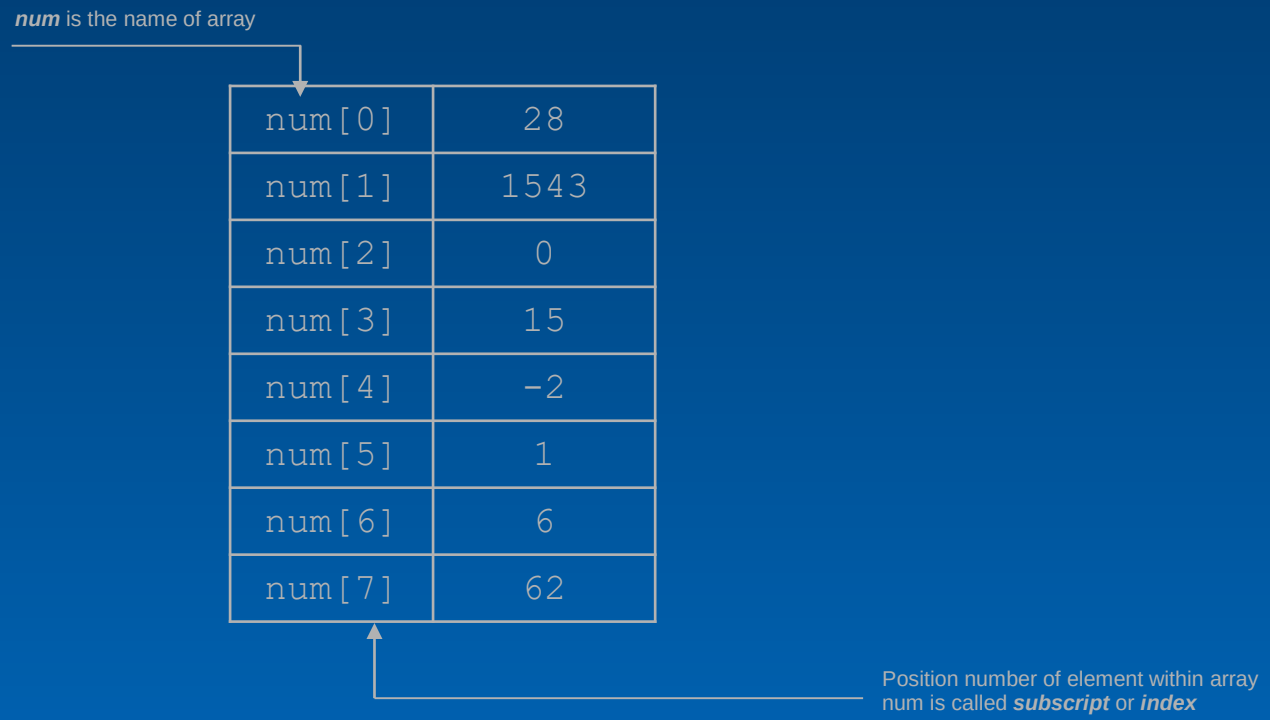


C++

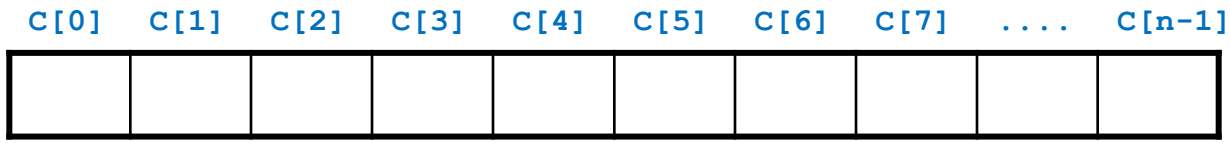
Introduction

DATA STRUCTURE - ARRAYS

05 ARRAYS

What is an Array?

