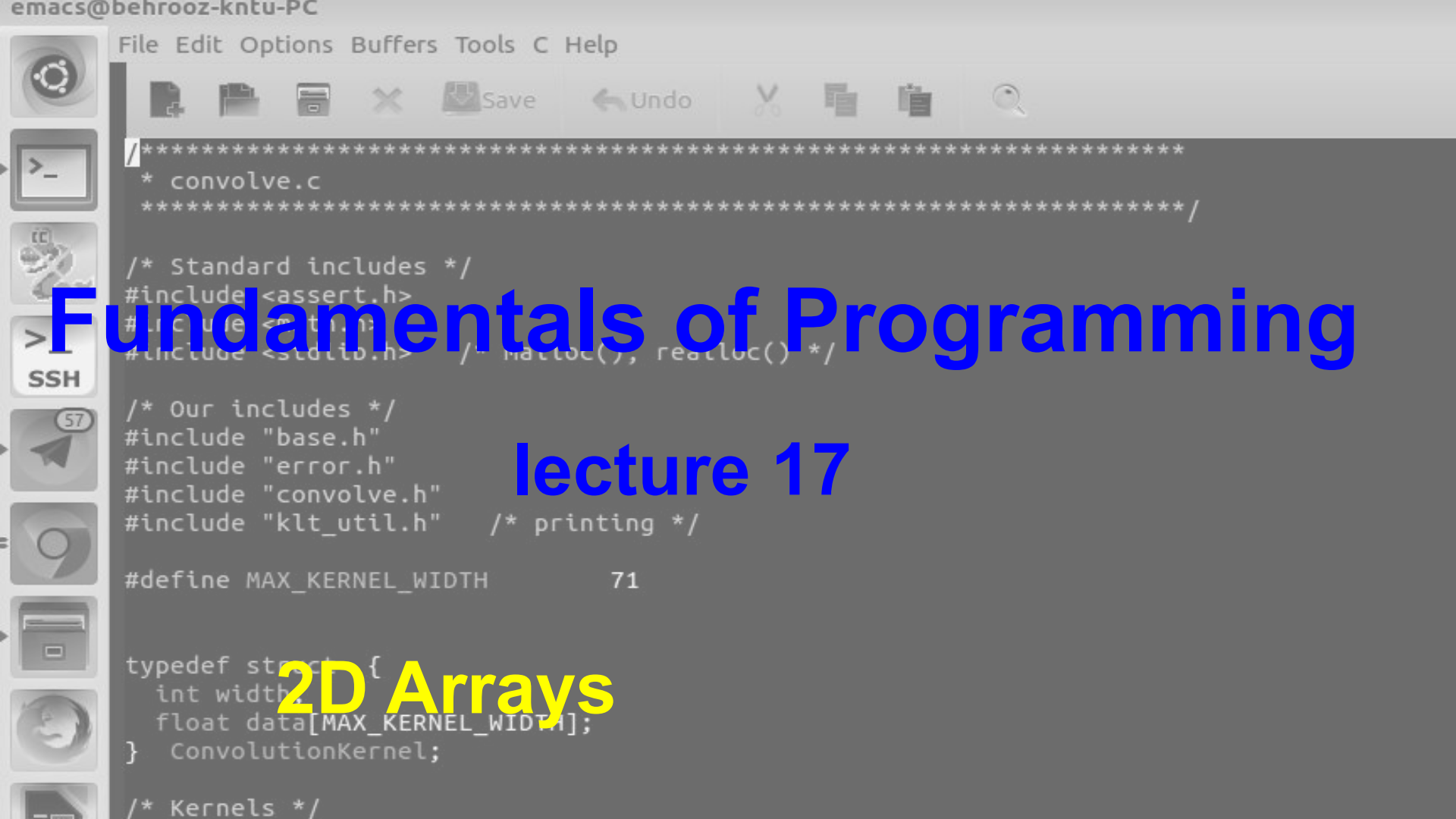

emacs@behrooz-kntu-PC
File Edit Options Buffers Tools C Help
Save
Undo
/**********
* convolve.c
**********/
/* Standard includes */
#include <assert.h>
/* Our includes */
#include "base.h"
#include "error.h"
#include "convolve.h"
#include "klt_util.h" /* printing */
#define MAX_KERNEL_WIDTH 71
typedef struct {
int width;
float data[MAX_KERNEL_WIDTH];
} ConvolutionKernel;
/* Kernels */
SSH
Fundamentals of Programming
lecture 17
2D Arrays

2D arrays

0	0	0	0	0	0
0	1	2	3	4	5
0	2	4	6	8	10
0	3	6	9	12	15
0	4	8	12	16	20
0	5	10	15	20	25

- tabular data
- rows and columns

2D arrays

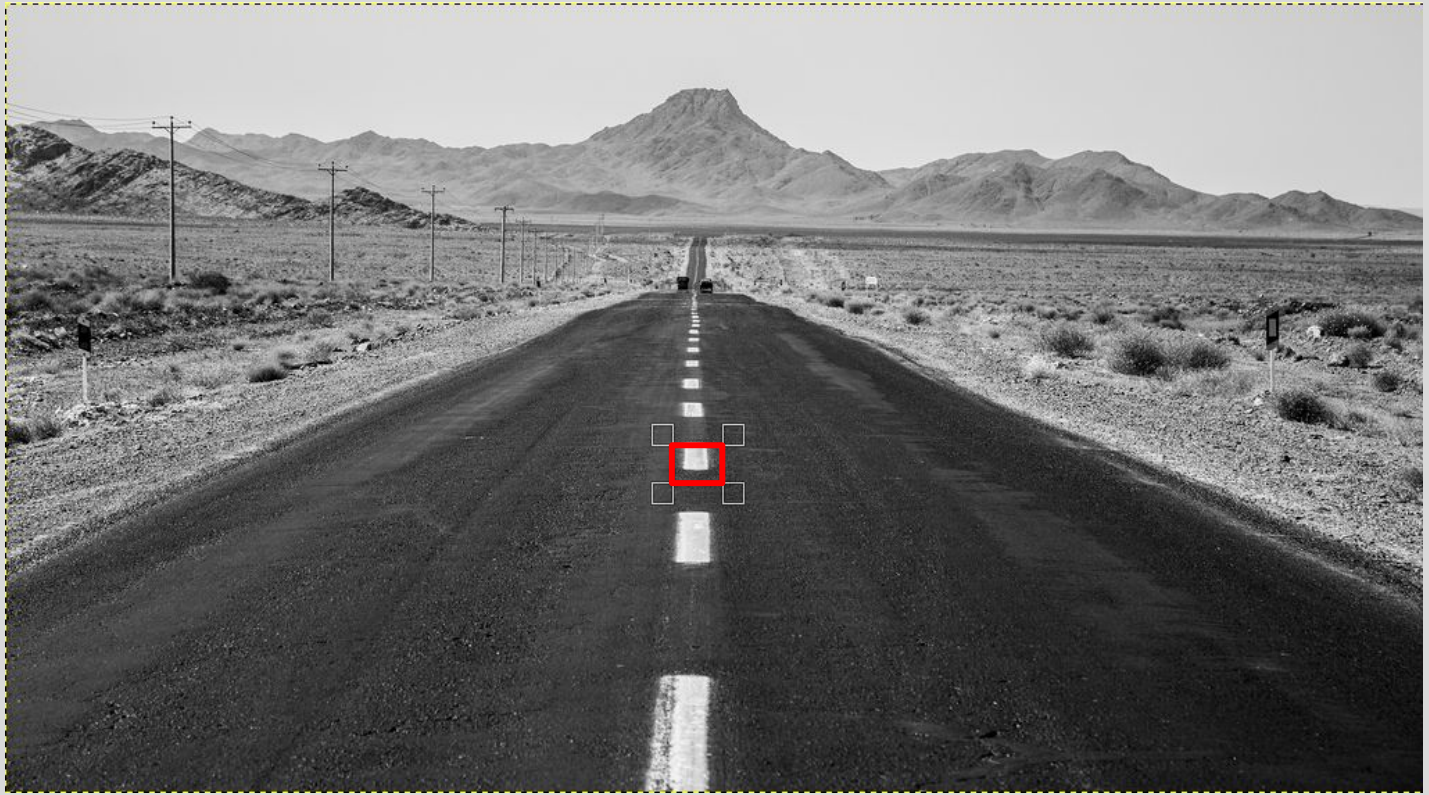
0	0	0	0	0	0
0	1	2	3	4	5
0	2	4	6	8	10
0	3	6	9	12	15
0	4	8	12	16	20
0	5	10	15	20	25

$$A = \begin{pmatrix} 3 & -5 & 4 \\ 9 & 8 & -7 \\ -6 & 4 & 2 \end{pmatrix}, B = \begin{pmatrix} -2 & -1 & 1 \\ 5 & -7 & 6 \\ 9 & 3 & 2 \end{pmatrix}$$

