Introduction to Programming and Data Structure

Subject Code: CS1L001

Dr Sudipta Saha

Assistant Professor

Arrays

Computer Science and Engineering,
School of Electrical Sciences
Indian Institute of Technology Bhubaneswar

Arrays

What is it

Why required in C programming – in general

What sort of problem we can do using arrays and cannot do without arrays

Arrays

How to use it with loops

How to use it with conditional statements

Solving general and special problems using Arrays

Arrays

An array in C or C++ is a collection of items stored at contiguous memory locations

Elements can be accessed randomly using indices

Used to store similar type of elements (The data type must be the same for all elements)

Can be used with - *int, float, double, char*, etc. of any particular type [and more …]

Is there any way to have these 1000 or more number of variables get declared to together in a single statement in the programming language ??

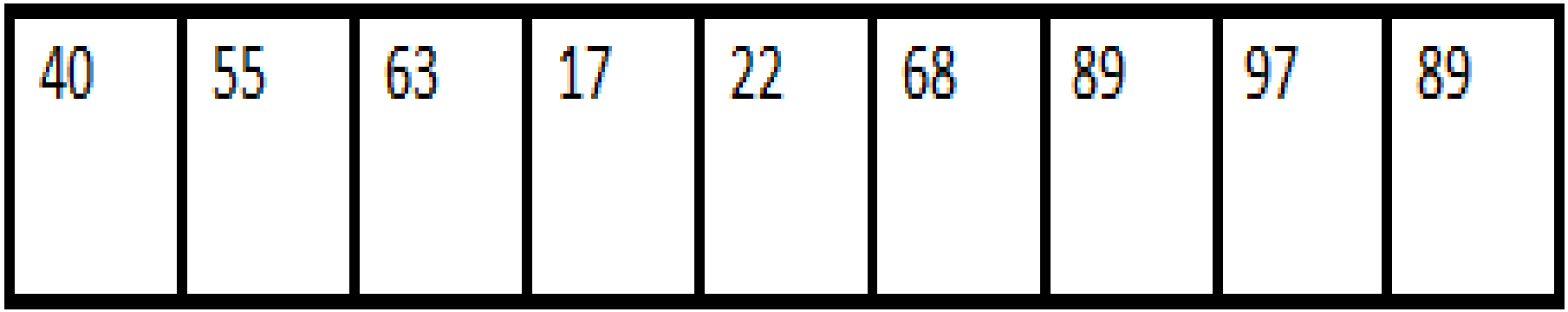
Also, can we get some efficient / uniform way to access these variables….

```
int main(){

        int a, b, c, d ….., aa, ab, ….;
        ……
        // Declare one thousand variable together
        int a[1000];

        a[0],…. a[10], a[999]…..
}
```

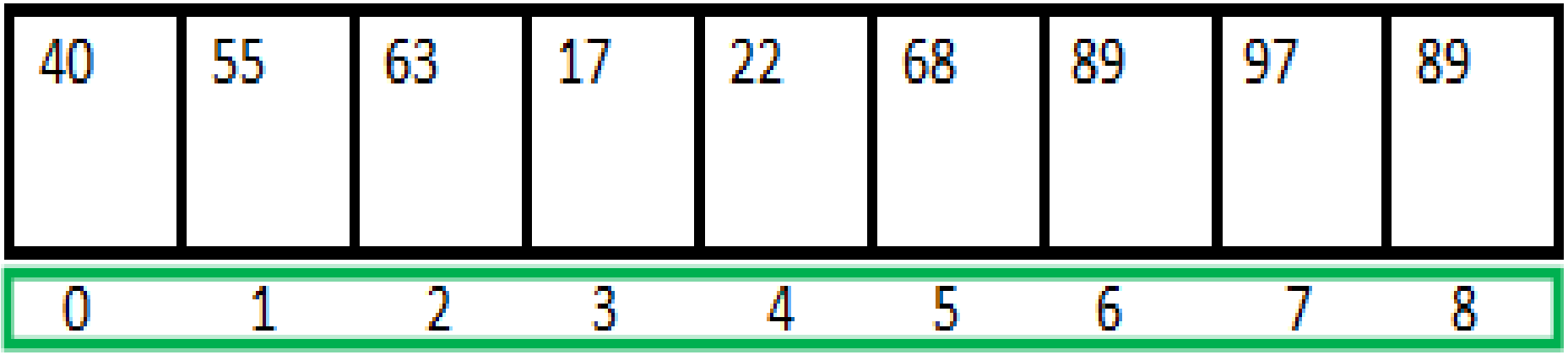
```
int main(){

        int a1, a2, a3,…, a1000;
        // Declare one thousand variable together
        int a[1000];

        a[0],…. a[10], a[999]…..

        i varies from 0 to 999
        a[i]

}
```

```
int main(){

        int a[10];          // 10 means 0 to 9; total 10 variables
                            // are there
        a[0]=1;
        a[1]=2;
        a[9]=9;             //….


        printf(“%d”,a[0]);
        printf(“%d”,a[10]);

}
```