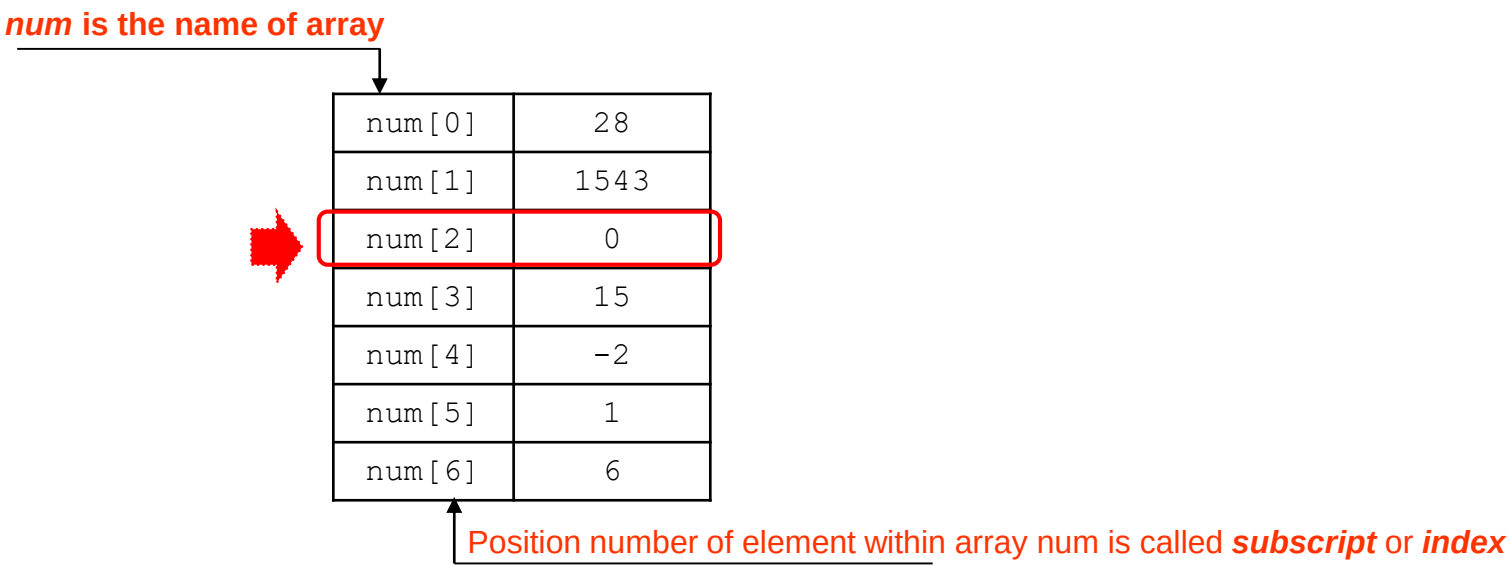
- A data structure that is used to store a group of objects of the same type sequentially in memory.
- Suppose an array named *C* contains *n* elements.
  - This array is indexed from 0 to (n-1).



  - The first element of *array C* is at index *0* and is accessed as *C[0]* .
  - The second element is at index *1* and is accessed as *C[1]* .
  - The last element is at index *(n-1)* and is accessed as *C[n-1]*

Exploring an Array

- Suppose: int num[] = { 28, 1543, 0, 15, -2, 1, 6};

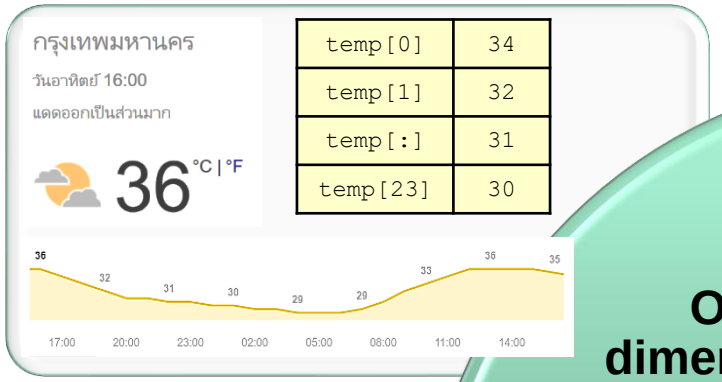


- To print the values contained in the third element of array *num*

```
Console.WriteLine("The third element is " +          + ".");
```

```
The third element is 0.
```