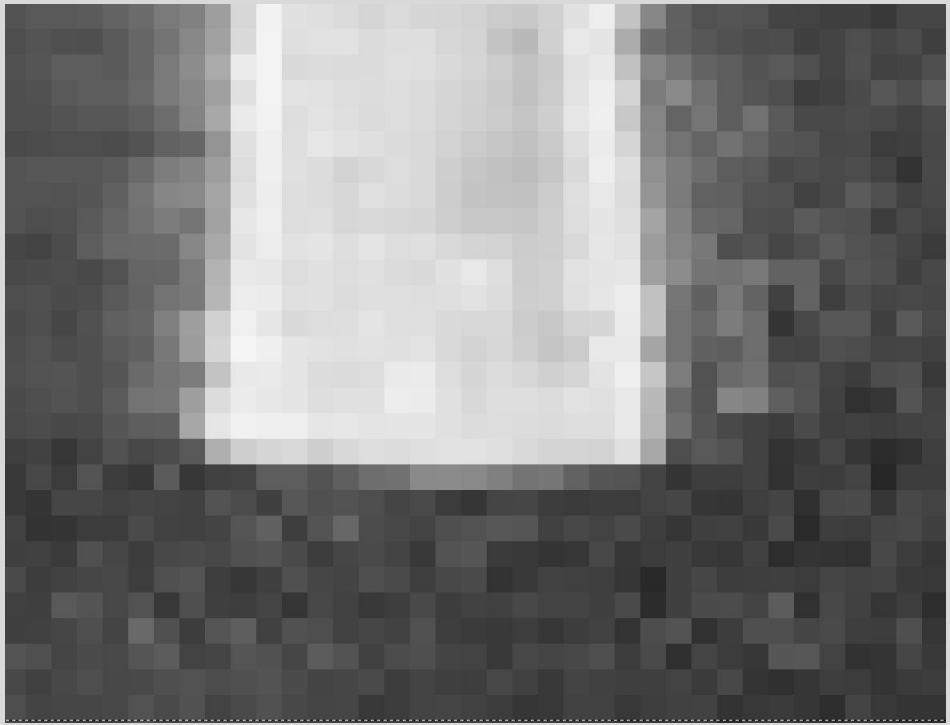
https://advancedmathclubsk.weebly.com/matrices.html

2D arrays

https://hiveminer.com/Tags/desert,isfahan/Recent



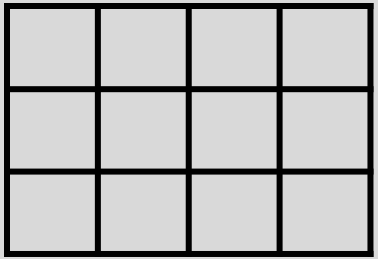
https://hiveminer.com/Tags/desert,isfahan/Recent



https://hiveminer.com/Tags/desert,isfahan/Recent

Defining 2D arrays

```
int a[3][4];
```



Initializing 2D arrays

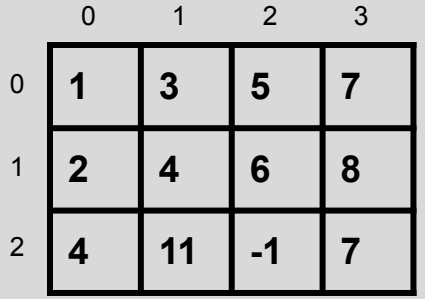
```
int a[3][4] = {{1,3,5,7}, {2,4,6,8}, {4,11,-1,7}};
```

1	3	5	7
2	4	6	8
4	11	-1	7

Initializing 2D arrays

```
int a[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {4,11,-1,7}};
```

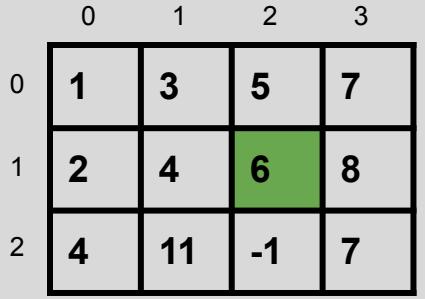
```
printf("%d\n", a[1][2]);
```



2D arrays

```
int a[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {4,11,-1,7}};
```

```
printf("%d\n", a[1][2]);
```



2D arrays

```
int a[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {4,11,-1,7}};
```

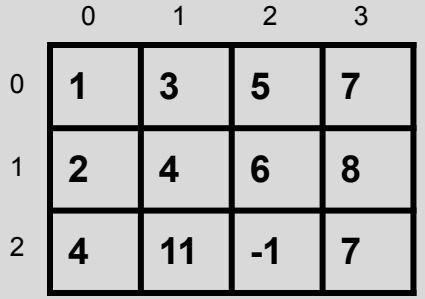
```
printf("%d\n", a[i][j]);
```

	0	1	2	3
0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

2D arrays

```
int a[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {4,11,-1,7}};
```

```
printf("%d\n", a[0][0]);
```



	0	1	2	3
0	1	3	5	7
1	2	4	6	8
2	4	11	-1	7

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int i = 0; i < 3; i++)
    printf("%d\n", a[i][2]);

  return 0;
}
```

2darray.c

	0	1	2	3
0	**1**	**3**	**5**	**7**
1	**2**	**4**	**6**	**8**
2	**4**	**11**	**-1**	**7**

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int i = 0; i < 3; i++)
    printf("%d\n", a[i][2]);

  return 0;
}
```

2darray.c

	0	1	2	3
0	**1**	**3**	**5**	**7**
1	**2**	**4**	**6**	**8**
2	**4**	**11**	**-1**	**7**

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int i = 0; i < 3; i++)
    for (int j = 0; j < 4; j++)
      printf("%d\n", a[i][j]);

  return 0;
}
```

2darray2.c

	0	1	2	3
0	1	3	5	7
1	2	4	6	8
2	4	11	-1	7

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int i = 0; i < 3; i++)
    for (int j = 0; j < 4; j++)
      printf("%d\n", a[i][j]);

  return 0;
}
```

2darray2.c

```
$ gcc 2darray2.c  && ./a.out
1
3
5
7
2
4
6
8
4
11
-1
7
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int j = 0; j < 4; j++)
    for (int i = 0; i < 3; i++)
      printf("%d\n", a[i][j]);

  return 0;
}
```

2darray3.c

	0	1	2	3
0	1	3	5	7
1	2	4	6	8
2	4	11	-1	7

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int j = 0; j < 4; j++)
    for (int i = 0; i < 3; i++)
      printf("%d\n", a[i][j]);

  return 0;
}
```

2darray3.c

```
$ gcc 2darray3.c  && ./a.out
1
2
4
3
4
11
5
6
-1
7
8
7
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int i = 0; i < 3; i++)
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

  putchar('\n');


  return 0;
}
```

2darray4.c

	0	1	2	3
0	1	3	5	7
1	2	4	6	8
2	4	11	-1	7

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  for (int i = 0; i < 3; i++)
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

  putchar('\n');


  return 0;
}
```

2darray4.c

```
$ gcc 2darray4.c  && ./a.out
  1,  3,  5,  7,  2,  4,  6,  8,  4, 11, -1,  7,
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray5.c

	0	1	2	3
0	1	3	5	7
1	2	4	6	8
2	4	11	-1	7

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray5.c

```
$ gcc 2darray5.c  && ./a.out
  1,  3,  5,  7,
  2,  4,  6,  8,
  4, 11, -1,  7,
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3},
                 {2, 4, 6, 8},
                 {4}};


  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray6.c

```
$ gcc 2darray6.c  && ./a.out
  1,  3,  0,  0,
  2,  4,  6,  8,
  4,  0,  0,  0,
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {

  int a[3][4] = {1,3,5,7,2,4,6,8,4,11,-1,7};

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray7.c

```
$ gcc 2darray7.c  && ./a.out
  1,  3,  5,  7,
  2,  4,  6,  8,
  4, 11, -1,  7,
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {

  int a[3][4] = {1,3,5,7,2,4};

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray8.c

```
$ gcc 2darray8.c  && ./a.out
  1,  3,  5,  7,
  2,  4,  0,  0,
  0,  0,  0,  0,
```

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  for (int i = 0; i < 3; i++)
    a[i][2] = 0;

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray9.c

	0	1	2	3
0	1	3	5	7
1	2	4	6	8
2	4	11	-1	7

Working with 2D arrays

```c
#include <stdio.h>

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  for (int i = 0; i < 3; i++)
    a[i][2] = 0;

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

2darray9.c

```
$ gcc 2darray9.c  && ./a.out
  1,  3,  0,  7,
  2,  4,  0,  8,
  4, 11,  0,  7,
```

Store times table in a 2D array

```c
#include <stdio.h>

#define M 10
#define N 10

int main() {
  int a[M][N];

