Object-Oriented Programming in C++

Arrays and Vectors - 2

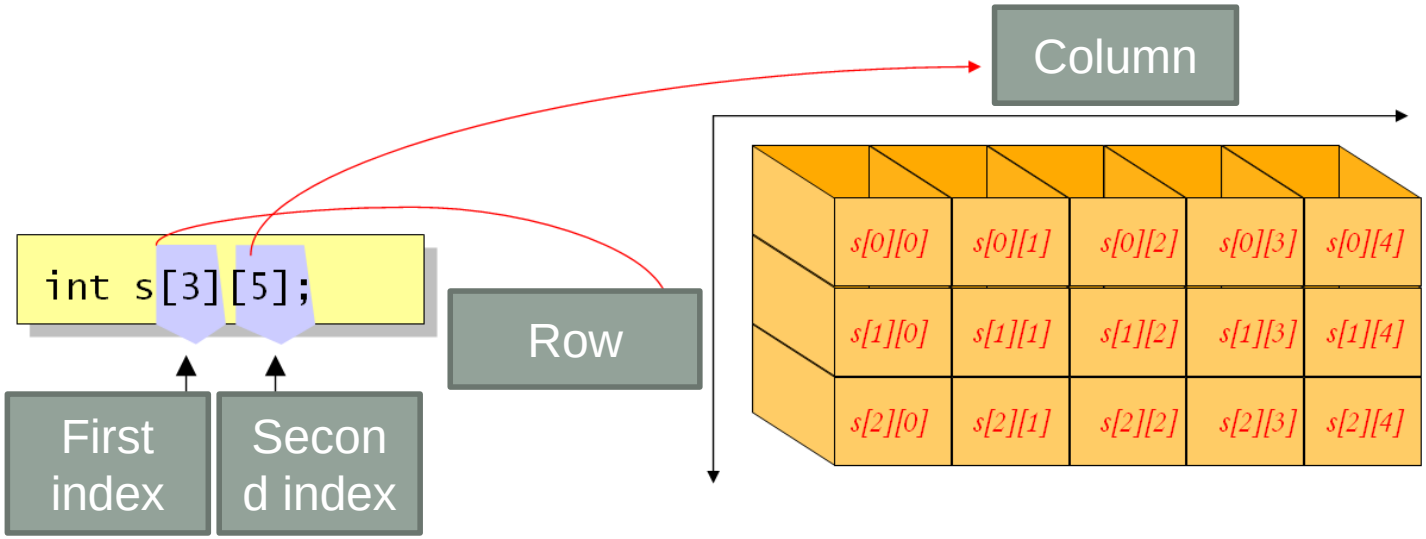
Professor: 최학남, xncui@inha.ac.kr
Office: high-tech 401

Contents

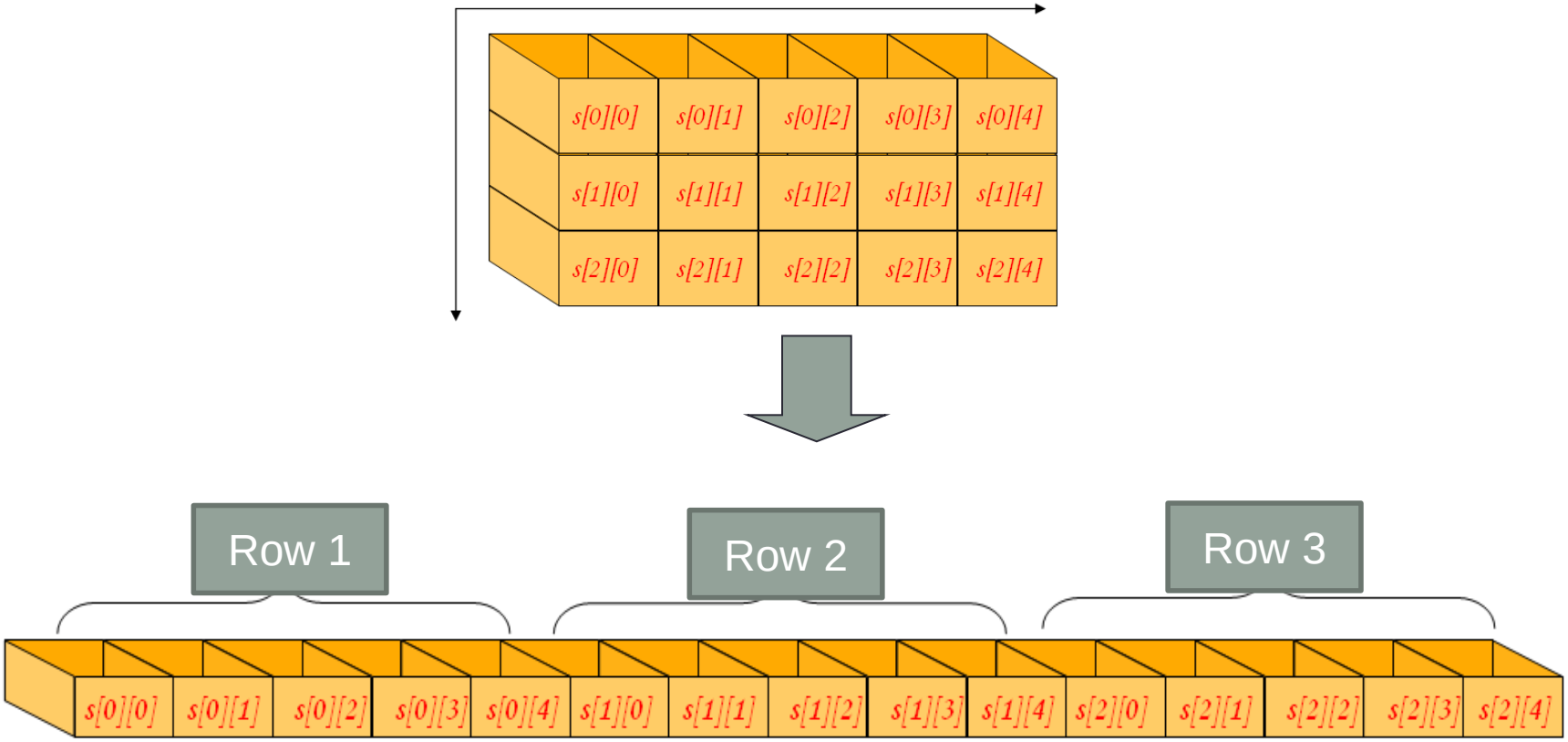
- What are arrays
- A simple program using array
  - ✓ Array declaration
  - ✓ Array initialization
- 1D-Array
- 2D-Array
- ND-Array