Types of Arrays

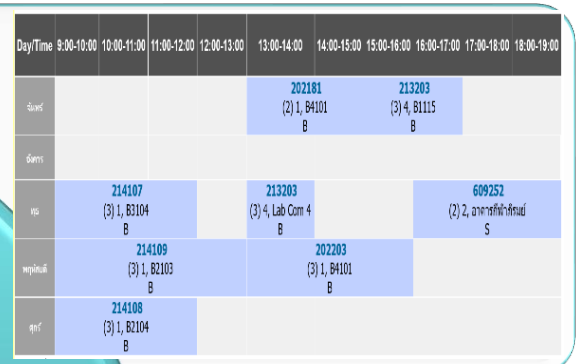


temp[0]	34
temp[1]	32
temp[:]	31
temp[23]	30

Today Weather

One-dimensional Arrays

Two-dimensional Arrays

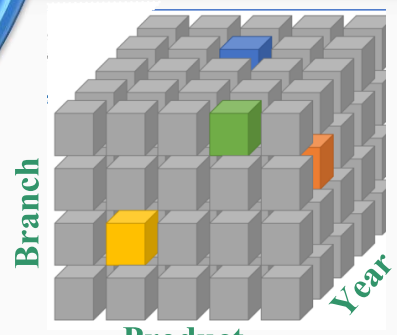


Class Schedule

Multi-dimensional Arrays

Three-dimensional Arrays

https://theresasreviews.com/4dx-movie-theater-washington-dc-giveaway/



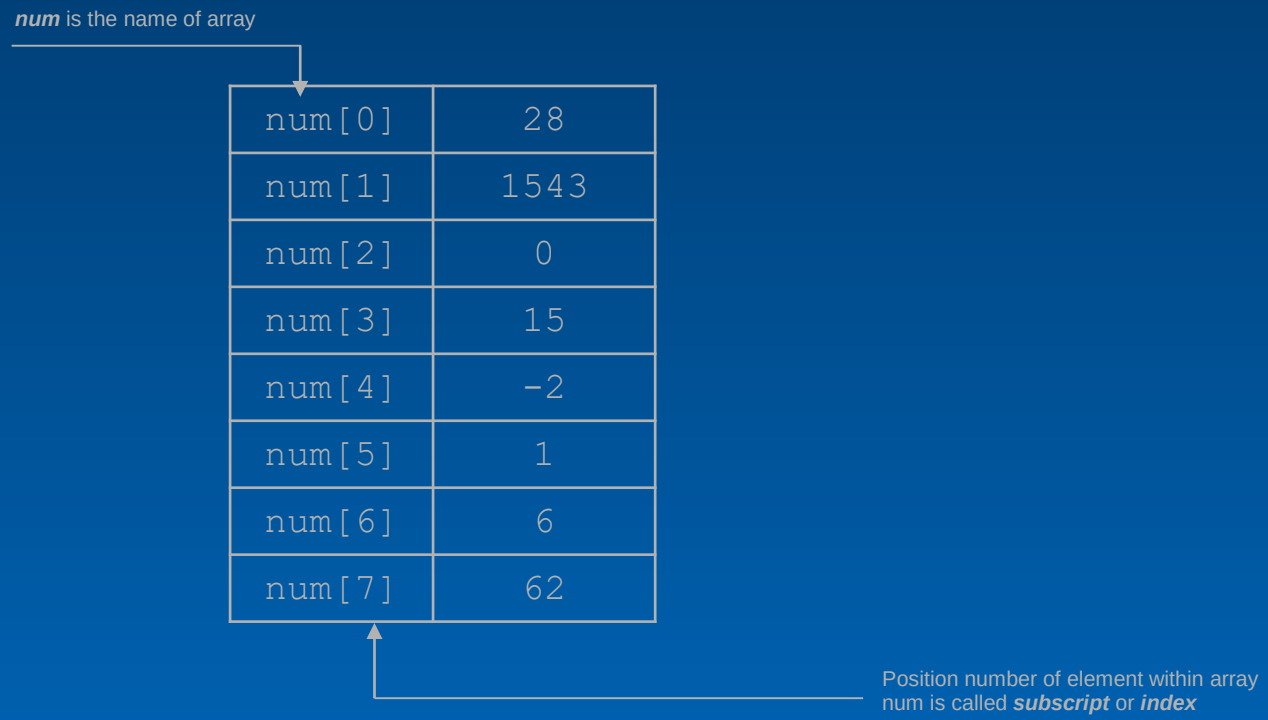
Arrays

I. Arrays of Integer — 1, 2, 100, 999

II. Arrays of Float — 1.9, 3.2, 50.0, 99.99

III. Arrays of Characters — s, t, d, #, ?

IV. Arrays of String — Student#1, of, Digital, Technology



C++

One-Dimensional Arrays
ARRAYS OF INTEGERS

05 ARRAYS

Declaring an Array of Integer

- One-dimensional array definition may be expressed as

```
data_type  array_name[elements];
```

  - *data_type*: int
  - *array_name*: any valid variable or identifier name.
  - ***elements***: a positive-value integer constant. (*variable is not allow here*).

- **Ex.** `int f1[5];`

Initializing an Array of Integer

```
//Declaring an Array
int a[8];
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]

```
//Initializing an Array
a[0] = 28;
a[1] = 1543;
a[2] = 0;
a[3] = 15;
a[4] = -2;
a[5] = 1;