Arrays



Arrays



Array Length = 9

First Index = 0

Last Index = 8

Why do we need arrays?

Without array / With array –

We can use normal variables (v1, v2, v3, ..) when we have a small number of elements to deal with

But if we want to store a large number of instances, it becomes difficult to manage them with normal variables

To represent many instances in one variable

Example

Collect the roll number of 100 students and store them. Print them in a serial order.

```
int  main(){
        int roll1, roll2, …..;
        while(i<=100){
                if i == 1   scan the number roll1;
                if I == 2 scan the number to roll2,….
                ……
                print the number
                i++
        }
}
```

Example

Collect the roll number of 100 students and store them. Print them in a serial order.

int roll1, roll2, roll3,……, roll100

Example

Collect the roll number of 100 students and store them. Print them in a serial order.

int roll1, roll2, roll3,……, roll100

Is this a general feasible way of doing the work ?

Example

Collect the roll number of 100 students and store them. Print them in a serial order.

No, Use Arrays

Example

Collect the roll number of 100 students and store them. Print them in a serial order.

No, Use Arrays

Array declaration in C

Example

int roll[100]; int a[5];

roll[0], roll[1], roll[2]…. roll[99].

General rule –

dataType arrayName[arraySize];

Example

Collect the roll number of 100 students and store them. Print them in a serial order.

```
int  main(){
        int roll[100]; i=0;
        while(i<100){
                scanf("%d", &roll[i++]);
        }
        i=0;
        while(i<100){
                printf("%d", &roll[i++]);
        }

}
```

Example

- int main(){

- int i, j, k, l, m;
- char x, y, z, a;
- int z, aa, bb, cc, dd;
- …
- Float ….

- }

More example declarations

float mark[5];