2D Array

```
int s[10];      // 1D Array
int s[3][10];   // 2D Array
int s[5][3][10]; // 3D Array
```

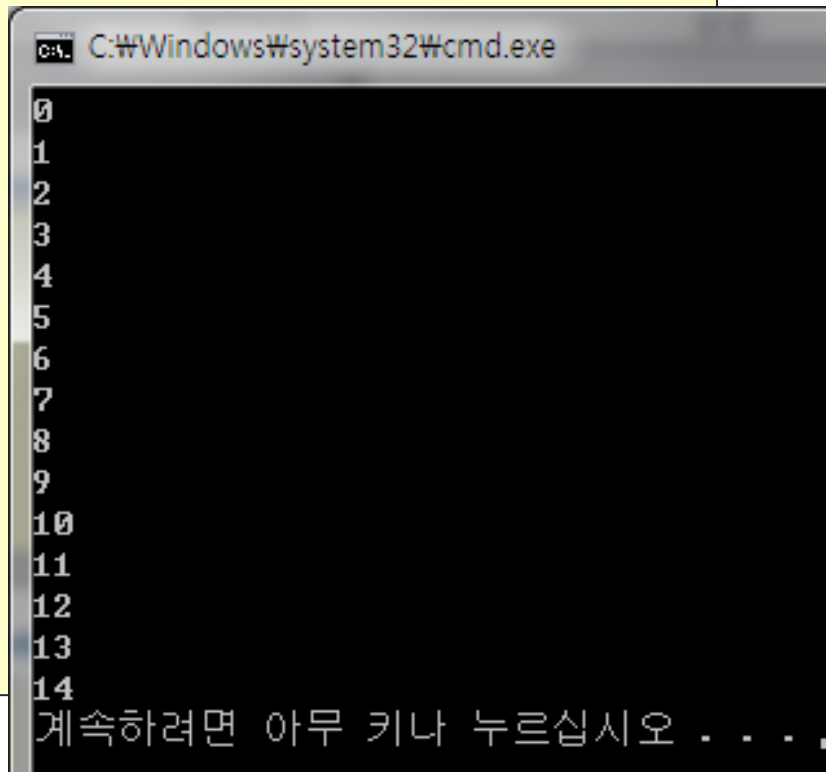


2D Array


s[0][0] s[0][1] s[0][2] s[0][3] s[0][4]
s[1][0] s[1][1] s[1][2] s[1][3] s[1][4]
s[2][0] s[2][1] s[2][2] s[2][3] s[2][4]
Row 1
Row 2
Row 3
s[0][0] s[0][1] s[0][2] s[0][3] s[0][4] s[1][0] s[1][1] s[1][2] s[1][3] s[1][4] s[2][0] s[2][1] s[2][2] s[2][3] s[2][4]

