```
runner% cat -n subs5.c
     1  /* Ordinary 2-dimentional arrays */
     2  #include <stdio.h>
     3
     4  void p(int t[][4])
     5  /* or "int t[5][4]", "int (*t)[4]"  */
     6  {
     7          int i, j;
     8          for (i = 0; i < 5; i++)
     9                  switch (i) {
    10                  case 0: for (j = 0; j < 5; j++)  /* j==4 out of bounds*/
    11                                  printf("%4i", t[i][j]);
    12                          printf("\n");
    13                          break;
    14                  case 1: for (j = 0; j < 5; j++)
    15                                  printf("%4i", (*(t+i))[j]);
    16                          printf("\n");
    17                          break;
    18                  case 2: for (j = 0; j < 5; j++)
    19                                  printf("%4i", *(t[i] + j));
    20                          printf("\n");
    21                          break;
    22                  case 3: for (j = 0; j < 5; j++)
    23                                  printf("%4i", *(*(t + i) + j));
    24                          printf("\n");
    25                          break;
    26                  case 4: for (j = 0; j < 5; j++)
    27                                  printf("%4i", *(&t[0][0] + 4*i + j));
    28                          printf("\n");
    29                          break;
    30                  }
    31  }
    32
    33  void q(int t[], int rowsize) /* treat as 1-dimensional array */
    34  {
    35          int i;
    36          for (i = 0; i < 20; i++) {
    37                  printf("%4i", t[i]);
    38                  if (((i+1)%rowsize) == 0) printf("\n");
    39          }
    40  }
    41
    42  void main(void)
    43  {
    44          int s[5][4]  =  /* or "int s[][4];"  */
    45                  {  { 0,  1,  2,  3},
    46                     {10, 11, 12, 13},
    47                     {20, 21, 22, 23},
    48                     {30, 31, 32, 33},
    49                     {40, 41, 42, 43}
    50                  };
    51          p(s);
    52          printf("******************\n");
    53          q(s, 4); /* get compiler and lint warning here */
    54  }
runner% lint -m -u subs5.c
(53) warning: argument is incompatible with prototype: arg #1

function returns value which is always ignored
    printf
runner% cc -o subs5 subs5.c
"subs5.c", line 53: warning: argument is incompatible with prototype: arg #1
runner$ subs5
   0   1   2   3  10
  10  11  12  13  20
  20  21  22  23  30
  30  31  32  33  40
  40  41  42  43   1
```

```
******************
   0   1   2   3
  10  11  12  13
  20  21  22  23
  30  31  32  33
  40  41  42  43

runner% cat subs6.c
/* Array of pointers to char */
#include <stdio.h>

void p(int *t[])
/* or "int **t", "int *t[5]", "int *(t[5])", "int *(t[])"  */
{
        int i, j;
        for (i = 0; i < 5; i++)
                switch (i) {
                case 0: for (j = 0; j < 5; j++)
                                printf("%4i", t[i][j]);
                        printf("\n");
                        break;
                case 1: for (j = 0; j < 5; j++)
                                printf("%4i", (*(t+i))[j]);
                        printf("\n");
                        break;
                case 2: for (j = 0; j < 5; j++)
                                printf("%4i", *(t[i] + j));
                        printf("\n");
                        break;
                case 3: for (j = 0; j < 5; j++)
                                printf("%4i", *(*(t + i) + j));
                        printf("\n");
                        break;
                case 4: for (j = 0; j < 5; j++)
                                printf("%4i", t[i][j]);
                        printf("\n");
                        break;
                        /* Note: *(&t[0][0] + 4*i + j) does not
                            work here */
                }
}

void main(void)
{
        int *s[5];  /* or "int *(s[5]);"  */
        int y[4] = {10, 11, 12, 13};
        int z[4] = {20, 21, 22, 23};
        int x[4] = { 0,  1,  2,  3};
        int v[4] = {40, 41, 42, 43};
        int u[4] = {30, 31, 32, 33};
        s[0] = x; s[1] = y; s[2] = z; s[3] = u; s[4] = v;
        p(s);
}
runner% lint -m -u subs6.c

function returns value which is always ignored
    printf
runner% cc -o subs6 subs6.c
runner% subs6
   0   1   2   3  20
  10  11  12  13-268436252
  20  21  22  23  10
  30  31  32  33  40
  40  41  42  43   0
```