An array, mark, of floating-point type

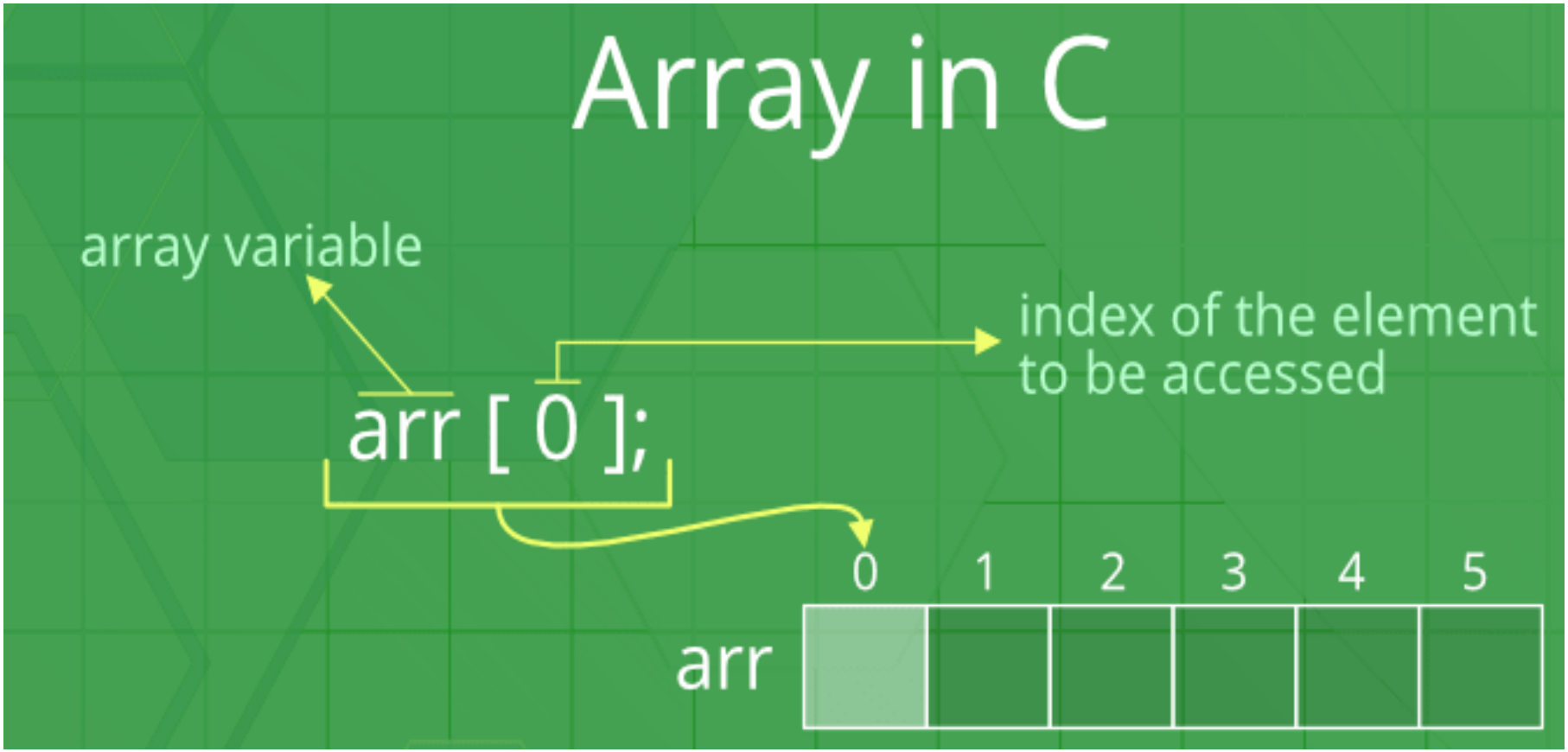
Size is 5

It can hold 5 floating-point values

Size and type of an array cannot be changed once it is declared.

Access Array Elements

Access elements of an array by indices – int arr[6];



Example

```
#include <stdio.h>

int main()
{
    int arr[5];
}
```

```
  |  arr[0]          arr[1]    arr[2]     arr[3]    arr[4]    |
```

Example

```
#include <stdio.h>

int main()
{
    int arr[5];
    arr[0] = 5;
}
```

Example

```
#include <stdio.h>

int main()
{
    int arr[5];
    arr[0] = 5;
    arr[2] = -10;
}
```

Example

```
#include <stdio.h>

int main()
{
    int arr[5];
    arr[0] = 5;
    arr[2] = -10;
    arr[3 / 2] = 2; // this is same as arr[1] = 2
}
```

Example

```
#include <stdio.h>

int main()
{
    int arr[5];
    arr[0] = 5;
    arr[2] = -10;
    arr[3 / 2] = 2; // this is same as arr[1] = 2
    arr[3] = arr[0];
}
```

Example

```
#include <stdio.h>

int main()
{
    int arr[5];
    arr[0] = 5;
    arr[2] = -10;
    arr[3 / 2] = 2; // this is same as arr[1] = 2
    arr[3] = arr[0];
    printf("%d %d %d %d", arr[0], arr[1], arr[2], arr[3]);
    return 0;
}
```

Output

5 2 -10 5

Bounds

No Index Out of bound Checking:

There is no index out of bounds checking in C,

But may produce unexpected output when run

| when we try to access the array variables – say arr[5] ;

Example

```
// This C program compiles fine
// as index out of bound
// is not checked in C.

#include <stdio.h>

int main()
{
    int arr[2];

    printf("%d ", arr[3]);
    printf("%d ", arr[-2]);

    return 0;
}
```

Example

Garbage values

2008101287 4195777

How to initialize an array?

Initialize an array during declaration,

int mark[5] ;

How to initialize an array?

Initialize an array during declaration,

```
int mark[5] = {19, 10, 8, 17, 9};
```

How to initialize an array?

Also possible –

```
int mark[] = {19, 10, 8, 17, 9};
```

How to initialize an array?