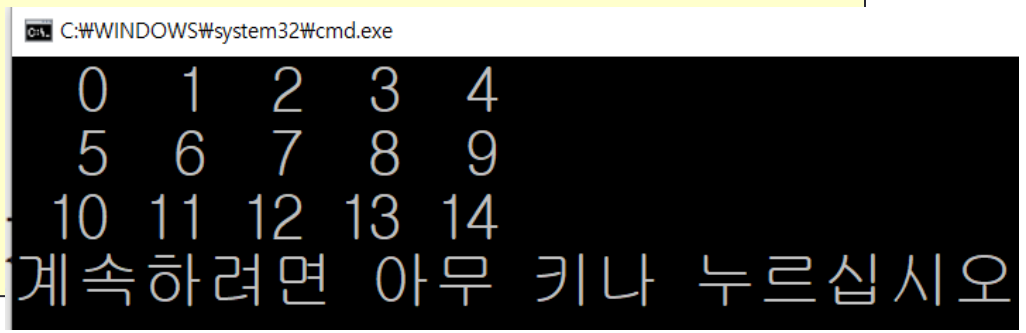
EX #11-1 2D Array

```c
#include <stdio.h>

int main(void)
{
	int s[3][5];		// declaration of 2D array
	int i, j;		// for array index
	int value = 0;

	for(i=0;i<3;i++)
		for(j=0;j<5;j++)
			s[i][j] = value++;

	for(i=0;i<3;i++)
		for(j=0;j<5;j++)
			printf("%d\n", s[i][j]);
	return 0;
}
```



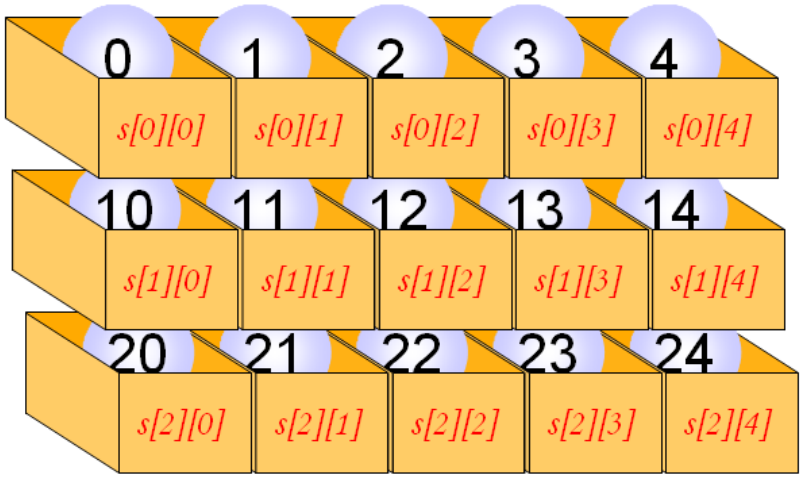
EX #11-2 2D Array

```
#include <stdio.h>

int main(void)
{
        int s[3][5];        // declaration of 2D array
        int i, j;           // for array index
        int value = 0;

        for(i=0;i<3;i++)
                for(j=0;j<5;j++)
                        s[i][j] = value++;

        for(i=0;i<3;i++)  {
                for(j=0;j<5;j++)    {
                        printf("%3d", s[i][j]);
                }
         printf("\n");
       }
        return 0;
}
```