a[6] = 0;
a[7] = 62;
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
28	1543	0	15	-2	1	0	62

```
//Declaring an Array with initialized value
int c[] = {28,1543,0,15,-2,1,0,62};
```

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]	c[7]
28	1543	0	15	-2	1	0	62

Ex. An Array of Integers

- **Example:** Input the elements in the array

```
1.  #include <iostream>;
2.
3.  using namespace std;
4.
5.  int main()
6.  {
7.      int a1, a2, a3, a4, a5;
8.
9.      cin >> a1;
10.     cin >> a2;
11.     cin >> a3;
12.     cin >> a4;
13.     cin >> a5;
14. }
```

How do you code program…

if you want to input 100 elements of integer?

Problem 1: An Array of Integers

- **Example:** Write a program to store elements in an array and print it.

```
1.  #include <iostream>;
2.
3.  using namespace std;
4.
5.  int main()
6.  {
7.      int a[100];
8.      int  i, n = 0;
9.
10.     cout << "Enter elements in the array: ";
11.     cin >> n;
12.
13.     // i <= Size of the Array a - 1
14.     for( i = 0; i <= n-1; i++ )
15.          cin >> a[ i ];
16.
17.     cout << "Elements in the array are: ";
18.     for( i = 0; i <= n-1; i++ )
19.          cout << a[ i ];
20.
21. }
```

Answer?

