *cfg)
57  {
58      CONFIG *cfgp = (CONFIG*)cfg;
59      char *module = cfgp->module;
60
61      if (module)
62      {
63          printf("\nget_module:%s", module);
64      }
65      return module;
66  }
67  void displ(void *cfg)
68  {
69      int portcnt = get_portcount(cfg);
70      char *mod = get_module(cfg);
71
72      printf("\nReading config: %s %d", mod, portcnt);
73  }
74  int main()
75  {
76      CONFIG *cfg = NULL;
77      int ret = FAILURE;
78      cfg = malloc(sizeof(CONFIG));
79      if (cfg == NULL)
80      {
81          printf("\nMemory allocation failed");
82          exit(FAILURE);
83      }
84      memset(cfg, 0, sizeof(CONFIG));
85      printf("\nTo Set the config module module:DNS, portcount:20");
86      ret = setconfig(cfg, 20, "DNS");
87      displ(cfg);
88      FREE(cfg->module);
89      FREE(cfg);
90
91      return SUCCESS;
92  }
user72@trainux01:~/Assignments$ vi padding.c
user72@trainux01:~/Assignments$ gcc padding.c
user72@trainux01:~/Assignments$ ./a.out

To Set the config module module:DNS, portcount:20
setconfig:  module:DNS, portcount:20
get_portcount:port:20
get_module:DNS
Reading config: DNS 20user72@trainux01:~/Assignments$ vi padding.c
```