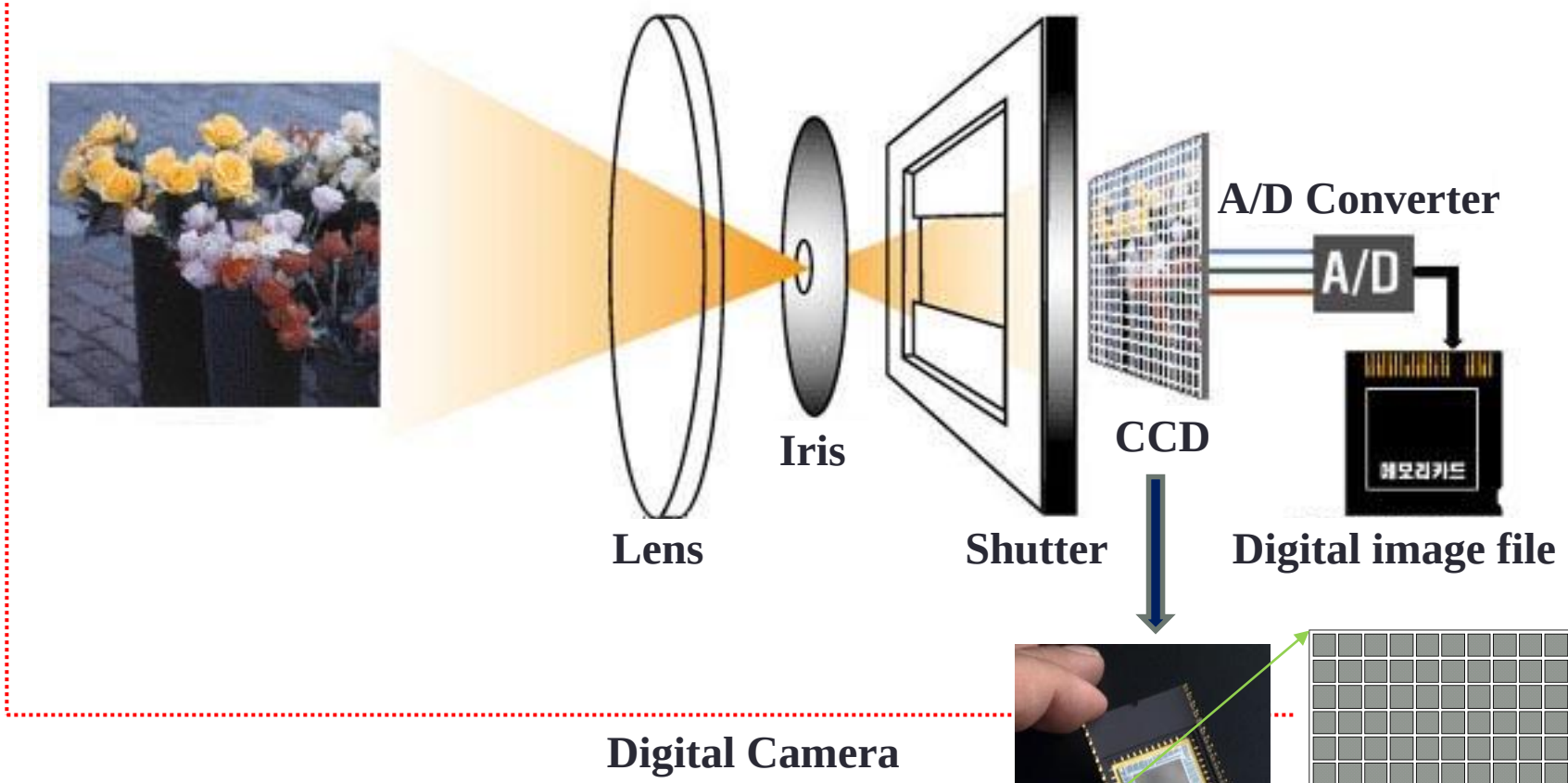


2D Array initialization

```
int s[3][5] = {
  { 0, 1, 2, 3, 4 }, // first row
  { 10, 11, 12, 13, 14 }, // second row
  { 20, 21, 22, 23, 24 } // third row
};
```