Problem 2: An Array of Integers

- **Example:** Write a program to find the sum of all elements of the array.

```
1.  #include <iostream>;
2.
3.  using namespace std;
4.
5.      int main()
6.      {
7.              int a[] = { 10, 30 };
8.              int  i, total = 0;
9.
10.             for( i = 0; i < 2; i++ )       // i <= Size of the Array a - 1
11.                  total = total + a[ i ];
12.
13.             cout << "The sum is " << total << ".");
14.
15.         }
16.     }
```

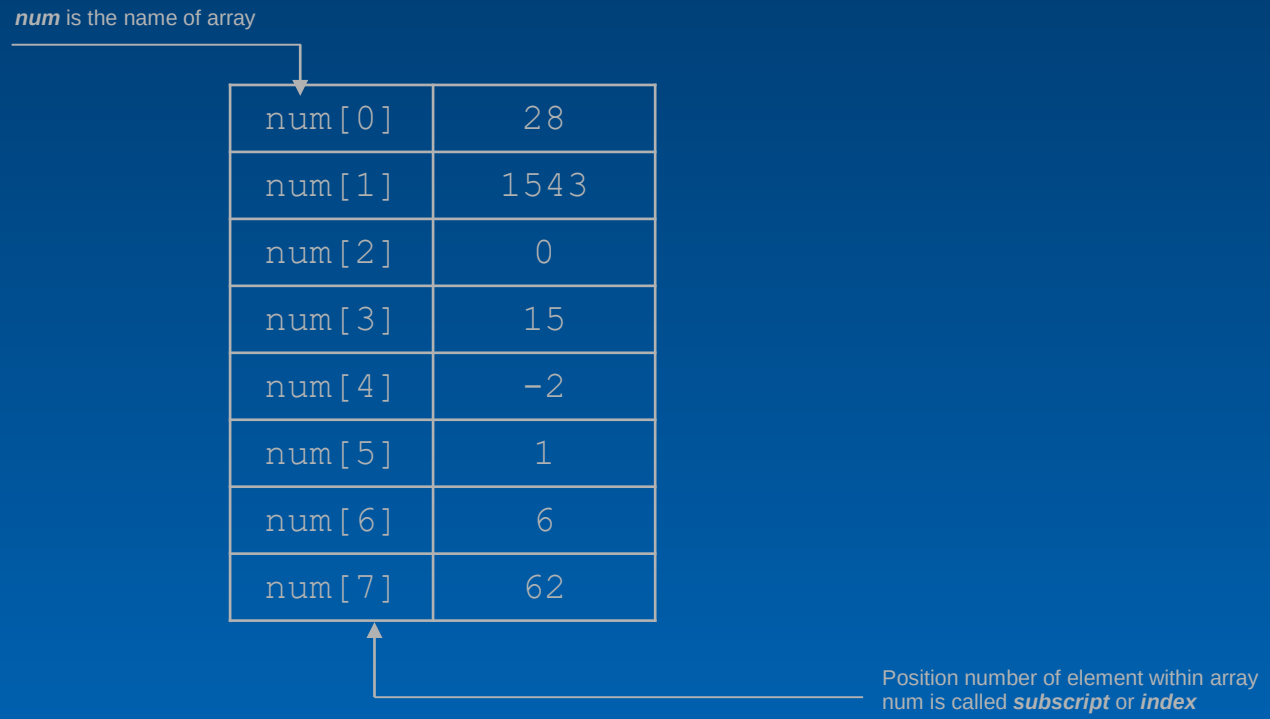
Answer?

Problem 3: An Array of Integers

- **Example:** Write a program to copy the elements one array into another array.

```
1.  #include <iostream>;
2.
3.  using namespace std;
4.
5.      int main()
6.      {
7.              int a[] = { 10, 30 };
8.              int  i, b[2];
9.
10.             for( i = 0; i < 2; i++ )        // i <= Size of the Array a - 1
11.                  b[ i ] = a[ i ];
12.
13.             cout << "The elements copied into the second array are ";
14.             for( i = 0; i < 2; i++ )
15.                  cout << b[i] << " ";
16.
17.         }
18.     }
```

Answer?