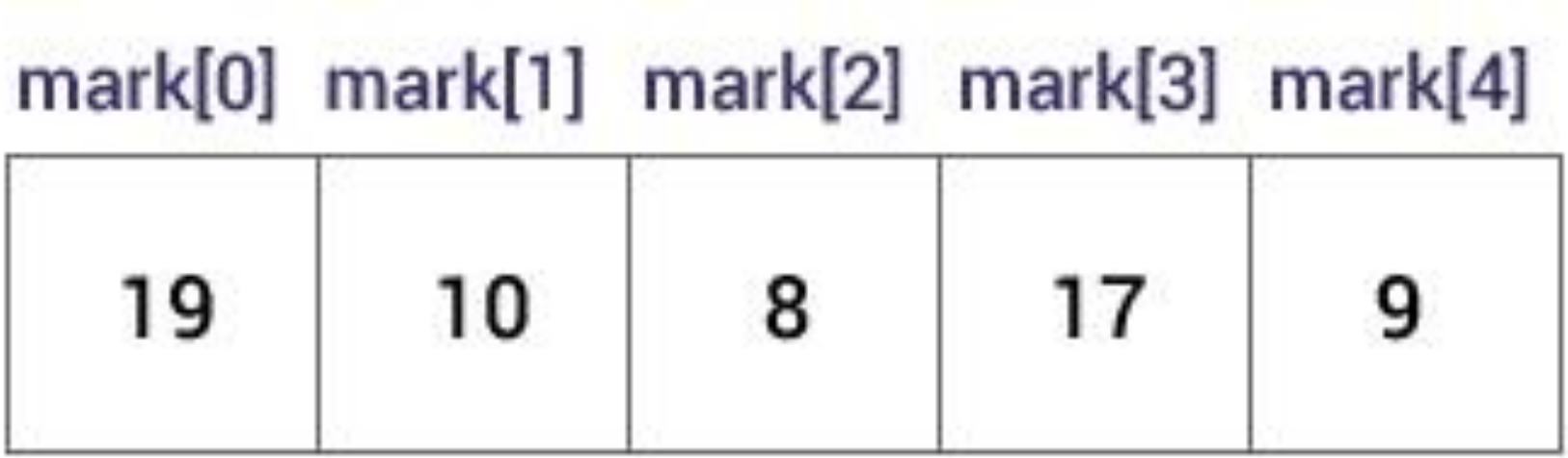
Also possible –

```
int mark[] = {19, 10, 8, 17, 9};
```

Here, we haven't specified the size

Compiler knows its size is 5 as we are initializing it with 5 elements

How to initialize an array?



Boundary issue

In C, it is not compiler error to initialize an array with more elements than the specified size.

Boundary issue

Program compiles fine and shows just Warning

```
#include <stdio.h>
int main()
{

    // Array declaration by initializing it with more
    // elements than specified size.
    int arr[2] = { 10, 20, 30, 40, 50 };

    return 0;
}
```

Boundary issue

Warnings:

```
prog.c: In function 'main':
prog.c:7:25: warning: excess elements in array initializer
  int arr[2] = { 10, 20, 30, 40, 50 };
                         ^
prog.c:7:25: note: (near initialization for 'arr')
prog.c:7:29: warning: excess elements in array initializer
  int arr[2] = { 10, 20, 30, 40, 50 };
                             ^
prog.c:7:29: note: (near initialization for 'arr')
prog.c:7:33: warning: excess elements in array initializer
  int arr[2] = { 10, 20, 30, 40, 50 };
                                 ^
prog.c:7:33: note: (near initialization for 'arr')
```

Changing the values

```
int mark[5] = {19, 10, 8, 17, 9}
// make the value of the third element to -1
mark[2] = -1;
// make the value of the fifth element to 0
mark[4] = 0;
```

Input and Output Array Elements

```
// take input and store it in the 3rd element
scanf("%d", &mark[2]);

// take input and store it in the ith element
scanf("%d", &mark[i-1]);
```

Input and Output Array Elements

```
// print the first element of the array
printf("%d", mark[0]);
// print the third element of the array
printf("%d", mark[2]);
// print ith element of the array
printf("%d", mark[i-1]);
```

Example

WAP to take 5 values from the user and store them in an array

Print the elements stored in the array

Example

```
#include <stdio.h>
int main() {
  int values[5];
  printf("Enter 5 integers: ");
```