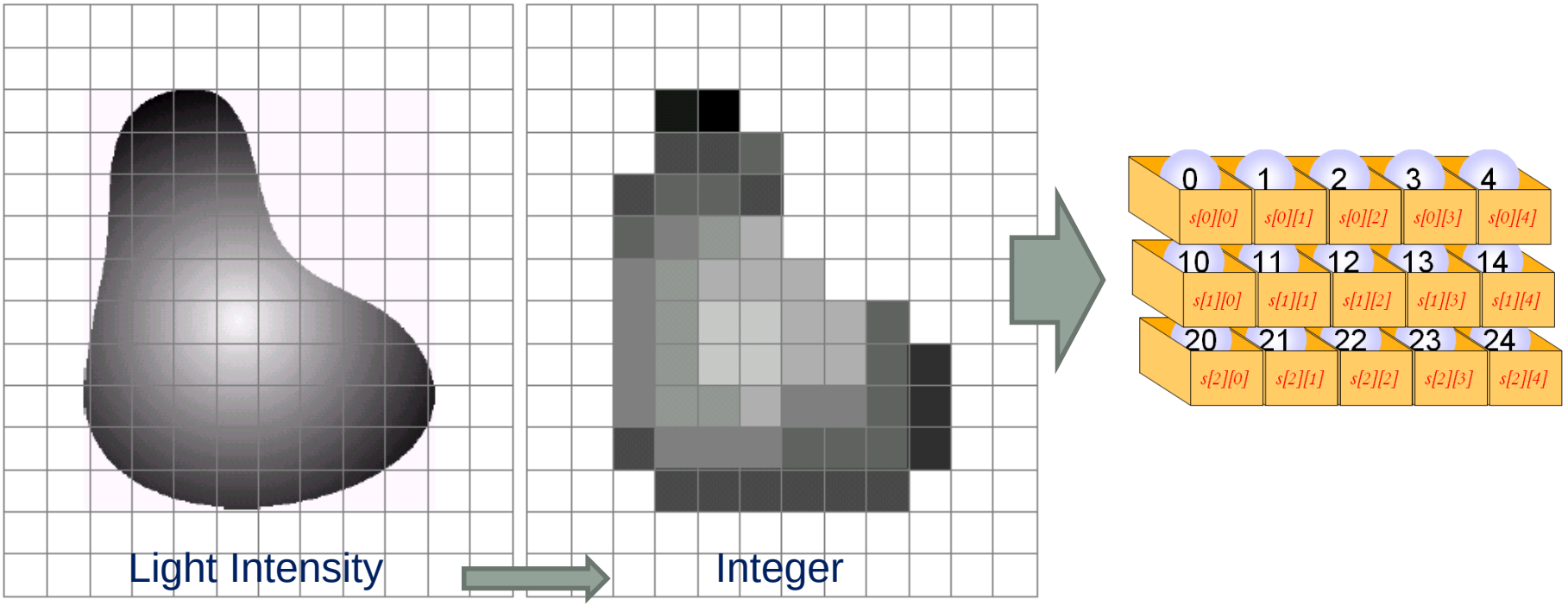


Digital Camera System



Digital Image

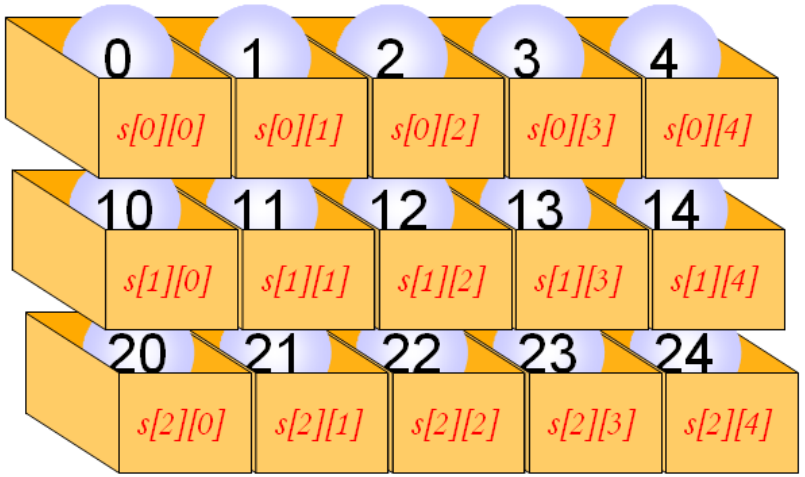
Natural Scene **B/W(or Gray) Image**



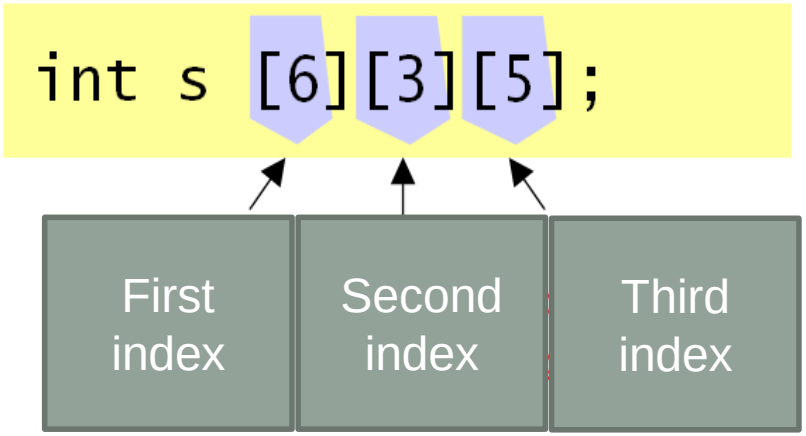
Sampling **Quantization**

2D Array initialization

```
int s[ ][5] = {
  { 0, 1, 2, 3, 4 }, // first row
  { 10, 11, 12, 13, 14 }, // second row
  { 20, 21, 22, 23, 24 } // third row
};
```