C++

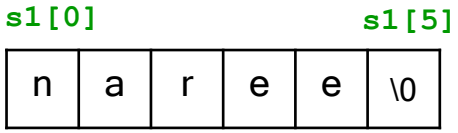
One-dimensional Arrays

ARRAYS OF CHARACTES

05 ARRAYS

Declaring and Initializing an Array of Characters

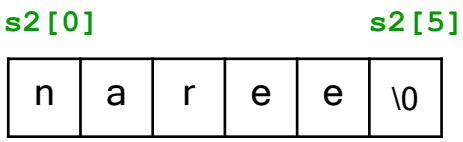
```
char s1[] = {'n','a','r','e','e','\0'};
```



=

```
//Array of characters
char s1[6];
s1[0] = 'n';
s1[1] = 'a';
s1[2] = 'r';
s1[3] = 'e';
s1[4] = 'e';
s1[5] = '\0';
```

```
char s2[] = "naree";
```



=

```
//Array of characters
char s2[6];
s2 = "naree";
```

The String Class in C++ ⇨

```
char s1[] = {'n','a','r','e','e','\0'};
char s2[] = "naree";
String s3 = "naree";
```

Problem 1 : An Array of Characters

- String is an Array of Character.

```
1.  #include <iostream>;
2.  #include <string>;
3.  using namespace std;
4.
5.  int main() {
6.    string s1;
7.    char s2[] = { 'O', 'O', 'P','\0' };
8.    int i;
9.
10.   cout << "Enter a string: ";
11.   cin >> s1;
12.   cout << "S1 is " << s1 << endl;
13.   cout << "S2 is " << s2 << endl;
14.
15.   cout << "S1 with spaces is: ";
16.
17.   for (i = 0; i < s1.length(); i++) {
18.         cout << s1[i] << " ";
19.   }
20.
21.   return 0;
22.}
```

```
Enter a string: Tommy
S1 is
S2 is
S1 with spaces is:
```

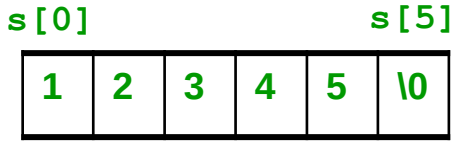
Problem 2 : An Array of Characters

- จงเขียนโปรแกรมเพื่อรับเลขจำนวนเต็ม 5 หลัก แล้วแสดงตัวเลขทั้งหมด โดยกลับลำดับตามลำดับที่รับเข้า ดังนี้

  Enter a 5-digit numbers: *12345*

  Your reverse enter is 54321

```
1. #include <iostream>;
2. using namespace std;
3.
4. int main() {
5.    string s;
6.    cout << "Enter a 5-digit number: ";
7.    cin >> s;
8.    cout << "Your reversed enter is " <<                    ;
9. }
```