  // write your code here

  for (int i = 0; i < M; i++) {
    for (int j = 0; j < N; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

timestable1.c

	0	1	2	3
0	1	2	3	4
1	2	4	6	8
2	3	6	9	12

Store times table in a 2D array

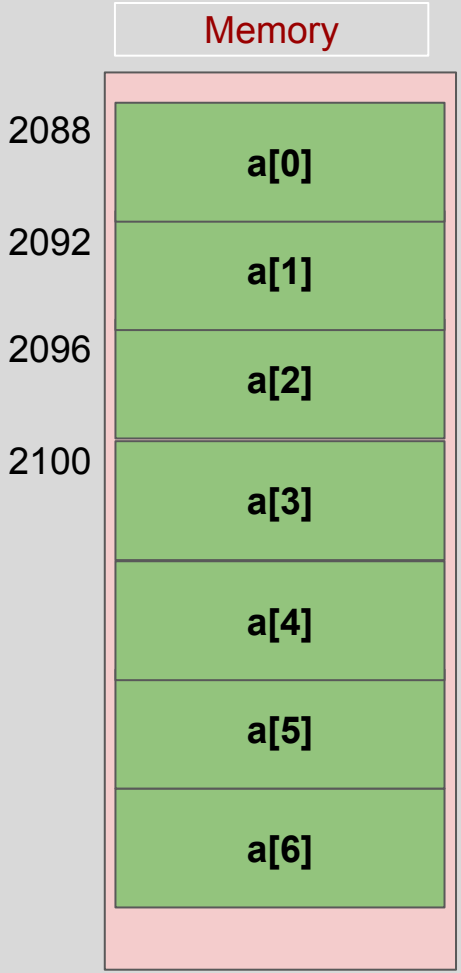
```c
#include <stdio.h>

#define M 10
#define N 10

int main() {
  int a[M][N];

  for (int i = 0; i < M; i++)
    for (int j = 0; j < N; j++)
      a[i][j] = (i+1)*(j+1);

  for (int i = 0; i < M; i++) {
    for (int j = 0; j < N; j++)
      printf("%3d,", a[i][j]);

    putchar('\n');
  }

  return 0;
}
```

timestable1.c

```
$ gcc timestable1.c && ./a.out
  1,  2,  3,  4,  5,  6,  7,  8,  9, 10,
  2,  4,  6,  8, 10, 12, 14, 16, 18, 20,
  3,  6,  9, 12, 15, 18, 21, 24, 27, 30,
  4,  8, 12, 16, 20, 24, 28, 32, 36, 40,
  5, 10, 15, 20, 25, 30, 35, 40, 45, 50,
  6, 12, 18, 24, 30, 36, 42, 48, 54, 60,
  7, 14, 21, 28, 35, 42, 49, 56, 63, 70,
  8, 16, 24, 32, 40, 48, 56, 64, 72, 80,
  9, 18, 27, 36, 45, 54, 63, 72, 81, 90,
 10, 20, 30, 40, 50, 60, 70, 80, 90,100,
```

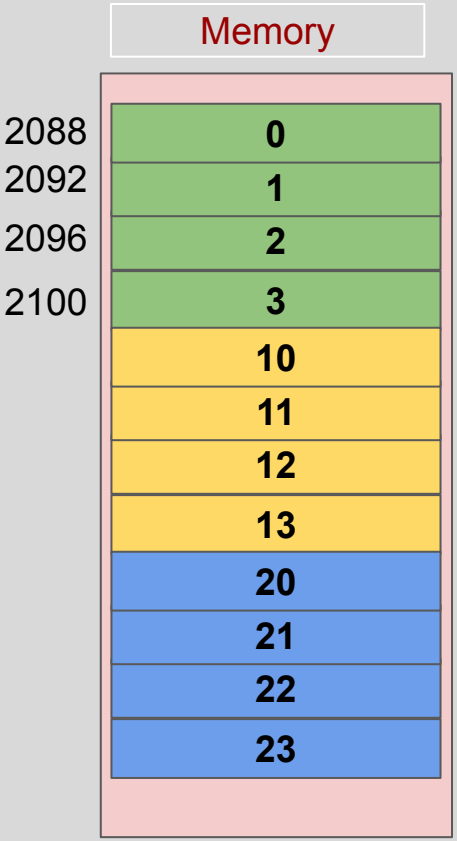
Remember: 1D arrays in memory

```
int a[7];
```



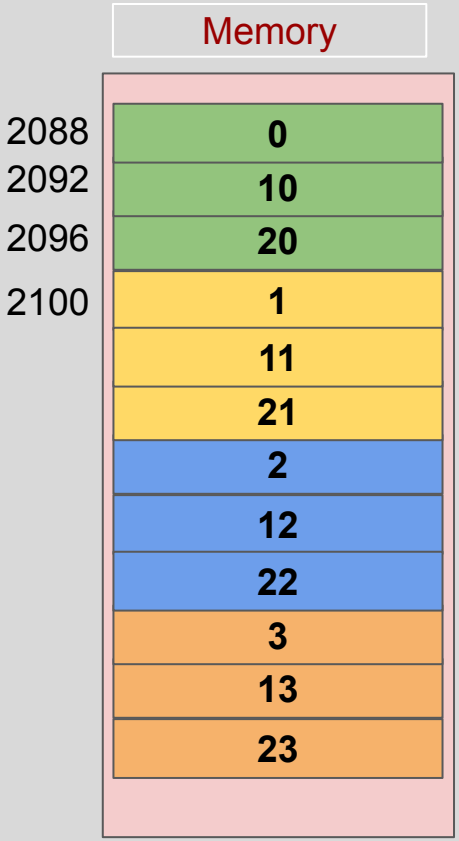
How are 2D arrays stored in memory?

	0	1	2	3
0	**0**	**1**	**2**	**3**
1	**10**	**11**	**12**	**13**
2	**20**	**21**	**22**	**23**