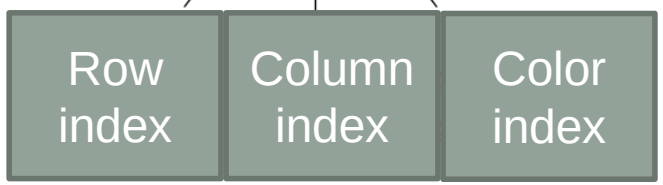
EX 12: 3D Array

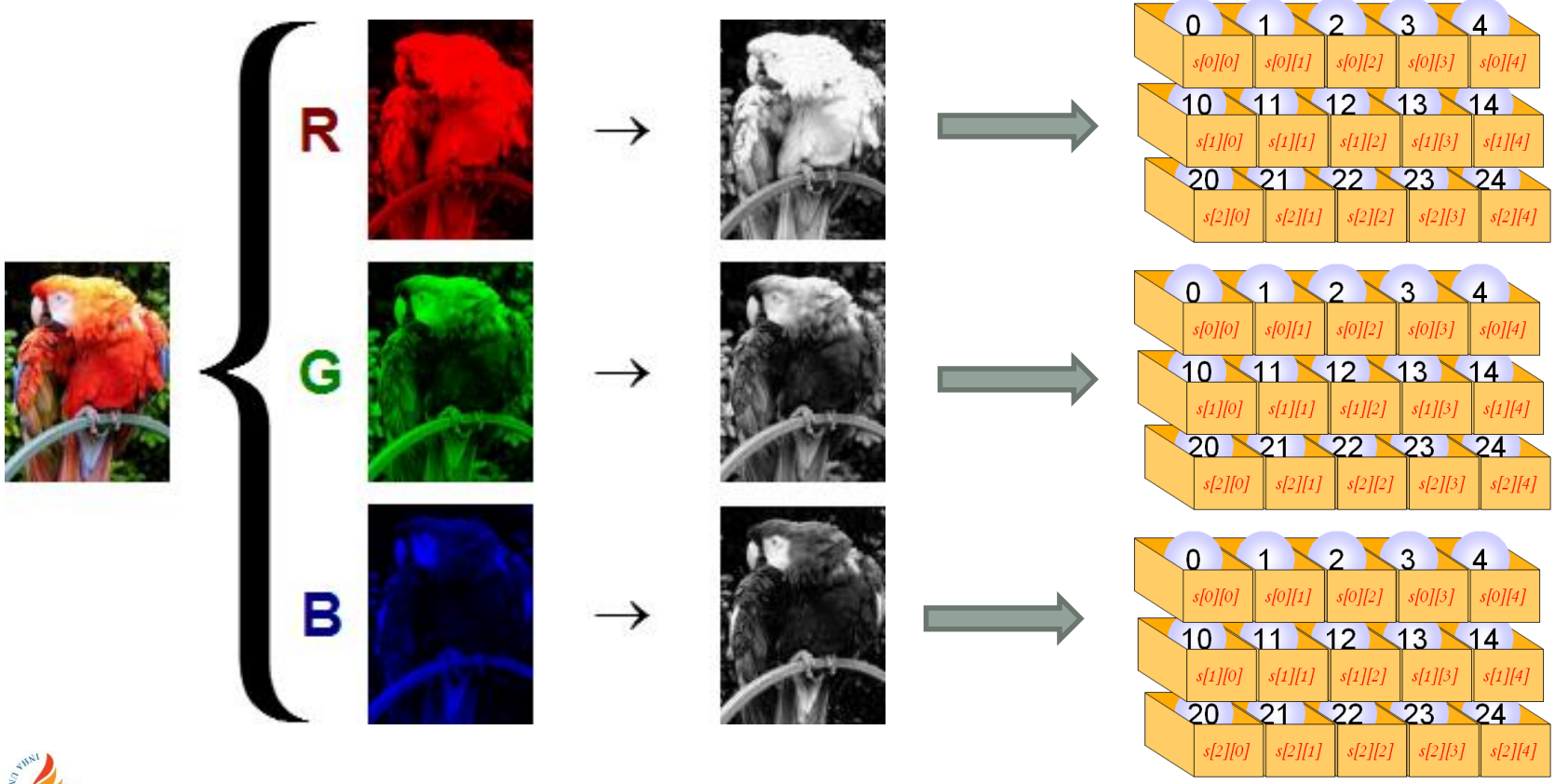



```
#include<iostream>
using namespace std;
int main(void)
{
    int s[6][3][5]={0};
    int x, y, z;
    int i = 1;

    for (z = 0; z < 6; z++) {
        for (y = 0; y < 3; y++) {
            for (x = 0; x < 5; x++) {
                s[z][y][x] = i++;
                cout << s[z][y][x] << " ";
            }
            cout << endl;
        }
        cout << endl;
    }

    return 0;
}
```

Color Image

```
int s [6][3][5];
```





Ex 13: Reverse

```c
#include <stdio.h>
#define SIZE 5

int main(void)
{
    int data[SIZE];
    int i;

    for(i = 0; i < SIZE; i++)       // 정수를 입력받는 루프
    {
        printf("정수를 입력하시오:");
        scanf("%d", &data[i]);
    }

    for(i = SIZE - 1;i >= 0; i--)   // 역순으로 출력하는 루프
    {
        printf("%d\n", data[i]);
    }

    return 0;
}
```

Ex 14: frequency

```c
#include <stdio.h>
#define SIZE 101

int main(void)
{
    int freq[SIZE];
    int i, score;

    for(i = 0; i < SIZE; i++)
        freq[i] = 0;

    while(1)
    {
        printf("숫자를 입력하시오(종료-1): ");
        scanf("%d", &score);
        if (score < 0) break;
        freq[score]++;
    }

    printf("값   빈도\n");

    for(i = 0; i < SIZE; i++)
        printf("%3d     %3d \n", i, freq[i]);

    return 0;
}
```

Ex 15: dice

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

#define SIZE 6

int main(void)
{
    int freq[SIZE] = { 0 };      // 주사위의 면의 빈도를 0으로 한다.
    int i;

    for(i = 0; i < 10000; i++)     // 주사위를 10000번 던진다.
        ++freq[ rand() % 6 ];  // 해당면의 빈도를 하나 증가한다.