Example

```
#include <stdio.h>
int main() {
  int values[5];
  printf("Enter 5 integers: ");
  // taking input and storing it in an array
  for(int i = 0; i < 5; ++i) {
     scanf("%d", &values[i]);
  }
```

```
#include <stdio.h>
int main() {
  int values[5];

  printf("Enter 5 integers: ");
//-------------------------------------------------------------------
  // taking input and storing it in an array
  for(int i = 0; i < 5; ++i) {
     scanf("%d", &values[i]);
  }
//-------------------------------------------------------------------

  printf("Displaying integers: ");
  // printing elements of an array
  for(int i = 0; i < 5; ++i) {
     printf("%d\n", values[i]);
  }
  return 0;
}
```

```c
#include <stdio.h>
int main() {
  int values[5];
  printf("Enter 5 integers: ");
  // taking input and storing it in an array
  for(int i = 0; i < 5; ++i) {
    scanf("%d", &values[i]);
  }
  printf("Displaying integers: ");
  // printing elements of an array
  for(int i = 0; i < 5; ++i) {
    printf("%d\n", values[i]);
  }
  return 0;
}
```

Output:

```
Enter 5 integers: 1
-3
34
0
3
Displaying integers: 1
-3
34
0
3
```

Example 2: Calculate Average

Take n numbers from the user

Print the average of these n numbers

What is the number of numbers user wants to do computation with ??

Average

```
#include <stdio.h>
 # define LIMIT 100
int main()
{
    int marks[LIMIT];

   //
int i=5;
   marks[i];
   marks[100];
```

XX Accepted.

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
```

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
```

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
```

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
    {

    }
}
```

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
    {
        printf("Enter number%d: ",i+1);
        scanf("%d", &marks[i]);

    }
```

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
    {
        printf("Enter number%d: ",i+1);
        scanf("%d", &marks[i]);

        // adding integers entered by the user to the sum
variable
        sum += marks[i];
    }
}
```

Average

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
    {
        printf("Enter number%d: ",i+1);
        scanf("%d", &marks[i]);

        // adding integers entered by the user to the sum variable
        sum += marks[i];
    }
    average = sum/n;
    printf("Average = %d", average);
    return 0;
}
```

```
#include <stdio.h>
int main()
{
    int marks[10], i, n, sum = 0, average;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
    {
        printf("Enter number%d: ",i+1);
        scanf("%d", &marks[i]);

        // adding integers entered by the
user to the sum variable
        sum += marks[i];
    }
    average = sum/n;
    printf("Average = %d", average);
    return 0;
}
```

```
Enter n: 5
Enter number1: 45
Enter number2: 35
Enter number3: 38
Enter number4: 31
Enter number5: 49
Average = 39
```

Float / double array

```
double balance[5] = {1000.0, 2.0, 3.4, 7.0, 50.0};
```

Nested loops and Multi-dimensional arrays

Certain problems directly maps to a 2- or higher dimensional objects

2D plane

Matrix

Multi-dimensional data

etc

Nested loops and Multi-dimensional arrays

A Single dimensional Array is quite enough to represent all these multi-dimensional objects

How ?

Represent a matrix of size 100 X 100 using a 1 D array

Nested loops and Multi-dimensional arrays

Problem –

Store and reproduce a matrix -

2-D Arrays

float x[3][4]; --- // float y[12]

The array can hold 12 elements;

What is the name ?
What are the elements of this array?
How to access the elements ?

2-D Arrays

```
int a;
int b[10];
int c[6][10];
int d[5][7][10];
```

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

3-D array

```
float y[2][3][4];
```

Here, the array y can hold 24 elements total

y[0][3][4];

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

y[1][3][4];

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

How to initialize ?

```
int c[2][3] = {{1, 3, 0}, {-1, 5, 9}};

int c[][3] = {{1, 3, 0}, {-1, 5, 9}};

int c[2][3] = {1, 3, 0, -1, 5, 9};
```

1, 3, 0

-1, 5, 9

int C[2][3] = {{1,3}, {0,-1}, {5,9}};

Compilation error ?

If not – then are they stored properly ?

How to initialize ?

```
int c[2][3] = {{1, 3, 0}, {-1, 5, 9}};

int c[][3] = {{1, 3, 0}, {-1, 5, 9}};

int c[2][3] = {1, 3, 0, -1, 5, 9};
```

How to initialize

```
int test[2][3][4] = {
    {     {3, 4, 2, 3},
          {0, -3, 9, 11},
          {23, 12, 23, 2}
    },
     {    {13, 4, 56, 3},
          {5, 9, 3, 5},
          {3, 1, 4, 9}
     }
};
```

Problems

Store temperatures of two cities for a week and print

What are the possible solutions ?