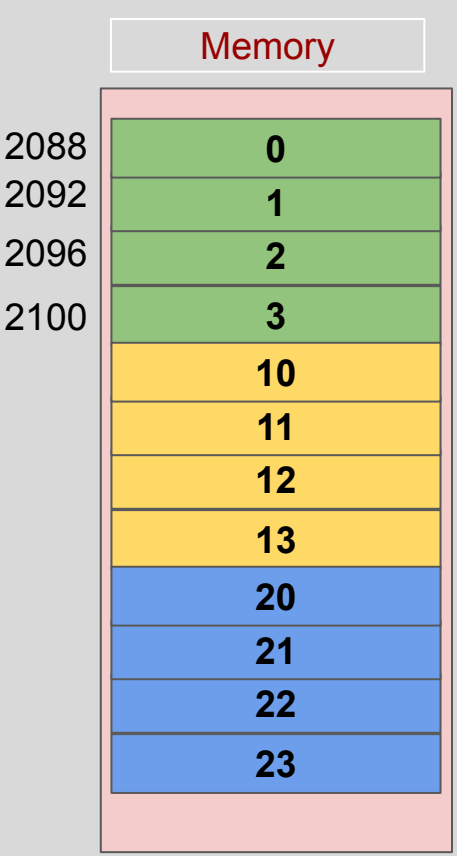


row by row

	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23

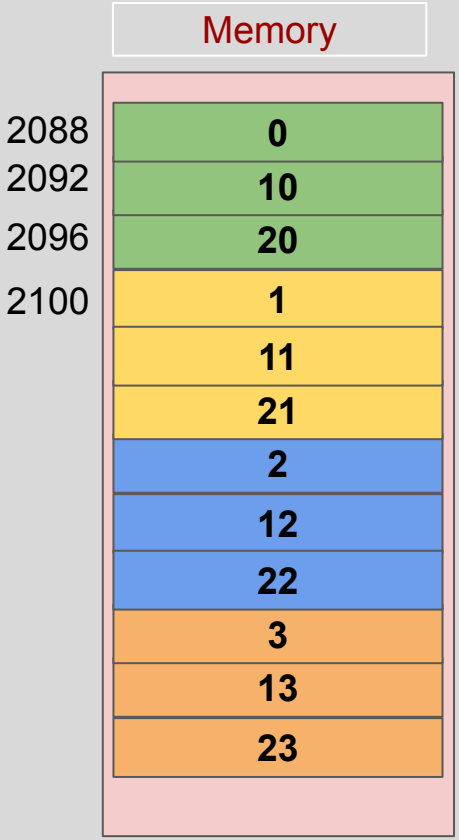


column by column

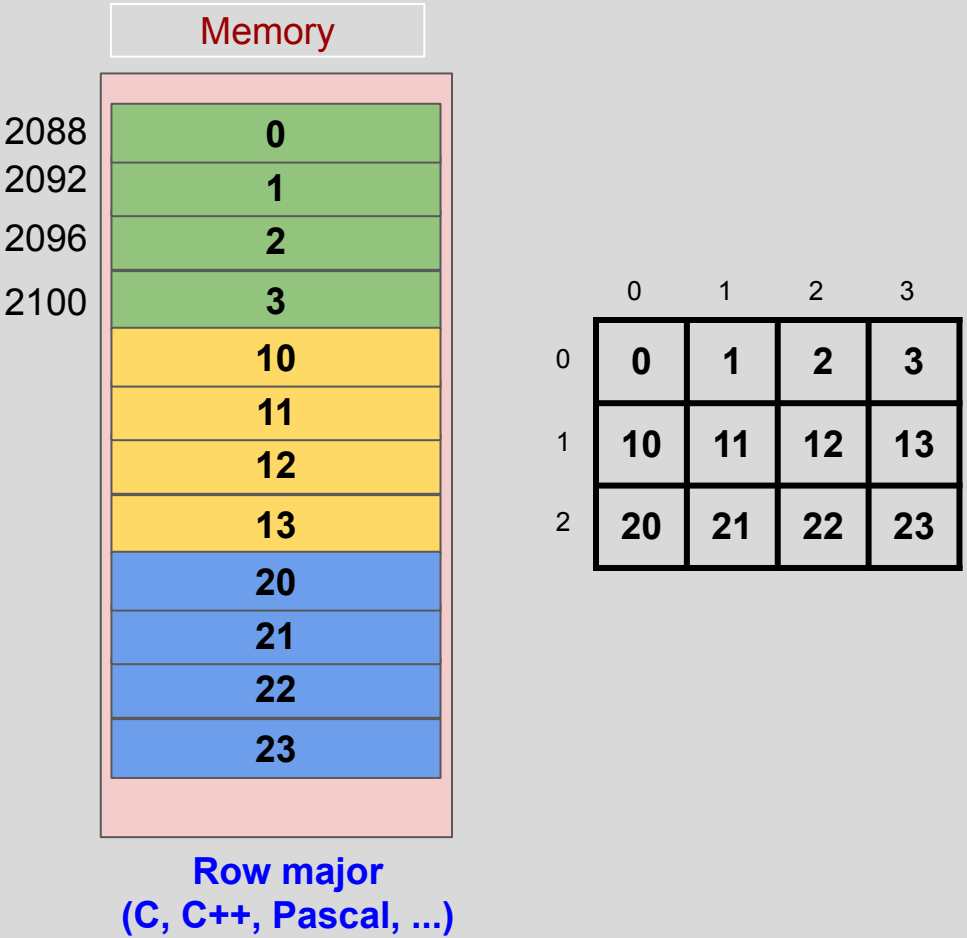


Row major
(C, C++, Pascal, ...)

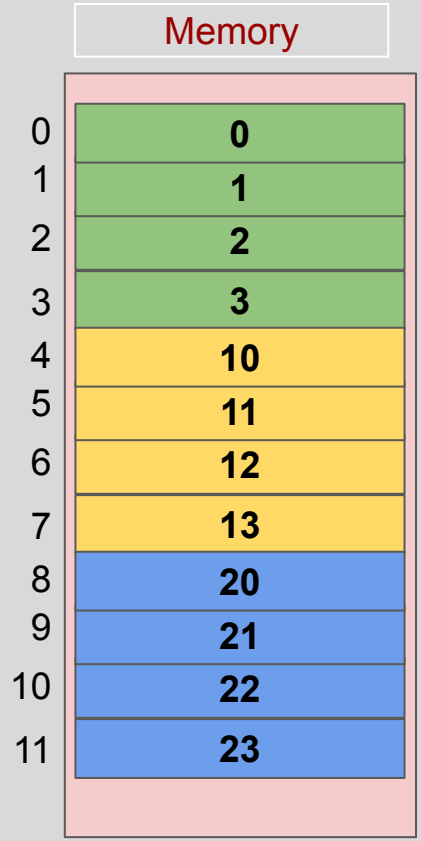
	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23



Column major (Fortran,
Matlab, R, ...)

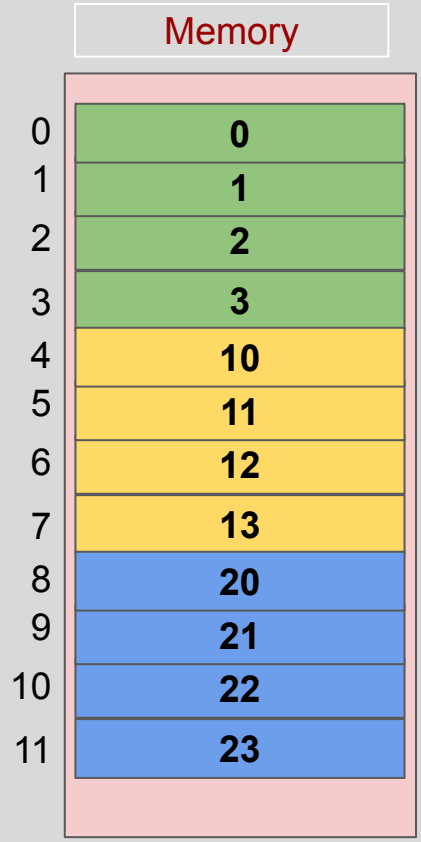


	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23



	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23

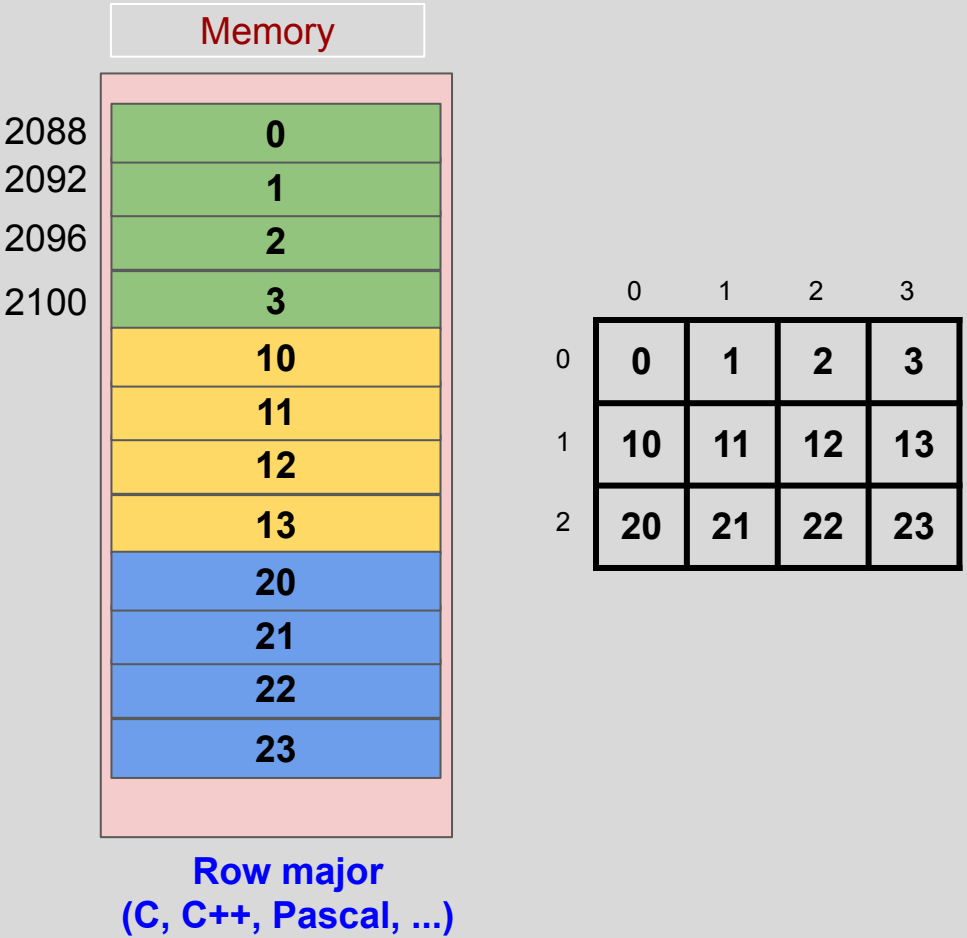
```
a[0][1]
a[0][2]
a[0][j]

a[1][0]
a[1][1]
a[1][2]
a[1][j]

a[2][j]

a[i][j]
```



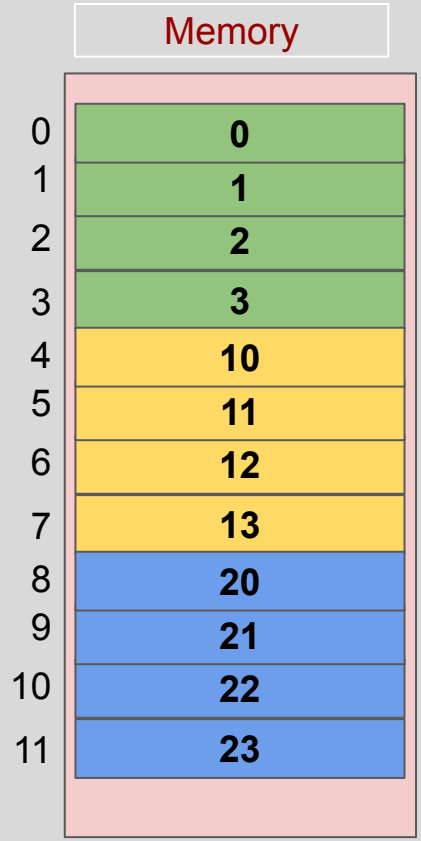
Row major
(C, C++, ...)