    printf("====================\n");
    printf("면      빈도\n");
    printf("====================\n");

    for(i = 0; i < SIZE; i++)
        printf("%3d      %3d \n", i, freq[i]);

    return 0;
}
```

Ex 16: sorting - selection sort

```c
#include <stdio.h>
#define SIZE 10

void selection_sort(int list[], int n);
void print_list(int list[], int n);

int main(void)
{
    int grade[SIZE] = { 3, 2, 9, 7, 1, 4, 8, 0, 6, 5 };

    // 원래의 배열 출력
    printf("원래의 배열\n");
    print_list(grade, SIZE);

    selection_sort(grade, SIZE);

    // 정렬된 배열 출력
    printf("정렬된 배열\n");
    print_list(grade, SIZE);

    return 0;
}
```

```c
void print_list(int list[], int n)
{
    int i;
    for(i = 0;i < n; i++)
        printf("%d ", list[i]);
    printf("\n");
}

void selection_sort(int list[], int n)
{
    int i, j, temp, least;

    for(i = 0; i < n-1; i++)
    {
        least = i;

        for(j = i + 1; j < n; j++)    // 최소값 탐색
            if(list[j] < list[least])
                least = j;
        // i번째 원소와 least 위치의 원소를 교환
        temp = list[i];
        list[i] = list[least];
        list[least] = temp;
    }
}
```

Ex 17: search - sequential search

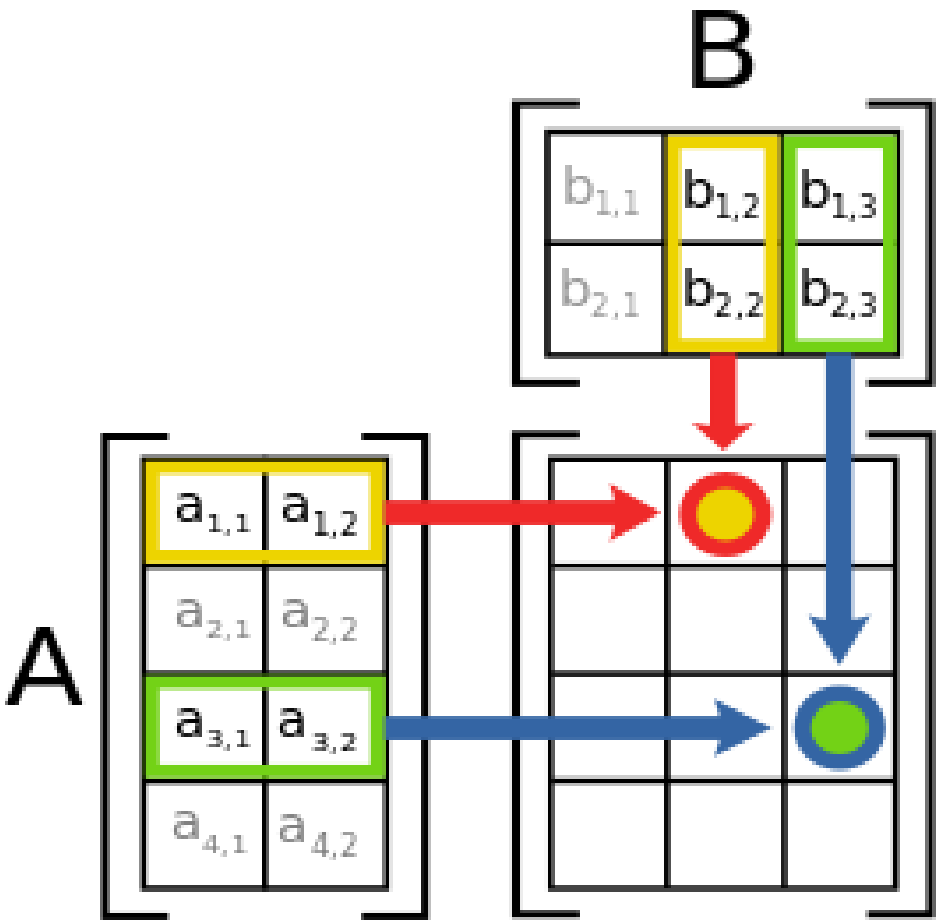
```c
#include <stdio.h>
#define SIZE 10

int seq_search(int list[], int n, int key);

int main(void)
{
    int key;
    int grade[SIZE] = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

    printf("탐색할 값을 입력하시오:");
    scanf("%d", &key);
    printf("탐색 결과 = %d\n", seq_search(grade, SIZE, key));

    return 0;
}

int seq_search(int list[], int n, int key)
{
    int i;

    for(i = 0; i < SIZE; i++)
        if(list[i] == key)
            return i; // 탐색이 성공하면 인덱스 반환
    return -1;          // 탐색이 실패하면 -1 반환
}
```

Ex 18: search - binary search

