```c
 1  #include <stdio.h>
 2
 3  #define NUM_ROWS 5
 4  #define NUM_COLUMNS 3
 5
 6  int main(void)
 7  {
 8      //variable declarations
 9      int iArray[NUM_ROWS][NUM_COLUMNS];
10      int i, j;
11
12      //code
13      // *** NAME OF AN ARRAY ITSELF IS ITS BASE ADDRESS ***
14      // *** HENCE, 'iArray' IS BASE ADDRESS OF 2D INTEGER ARRAY iArray[][]
15
16      // iArray[5][3] => iArray IS A 2D ARRAY HAVING 5 ROWS AND 3 COLUMNS. EACH
            OF THESE 5 ROWS IS A 1D INTGER ARRAY OF 3 INTEGERS ***
17      // iArray[0] => IS THE 0TH ROW ... HENCE, IS THE BASE ADDRESS OF ROW 0
18      // iArray[1] => IS THE 1ST ROW ... HENCE, IS THE BASE ADDRESS OF ROW 1
19      // iArray[4] => IS THE 4TH ROW ... HENCE, IS THE BASE ADDRESS OF ROW 4
20
21      // (iArray[0] + 0) => ADDRESS OF 0th INTEGER FROM BASE ADDRESS OF 0th ROW
            (iArray[0][0])
22      // (iArray[0] + 1) => ADDRESS OF 1ST INTEGER FROM BASE ADDRESS OF 0th ROW
            (iArray[0][1])
23      // (iArray[0] + 2) => ADDRESS OF 2ND INTEGER FROM BASE ADDRESS OF 0th ROW
            (iArray[0][2])
24
25      // (iArray[1] + 0) => ADDRESS OF 0th INTEGER FROM BASE ADDRESS OF 1ST ROW
            (iArray[1][0])
26      // (iArray[1] + 1) => ADDRESS OF 1ST INTEGER FROM BASE ADDRESS OF 1ST ROW
            (iArray[1][1])
27      // (iArray[1] + 2) => ADDRESS OF 2ND INTEGER FROM BASE ADDRESS OF 1ST ROW
            (iArray[1][2])
28
29      // iArray[0], iArray[1] ... ARE 1D INTEGR ARRAYS AND HENCE CAN BE TREATED
            AS 1D INTEGER ARRAYS USING POINTERS ...
30      // 'iArray' IS THE NAME AND BASE ADDRESS OF 2D INTEGER ARRAY ***
31      // (*(iArray + 0) + 0) = (iArray[0] + 0) = ADDRESS OF 0TH ELEMENT FROM BASE
            ADDRESS OF 0TH ROW = (iArray[0] + 0) = (iArray[0][0])
32      // (*(iArray + 0) + 1) = (iArray[0] + 1) = ADDRESS OF 1ST ELEMENT FROM BASE
            ADDRESS OF 0TH ROW = (iArray[0] + 1) = (iArray[0][1])
33      // (*(iArray + 0) + 2) = (iArray[0] + 2) = ADDRESS OF 2ND ELEMENT FROM BASE
            ADDRESS OF 0TH ROW = (iArray[0] + 2) = (iArray[0][2])
34
35      // (*(iArray + 1) + 0) = (iArray[1] + 0) = ADDRESS OF 0TH ELEMENT FROM BASE
            ADDRESS OF 1ST ROW = (iArray[1] + 0) = (iArray[1][0])
36      // (*(iArray + 1) + 1) = (iArray[1] + 1) = ADDRESS OF 1ST ELEMENT FROM BASE
            ADDRESS OF 1ST ROW = (iArray[1] + 1) = (iArray[1][1])
37      // (*(iArray + 1) + 2) = (iArray[1] + 2) = ADDRESS OF 2ND ELEMENT FROM BASE
            ADDRESS OF 1ST ROW = (iArray[1] + 2) = (iArray[1][2])
38
39
40      for (i = 0; i < NUM_ROWS; i++)
41      {
42          for (j = 0; j < NUM_COLUMNS; j++)
```

```c
43              *(*(iArray + i) + j) = (i + 1) * (j + 1); // 'iArray[i]' Can Be
                    Treated As 1D Array Using Pointers ...
44          }
45
46      printf("\n\n");
47      printf("2D Integer Array Elements Along With Addresses : \n\n");
48      for (i = 0; i < NUM_ROWS; i++)
49      {
50          for (j = 0; j < NUM_COLUMNS; j++)
51          {
52              printf("*(*(iArray + %d) + %d)= %d \t \t At Address (*(iArray + %d)
                    + %d) : %p\n", i,  j, *(*(iArray + i) + j), i, j, (*(iArray + i)
                    + j));
53          }
54          printf("\n\n");
55      }
56
57      return(0);
58  }
59
60
61
62
63
```