	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23

```
a[0][1]
a[0][2]
a[0][j]

a[1][0]
a[1][1]
a[1][2]
a[1][j]

a[2][j]

a[i][j] : 4*i + j
```

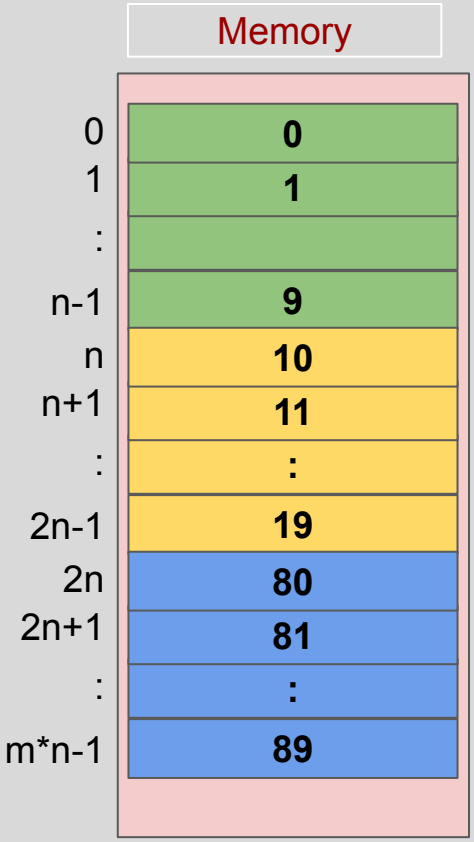


	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23



	0	1	2	3
0	0	1	2	3
1	10	11	12	13
2	20	21	22	23

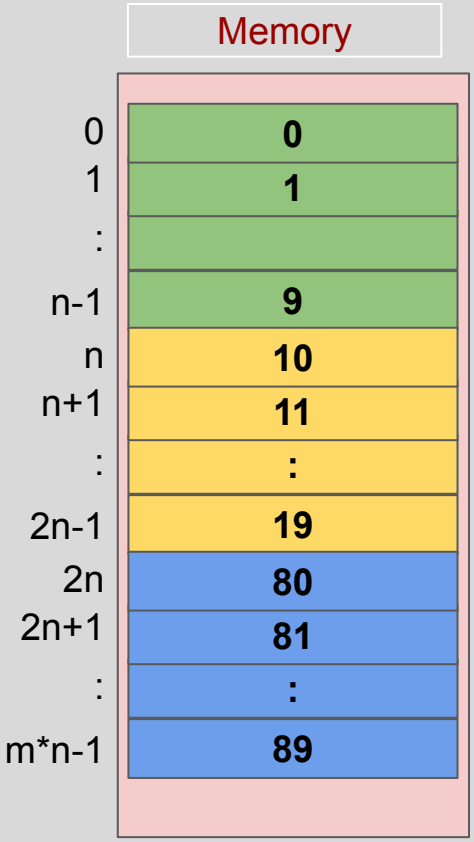
```
a[0][1]
a[0][2]
a[0][j]

a[1][0]
a[1][1]
a[1][2]
a[1][j]

a[2][j]

a[i][j]
```



	0	1		n-1
0	0	1	...	9
1	10	11	...	19
	:	:		:
m-1	80	81	...	83

```
int a[m][n];
```

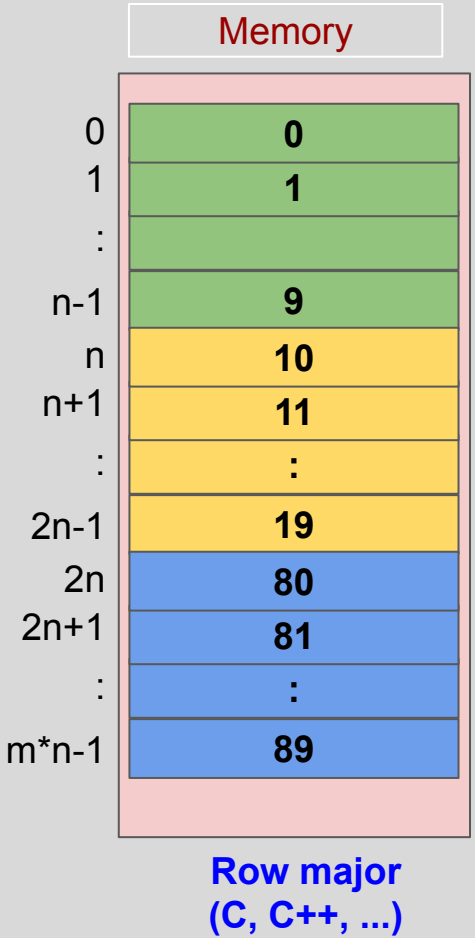
```
a[i][j] : n*i + j
```

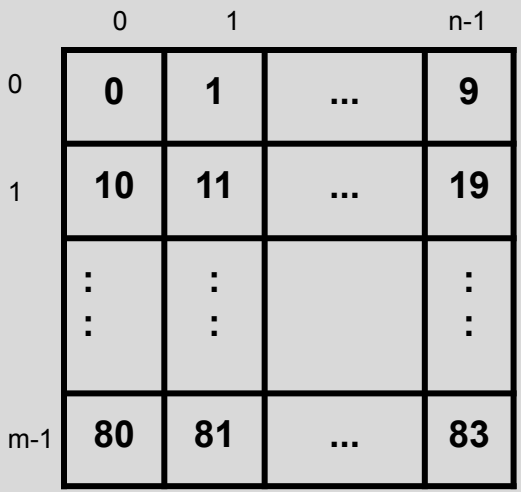


	0	1		n-1
0	0	1	...	9
1	10	11	...	19
	:	:		:
m-1	80	81	...	83

```
int a[m][n];
```

```
a[i][j] : n*i + j
```






```
int a[m][n];
```

```
a[i][j] : n*i + j
```

to find `a[i][j]` the **compiler** needs to know:

- **m** (no. of rows of a)
- **n** (no. of columns of a)
- both m and n

Passing 2D arrays to functions

```c
#include <stdio.h>

int print2Darray(int a[][4], int,int);

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  print2Darray(a, 3,4);

  return 0;
}

int print2Darray(int a[][4], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }


}
```

print2darray1.c

Passing 2D arrays to functions

```c
#include <stdio.h>

#define N 4

int print2Darray(int a[][N], int,int);

int main() {
  int a[3][N] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  print2Darray(a, 3,N);

  return 0;
}

int print2Darray(int a[][N], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }
}
```

print2darray2.c

A

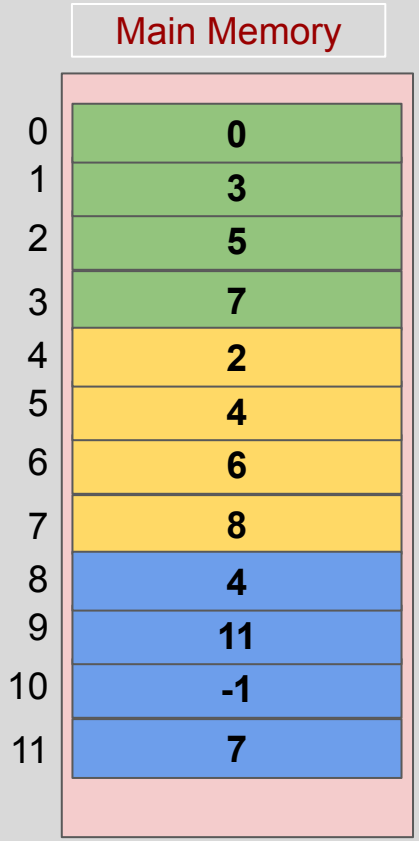
```c
#include <stdio.h>

#define N 4

void print2Darray(int a[][N], int,int);
void printArray(int a[], int);

int main() {
  int a[3][N] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  printArray(a[1],4);

  return 0;
}

void print2Darray(int a[][N], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }

}

void printArray(int arr[], int n) {
   for (int i = 0; i < n; i++)
      printf("%4d,", arr[i]);

   putchar('\n');
}
```

```c
#include <stdio.h>

#define N 4

void print2Darray(int a[][N], int,int);
void printArray(int a[], int);

int main() {
  int a[3][N] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  printArray(a[1],7);

  return 0;
}

void print2Darray(int a[][N], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }

}

void printArray(int arr[], int n) {
   for (int i = 0; i < n; i++)
      printf("%4d,", arr[i]);

   putchar('\n');
}
```

print2darray3.c

```c
#include <stdio.h>

#define N 4

void print2Darray(int a[][N], int,int);
void printArray(int a[], int);

int main() {
  int a[3][N] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  printArray(a[1],4);

  return 0;
}

void print2Darray(int a[][N], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }

}

void printArray(int arr[], int n) {
   for (int i = 0; i < n; i++)
      printf("%4d,", arr[i]);

   putchar('\n');
}
```

```c
#include <stdio.h>

#define N 4

void print2Darray(int a[][N], int,int);
void printArray(int a[], int);

int main() {
  int a[3][N] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  printArray(a[1],7);

  return 0;
}

void print2Darray(int a[][N], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }

}

void printArray(int arr[], int n) {
   for (int i = 0; i < n; i++)
      printf("%4d,", arr[i]);

   putchar('\n');
}
```



Row major
(C, C++, ...)