```
#include <stdio.h>
const int CITY = 2;
const int WEEK = 7;
```

```
#include <stdio.h>
const int CITY = 2;
const int WEEK = 7;
int main()
{
  int temperature[CITY][WEEK];
```

```
#include <stdio.h>
const int CITY = 2;
const int WEEK = 7;
int main()
{
  int temperature[CITY][WEEK], i, j;
  // Using nested loop to store values in a 2d array
  for (i = 0; i < CITY; ++i)
  {
    for (j = 0; j < WEEK; ++j)
    {
    }
  }
```

```
#include <stdio.h>
const int CITY = 2;
const int WEEK = 7;
int main()
{
  int temperature[CITY][WEEK];
  // Using nested loop to store values in a 2d array
  for (i = 0; i < CITY; ++i)
  {
    for (j = 0; j < WEEK; ++j)
    {
      printf("City %d, Day %d: ", i + 1, j + 1);
      scanf("%d", &temperature[i][j]);
    }
  }
```

```
#include <stdio.h>
const int CITY = 2;
const int WEEK = 7;
int main()
{
  int temperature[CITY][WEEK];
  // Using nested loop to store values in a 2d array
 for (j = 0; j < WEEK; ++j)
{
            for (i = 0; i < CITY; ++i)
             {
                 printf(“Day %d, City %d: ", j + 1, i + 1);
            scanf("%d", &temperature[i][j]);
    }
  }
```

```
printf("\nDisplaying values: \n\n");
  // Using nested loop to display vlues of a 2d array
  for (int i = 0; i < CITY; ++i)
  {
    for (int j = 0; j < WEEK; ++j)
    {
      printf("City %d, Day %d = %d\n", i + 1, j + 1,
temperature[i][j]);
    }
  }
  return 0;
}
```

City 1, Day 1: 33
City 1, Day 2: 34
City 1, Day 3: 35
City 1, Day 4: 33
City 1, Day 5: 32
City 1, Day 6: 31
City 1, Day 7: 30
City 2, Day 1: 23
City 2, Day 2: 22
City 2, Day 3: 21
City 2, Day 4: 24
City 2, Day 5: 22
City 2, Day 6: 25
City 2, Day 7: 26

Displaying values:

City 1, Day 1 = 33
City 1, Day 2 = 34
City 1, Day 3 = 35
City 1, Day 4 = 33
City 1, Day 5 = 32
City 1, Day 6 = 31
City 1, Day 7 = 30
City 2, Day 1 = 23
City 2, Day 2 = 22
City 2, Day 3 = 21
City 2, Day 4 = 24
City 2, Day 5 = 22
City 2, Day 6 = 25
City 2, Day 7 = 26

Matrix : Input and sum

```
#include <stdio.h>
int main()
{
  float a[2][2], b[2][2], result[2][2];
```

```
#include <stdio.h>
int main()
{
  float a[2][2], b[2][2], result[2][2];
  // Taking input using nested for loop
  printf("Enter elements of 1st matrix\n");
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
    {
      printf("Enter a%d%d: ", i + 1, j + 1);
      scanf("%f", &a[i][j]);
    }
```

```
// Taking input using nested for loop
  printf("Enter elements of 2nd matrix\n");
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
    {
      printf("Enter b%d%d: ", i + 1, j + 1);
      scanf("%f", &b[i][j]);
    }
```

Matrix : Input and sum

```
// adding corresponding elements of two arrays
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
    {
      result[i][j] = a[i][j] + b[i][j];
    }
```