```c
#include <stdio.h>
#define SIZE 10
int binary_search(int list[], int n, int key);

int main(void)
{
    int key;
    int grade[SIZE] = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

    printf("탐색할 값을 입력하시오:");
    scanf("%d", &key);
    printf("탐색 결과= %d\n", binary_search(grade, SIZE, key));

    return 0;
}
```

```c
int binary_search(int list[], int n, int key)
{
    int low, high, middle;

    low = 0;
    high = n-1;

    while( low <= high ){      // 아직 숫자들이 남아있으면
        middle = (low + high)/2;      // 중간 요소 결정
        if( key == list[middle] )   // 일치하면 탐색 성공
            return middle;
        else if( key > list[middle] )// 중간 원소보다 크다면
            low = middle + 1;   // 새로운 값으로 low 설정
        else
            high = middle - 1;   // 새로운 값으로 high 설정
    }
    return -1;
}
```

Matrix multiplication (행렬곱)



https://en.wikipedia.org/wiki/File:Matrix_multiplication_diagram_2.svg

Matrix multiplication (행렬곱)

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

$$\mathbf{C = AB}$$

$$\mathbf{C} = \begin{pmatrix} a_{11}b_{11} + \cdots + a_{1n}b_{n1} & a_{11}b_{12} + \cdots + a_{1n}b_{n2} & \cdots & a_{11}b_{1p} + \cdots + a_{1n}b_{np} \\ a_{21}b_{11} + \cdots + a_{2n}b_{n1} & a_{21}b_{12} + \cdots + a_{2n}b_{n2} & \cdots & a_{21}b_{1p} + \cdots + a_{2n}b_{np} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{11} + \cdots + a_{mn}b_{n1} & a_{m1}b_{12} + \cdots + a_{mn}b_{n2} & \cdots & a_{m1}b_{1p} + \cdots + a_{mn}b_{np} \end{pmatrix}$$

https://en.wikipedia.org/wiki/File:Matrix_multiplication_diagram_2.svg

EX #19 Matrix expression using array

```
#include <stdio.h>
#define ROWS 3
#define COLS 3
int main(void)
{
      int A[ROWS][COLS] = {  { 2,3,0 }, { 8,9,1 { 7,0,5 } };
      int B[ROWS][COLS] = {  { 1,0,0 },  { 1,0,0   { 1,0,0 }};
      int C[ROWS][COLS];
      int r,c;
      // matrix addition.
      for(r = 0;r < ROWS; r++)
            for(c = 0;c < COLS; c++)
                  C[r][c] = A[r][c] + B[r][c];
      // print the results.
      for(r = 0;r < ROWS; r++)
      {
            for(c = 0;c < COLS; c++)
                  printf("%d ", C[r][c]);
            printf("\n");
      }
       // matrix multiplication.
       int D[ROWS][COLS] = { 0 };
        //add matrix multiplication code here


      // print the results.
      for(r = 0;r < ROWS; r++)
      {
            for(c = 0;c < COLS; c++)
                  printf("%d ", D[r][c]);
            printf("\n");
      }
      return 0;
}
```

$$A = \begin{bmatrix} 2 & 3 & 0 \\ 8 & 9 & 1 \\ 7 & 0 & 5 \end{bmatrix}$$

```
3 3 0
9 9 1
8 0 5
```

```
5 0 0
18 0 0
12 0 0
```

HW #7

Declare a 20x20 array and provide following services. Use as many functions as possible.

1. fill: fill the array using rand()%100
2. show: show the array
3. row sum: read a row number and show the sum of that row
4. col sum: read a column number and show the sum of that column
5. diagonal sum: show the diagonal sum
6. backward diagonal sum: show the backward diagonal sum
7. row max: read a row number and show the max of that row
8. col max: read a column number and show the max of that column
9. max row: find the row whose sum is the greatest
10. max col: find the column whose sum is the greatest

Project #1

1. For the following set of n data points (x, y), compute the correlation coefficient r, given by

$$r = \frac{n\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

x	y
34.22	102.43
39.87	100.93
41.85	97.43
43.23	97.81
40.06	98.32
53.29	98.32
53.29	100.07
54.14	97.08
49.12	91.59
40.71	94.85
55.15	94.65

Have to use the function and array

1. Compute the **correlation coefficient r**
2. Draw the function call structures of the developed source code.

Project #1

2. For the following set of point given by **(x, y)** fit a straight line given by

$$y = a + bx$$

where,

$$a = \bar{y} - b\bar{x} \quad \text{and}$$

$$b = \frac{n\sum yx - \sum x \sum y}{[n\sum x^2 - (\sum x)^2]}$$

x	y
3.0	1.5
4.5	2.0
5.5	3.5
6.5	5.0
7.5	6.0
8.5	7.5
8.0	9.0
9.0	10.5
9.5	12.0
10.0	14.0

Have to use the function and array

1. Compute the **a** and **b**
2. Draw the function call structures of the developed source code.

Deadline: two weeks from today (refer the e-class)