Passing 2D arrays to functions

```c
#include <stdio.h>

#define N 4

int print2Darray(int a[][N], int,int);

int main() {
  int a[3][N] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};


  print2Darray(a, 3,N);

  return 0;
}

int print2Darray(int a[][N], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }
}
```

print2darray2.c

Passing 2D arrays to functions

```c
#include <stdio.h>

void print2Darray(int m, int n, int a[][n]);

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  print2Darray(3,4,a);

  return 0;
}

void print2Darray(int m, int n, int a[][n]) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }
}
```

print2darray4.c

```c
#include <stdio.h>

void print2Darray(int a[][n], int m, int n);

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  print2Darray(3,4,a);

  return 0;
}

void print2Darray(int a[][n], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }
}
```

print2darray5.c

Passing 2D arrays to functions

```c
#include <stdio.h>

void print2Darray(int m, int n, int a[][n]);

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  print2Darray(3,4,a);

  return 0
}

void print2Darray(int m, int n, int a[][n]) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }

}
```

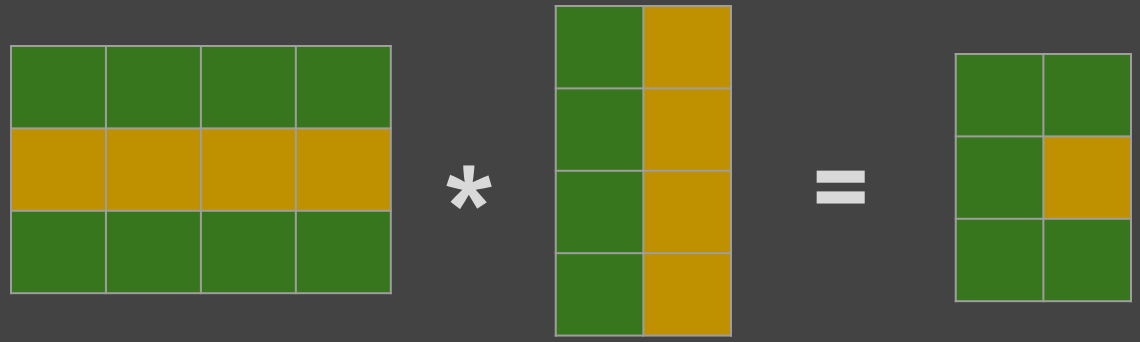
```
   1,   3,   5,   7,
   2,   4,   6,   8,
   4,  11,  -1,   7,
```

print2darray4.c

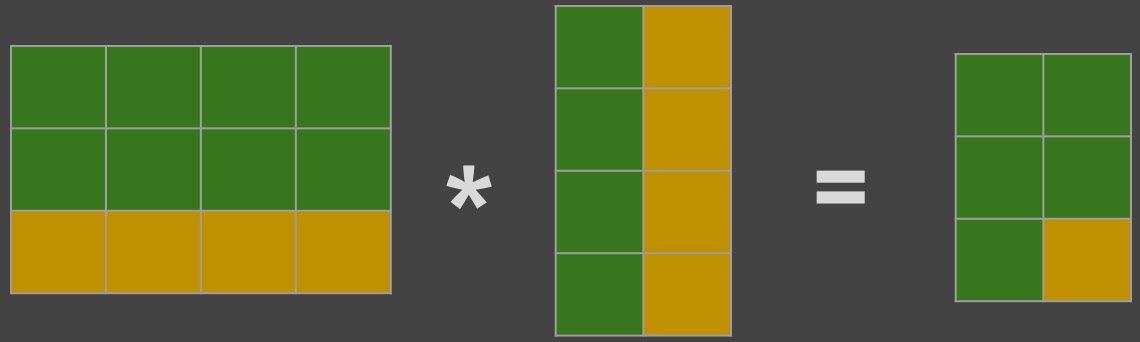
```c
#include <stdio.h>

void print2Darray(int a[][n], int m, int n);

int main() {
  int a[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {4,11,-1,7}};

  print2Darray(3,4,a);

void print2Darray(int a[][n], int m, int n) {
   for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++)
      printf("%4d,", a[i][j]);

    putchar('\n');
   }

}
```

```
print2Darray5.c: At top level:
print2Darray5.c:15:27: error: ‘n’ undeclared here (not in a function)
 void print2Darray(int a[][n], int m, int n) {
                           ^
```

Multiplying two matrices



Multiplying two matrices



Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1});

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = ?
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1}};

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = 8 * 0 + 7 * 3 + 6 * 2 + 5 * 1;
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1}};

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = A[i][0]*B[0][j] +
          A[i][1]*B[1][j] +
          A[i][2]*B[2][j] +
          A[i][3]*B[3][j];
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1}};

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = 0;
for (int k = 0; k < 4; k++)
  C[i][j] +=  A[i][k]*B[k][j];
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8,7,6,5}};

int B[4][2] = {{1, 0},
               {0, 3},
               {1, 2},
               {-1, 1}};

int C[3][2];

for (int i = 0; i < 3; i++) {
  for (int j = 0; j < 2; j++) {

    C[i][j] = 0;
    for (int k = 0; k < 4; k++)
      C[i][j] +=  A[i][k]*B[k][j];

  }
}
```

Multiplying two matrices

```c
int main() {
  int A[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {8,7,6,5}};

  int B[4][2] = {{1, 0},
                 {0, 3},
                 {1, 2},
                 {-1, 1}};

  int C[3][2];

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 2; j++) {

      C[i][j] = 0;
      for (int k = 0; k < 4; k++)
        C[i][j] += A[i][k]*B[k][j];

    }
  }

  print2Darray(3,4, A);
  putchar('\n');

  print2Darray(4,2, B);
  putchar('\n');

  print2Darray(3,2, C);
  putchar('\n');
```

```
nasihatkon@kntu:code$ gcc matrix_mul3.c && ./a.out
   1,   3,   5,   7,
   2,   4,   6,   8,
   8,   7,   6,   5,

   1,   0,
   0,   3,
   1,   2,
  -1,   1,

  -1,  26,
   0,  32,
   9,  38,
```

Matrix multiplication function

```c
#include <stdio.h>

void print2Darray(int m, int n, int a[][n]);
void matrix_mul(int m, int n, int p, int A[m][n], int B[n][p], int C[m][p]);

int main() {

  int A[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {8,7,6,5}};

  int B[4][2] = {{1, 0},
                 {0, 3},
                 {1, 2},
                 {-1, 1}};

  int C[3][2];

  matrix_mul(3,4,2,A,B,C);

  print2Darray(3,4, A);
  putchar('\n');

  print2Darray(4,2, B);
  putchar('\n');

  print2Darray(3,2, C);
  putchar('\n');

  return 0;
}
```

Matrix multiplication function

```
void matrix_mul(int m, int n, int p, int A[m][n], int B[n][p], int C[m][p]) {
  for (int i = 0; i < m; i++) {
    for (int j = 0; j < p; j++) {
      C[i][j] = 0;
      for (int k = 0; k < n; k++)
        C[i][j] +=  A[i][k]*B[k][j];
    }
  }
}
```

```
void matrix_mul(int m, int n, int p, int A[][n], int B[][p], int C[][p]) {
  for (int i = 0; i < m; i++) {
    for (int j = 0; j < p; j++) {
      C[i][j] = 0;
      for (int k = 0; k < n; k++)
        C[i][j] +=  A[i][k]*B[k][j];
    }
  }
}
```

Matrix multiplication function

```c
#include <stdio.h>

void print2Darray(int m, int n, int a[][n]);
void matrix_mul(int m, int n, int p, int A[m][n], int B[n][p], int C[m][p]);

int main() {

  int A[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {8,7,6,5}};

  int B[4][2] = {{1, 0},
                 {0, 3},
                 {1, 2},
                 {-1, 1}};

  int C[3][2];

  matrix_mul(3,4,2,A,B,C);

  print2Darray(3,4, A);
  putchar('\n');

  print2Darray(4,2, B);
  putchar('\n');

  print2Darray(3,2, C);
  putchar('\n');

  return 0;
}
```

```
nasihatkon@kntu:code$ gcc matrix_mul4.c && ./a.out
   1,   3,   5,   7,
   2,   4,   6,   8,
   8,   7,   6,   5,

   1,   0,
   0,   3,
   1,   2,
  -1,   1,

  -1,  26,
   0,  32,
   9,  38,
```

Tabular data

	Farzam	Ramin	Bardia	Sobhan
Programming	18	20	15	17
Calculus	12	2	20	18
Farsi	16	16	17	14

Course grades

Tabular data