```
// Displaying the sum
  printf("\nSum Of Matrix:");
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
    {
      printf("%.1f\t", result[i][j]);
      if (j == 1)
        printf("\n");
    }
  return 0;
}
```

Enter elements of 1st matrix
Enter a11: 2;
Enter a12: 0.5;
Enter a21: -1.1;
Enter a22: 2;
Enter elements of 2nd matrix
Enter b11: 0.2;
Enter b12: 0;
Enter b21: 0.23;
Enter b22: 23;

Sum Of Matrix:
2.2 0.5
-0.9 25.0

Sample programing problems

Print diagonal of a matrix

Exchange the diagonals of a matrix

Sum of the two diagonals

Transpose of a matrix ….

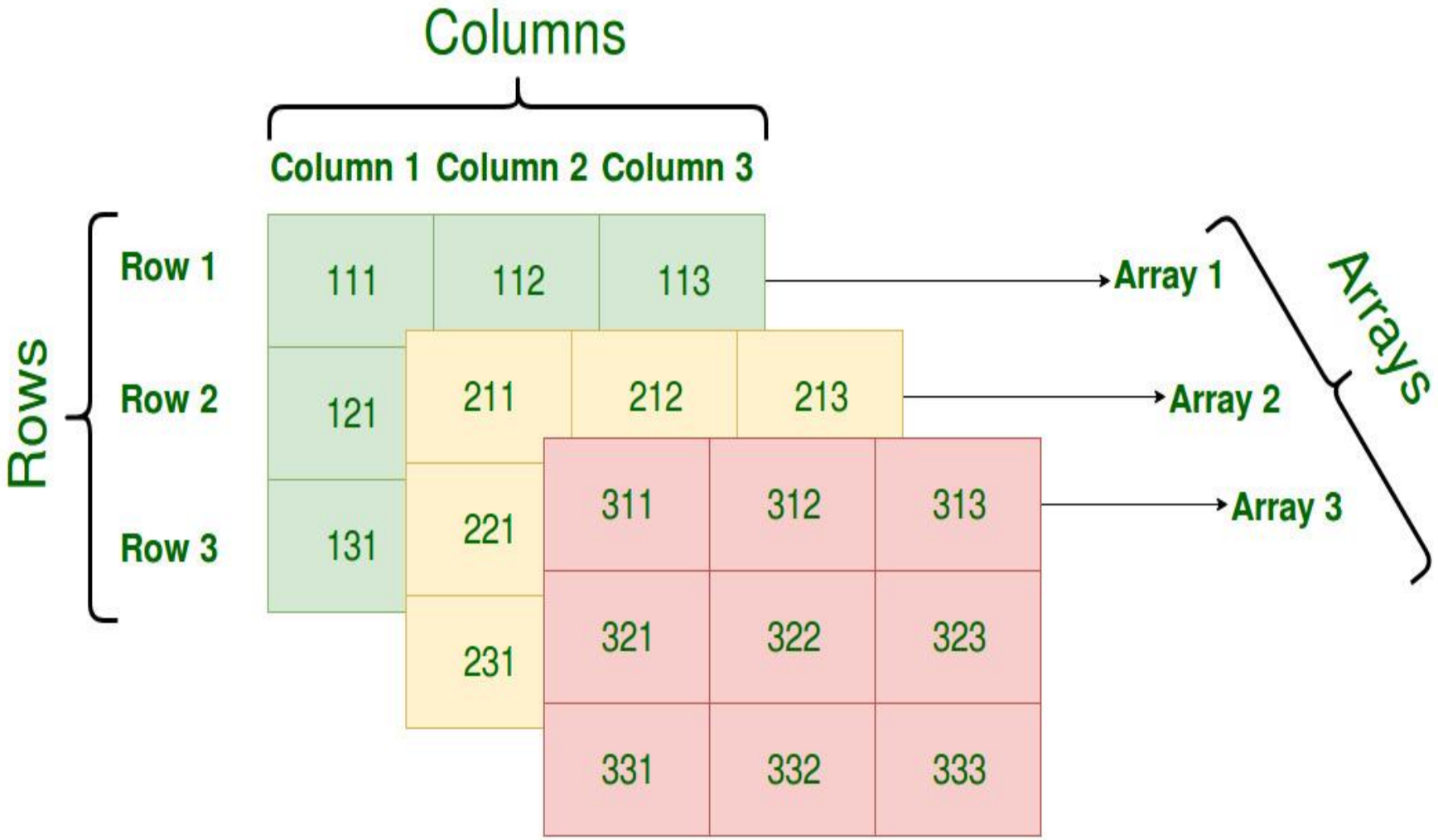
Generalization

data_type array_name[size1][size2]....[sizeN];

int two_d[10][20];

int three_d[10][20][30];

int three_d[3][3][3];



Multiply two matrices …. hints

```
 for (i = 0; i < r1; ++i) {
        for (j = 0; j < c2; ++j) {
            mult[i][j] = 0;
        }
    }
for (i = 0; i < r1; ++i) {
        for (j = 0; j < c2; ++j) {
            for (k = 0; k < c1; ++k) {
                mult[i][j] += first[i][k] * second[k][j];
            }
        }
    }
```