```
#define FARZAM    0
#define RAMIN     1
#define BARDIA    2
#define SOBHAN    3

#define PROGRAMMING  0
#define CALCULUS     1
#define FARSI        2

#define NO_COURSE  3   // number of courses
#define NO_STUDENT 4   // number of students

double max_course(double grades[][NO_STUDENT], int course);
double max_student(double grades[][NO_STUDENT], int student);
double average_course(double grades[][NO_STUDENT], int course);
double average_student(double grades[][NO_STUDENT], int student);
```

Tabular data

```c
int main() {

  // grades: FARZAM, RAMIN, BARDIA, SOBHAN
  double grades[3][4] = {{18, 20, 15, 17},  // PROGRAMMING
                         {12,  2, 20, 19},  // CALCULUS
                         {16, 16, 17, 14}}; // FARSI


  printf("Max grade of Programming is %.2f\n", max_course(grades, PROGRAMMING));
  printf("Max grade of Sobhan      is %.2f\n", max_student(grades, SOBHAN));
  printf("Average   of Farsi       is %.2f\n", average_course(grades, FARSI));
  printf("Average   of Bardia      is %.2f\n", average_student(grades, BARDIA));


  return  0;
}
```

Compute average/max of a 1D array

```
double avg_array(double array[], int size) {
  double sum = 0;

  for (int i = 0; i < size; i++)
    sum += array[i];

  return sum/size;
}

double max_array(double array[], int size) {
  double max = array[0];

  for (int i = 1; i < size; i++)
    if (array[i] > max)
      max = array[i];

  return max;

}
```

Compute average/max of a course

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

```
double max_course(double grades[][NO_STUDENT], int course) {
  return max_array(grades[course], NO_STUDENT);
}
```

```
double average_course(double grades[][NO_STUDENT], int course) {
  return avg_array(grades[course], NO_STUDENT);
}
```

Compute max of a student

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

	Farzam	Ramin	Bardia	Sobhan
Programming	18	20	15	17
Calculus	12	2	20	18
Farsi	16	16	17	14

Course grades

Compute max of a student

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

```
double max_student(double grades[][NO_STUDENT], int student) {
  double student_grades[NO_COURSE];

  for (int i = 0; i < NO_COURSE; i++)
    student_grades[i] = grades[i][student];

 max_array(student_grades, NO_COURSE);
}
```

Compute average of a student

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

```
double average_student(double grades[][NO_STUDENT], int student) {
  double student_grades[NO_COURSE];

  for (int i = 0; i < NO_COURSE; i++)
    student_grades[i] = grades[i][student];

  return avg_array(student_grades, NO_COURSE);
}
```

Running the code

```
int main() {

  // grades: FARZAM, RAMIN, BARDIA, SOBHAN
  double grades[3][4] = {{18, 20, 15, 17},  // PROGRAMMING
                         {12,  2, 20, 19},  // CALCULUS
                         {16, 16, 17, 14}}; // FARSI


  printf("Max grade of Programming is %.2f\n", max_course(grades, PROGRAMMING));
  printf("Max grade of Sobhan      is %.2f\n", max_student(grades, SOBHAN));
  printf("Average   of Farsi       is %.2f\n", average_course(grades, FARSI));
  printf("Average   of Bardia      is %.2f\n", average_student(grades, BARDIA));


  return  0;
}
```

Output

```c
int main() {

  // grades: FARZAM, RAMIN, BARDIA, SOBHAN
  double grades[3][4] = {{18, 20, 15, 17},  // PROGRAMMING
                         {12,  2, 20, 19},  // CALCULUS
                         {16, 16, 17, 14}}; // FARSI


  printf("Max grade of Programming is %.2f\n", max_course(grades, PROGRAMMING));
  printf("Max grade of Sobhan      is %.2f\n", max_student(grades, SOBHAN));
  printf("Average   of Farsi       is %.2f\n", average_course(grades, FARSI));
  printf("Average   of Bardia      is %.2f\n", average_student(grades, BARDIA));


  return  0;
}
```

```
nasihatkon@kntu:code$ gcc tabular1.c && ./a.out
Max grade of Programming is 20.00
Max grade of Sobhan      is 19.00
Average   of Farsi       is 15.75
Average   of Bardia      is 17.33
```

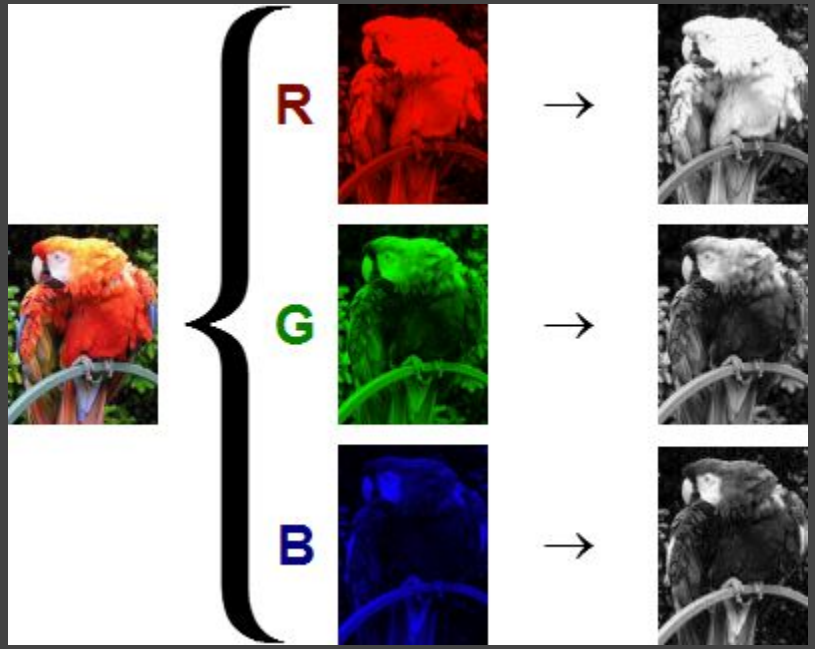
N-dimensional arrays

```
#define FARZAM    0
#define RAMIN     1
#define BARDIA    2
#define SOBHAN    3

#define PROGRAMMING  0
#define CALCULUS     1
#define FARSI        2

#define MIDTERM_EXAM  0
#define FINAL_EXAM    1

#define NO_COURSE  3   // number of courses
#define NO_STUDENT 4   // number of students
#define NO_EXAMS   2   // number of exams


double grades[3][4][2] = {
  { {10,20},{10, 20},{10, 20} },
  { {11,12},{13,14},{15,16} },
  { {20,19},{18,17},{10,9} }
}
```

N-dimensional arrays

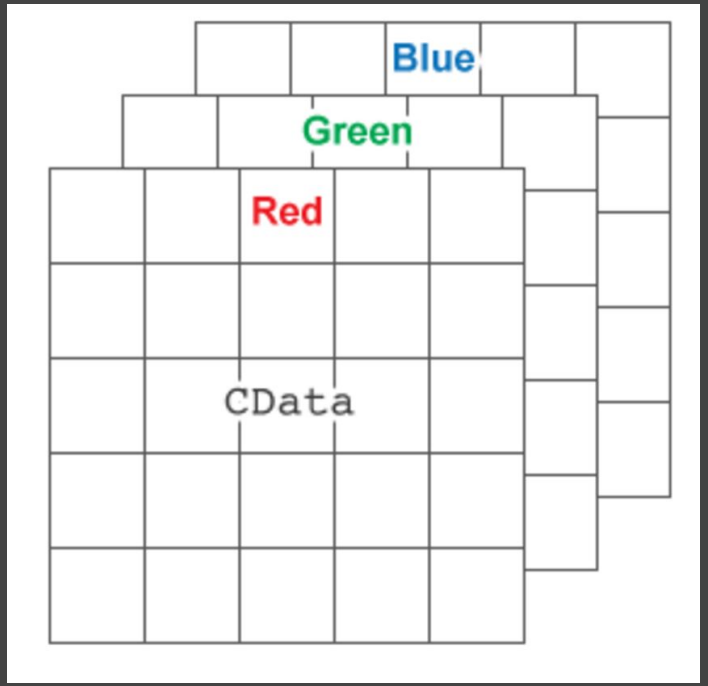
```
double max_course_exam(double grades[][4][2], int course, int exam);
```

Color Images



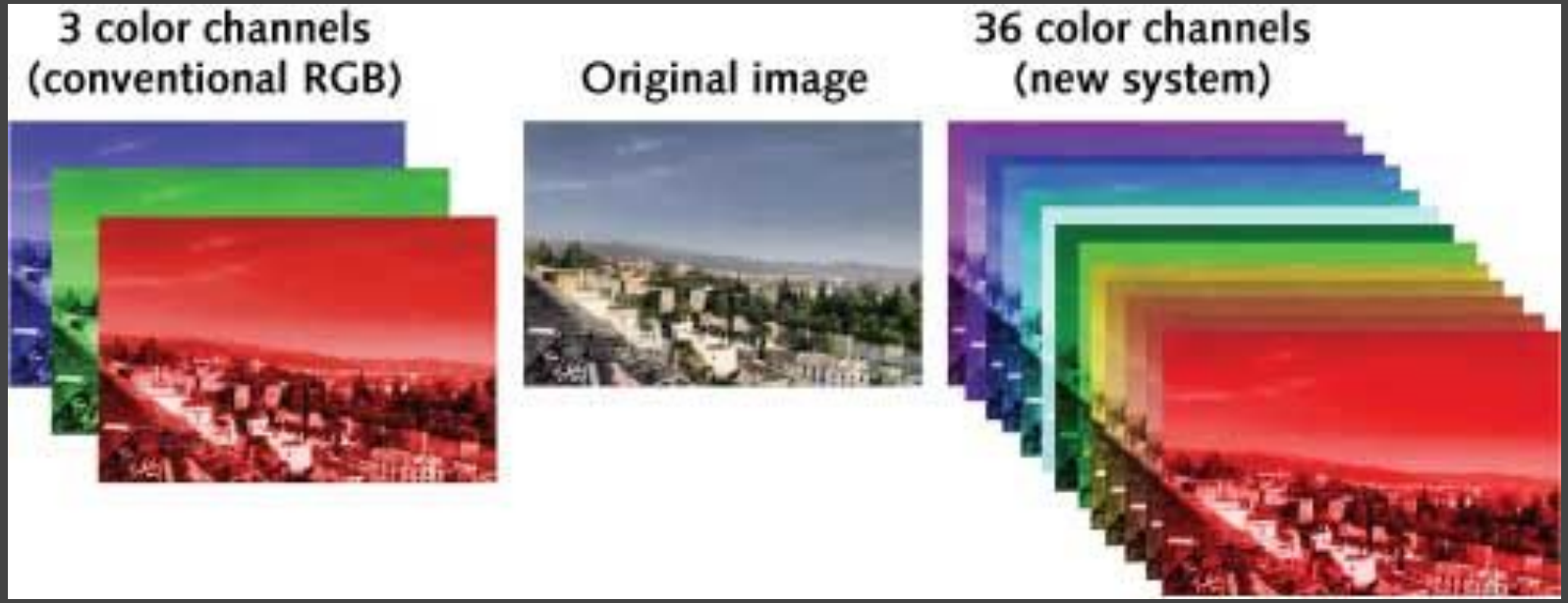
https://commons.wikimedia.org/wiki/File:Beyoglu_4671_tricolor.png#/media/File:RGB_channels_separation.png

Color Images



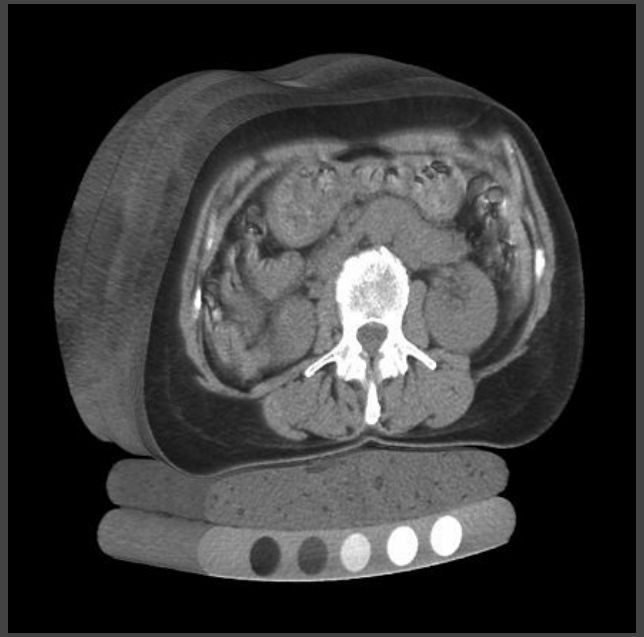
https://www.mathworks.com/help/matlab/ref/image.html?requestedDomain=www.mathworks.com

Multi-spectral images

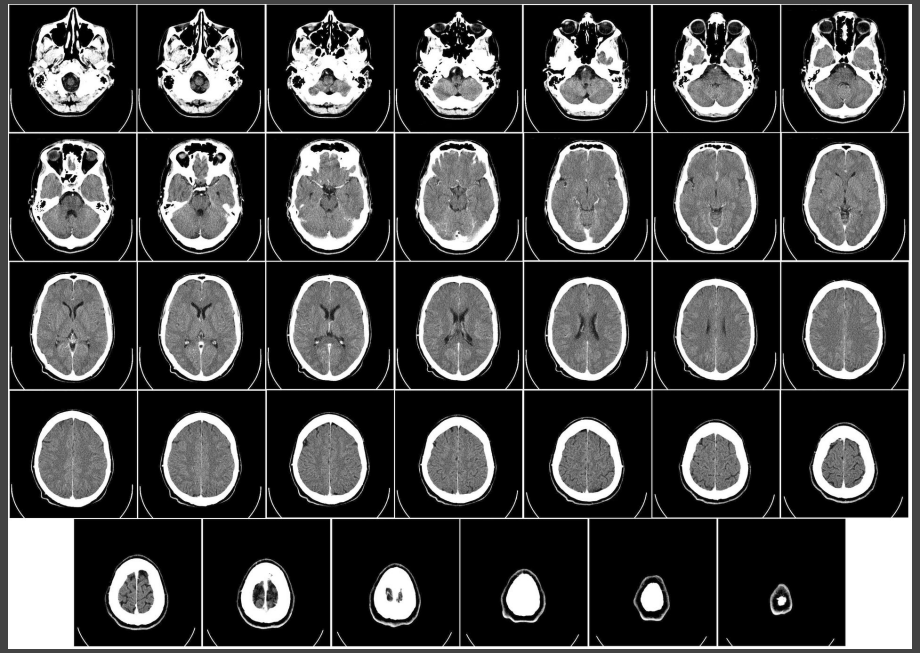


http://www.laserfocusworld.com/articles/print/volume-50/issue-11/world-news/multispectral-imaging-transverse-field-detector-sensor-has-36-color-channels.html

MRI/CT/PET image volume

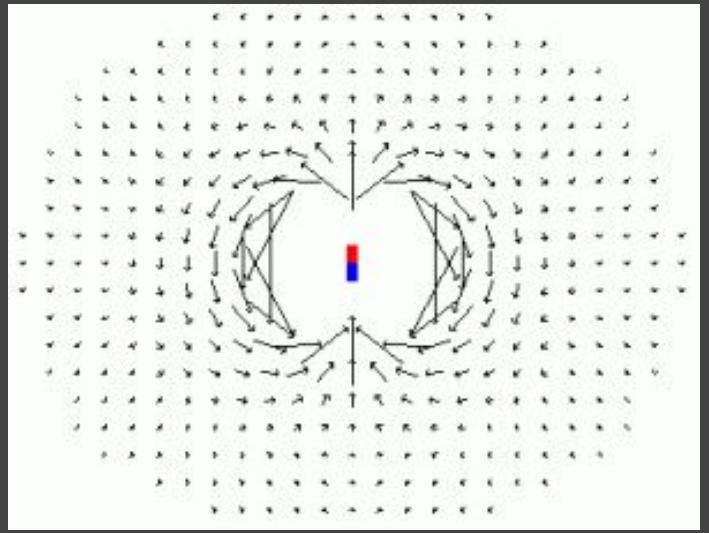


https://en.wikipedia.org/wiki/Volume_rendering

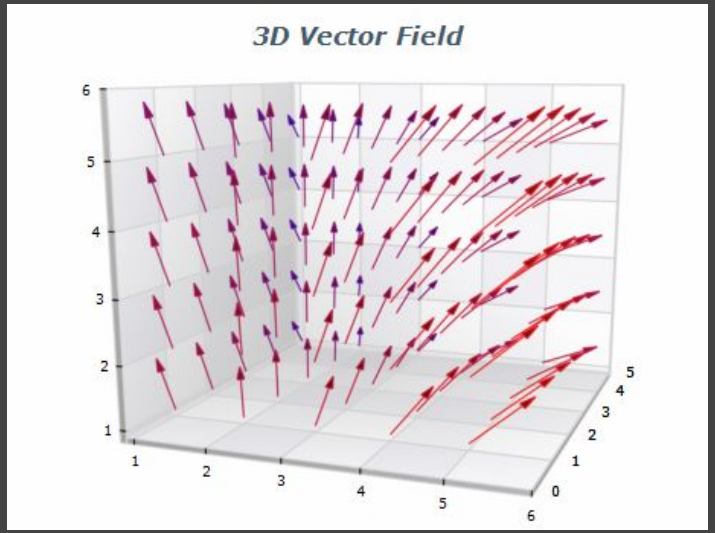


https://en.wikipedia.org/wiki/CT_scan

Vector fields



http://www.evsc.net/home/dipole-spin-system



https://glennmarshall.wordpress.com/2014/10/31/3d-neon-vector-fields/