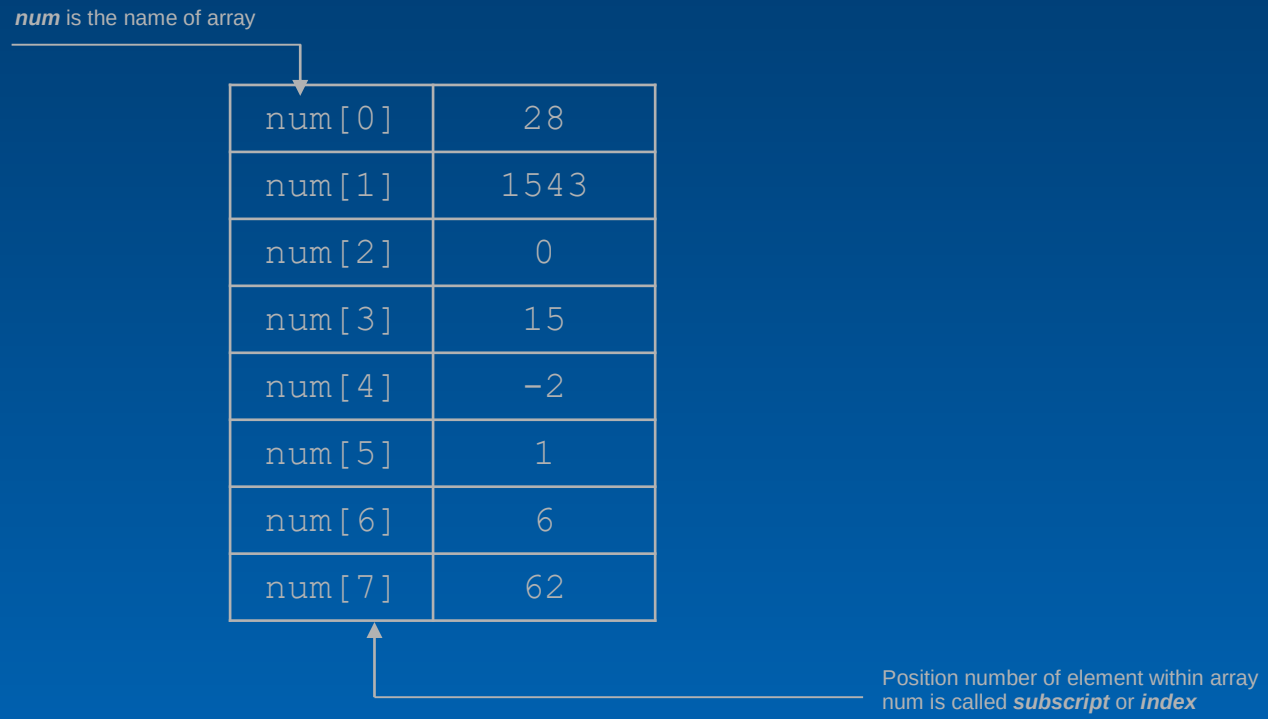
Problem 3 : An Array of Characters

- จงเขียนโปรแกรมเพื่อรับเลขจำนวนเต็ม 20 ตัว แล้วแสดงตัวเลขทั้งหมด โดยกลับลำดับตามลำดับที่รับเข้า ดังนี้

Enter a 20-digit numbers: 12345123451234512345

Your reverse enter is 54321543215432154321

```
1. ...
2. int main() {
3.    string s;
4.    int i;
5.    cout << "Enter a 20-digit number: ";
6.    cin >> s;
7.    cout << "Your reversed enter is ";
8.
9.    for (i = 20; i > 0; i--)
10.         cout << s[i-1];
11.
12.   return 0;
13.}
```



C++

One-dimensional Arrays

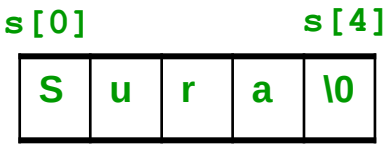
ARRAYS OF STRINGS

05 ARRAYS

Declaring and Initializing an Array of Strings

```
string str = "Sura";
```

➡ an Array of Characters



An Array of Strings

```
string str[2] = {"Sura","naree"};
```

=

```
//Array of strings
string str[2];
str[0] = "Sura";
str[1] = "naree";
```

Problem 1 : An Array of Strings

```
1. #include <iostream>;
2. using namespace std;
3. int main()
4. {
5.       string m[] = {"oU", "ll", "He" };
6.       int i;
7.
8.       for (i = 3; i > 0; i-- )
9.       {
10.           cout << m[i-1]);
11.      }
12.
13.      return 0;
14. }
```

Problem 2 : An Array of Strings

2.1

```
String s[] = {"He","She","It", "They", "We"}
int i, n=5;
for (  i = 0;  i <= n;  i++  )
      cout << str[ i ] << " ";
```

Answer?

2.2

```
String s[] = {"He","She","It", "They", "We"}
int i, n=5;
for (i = n-1; i >= 0; i--)
      cout << str[ i ] << " ";
```

Answer?

Problem 3 : An Array of Strings

3.1

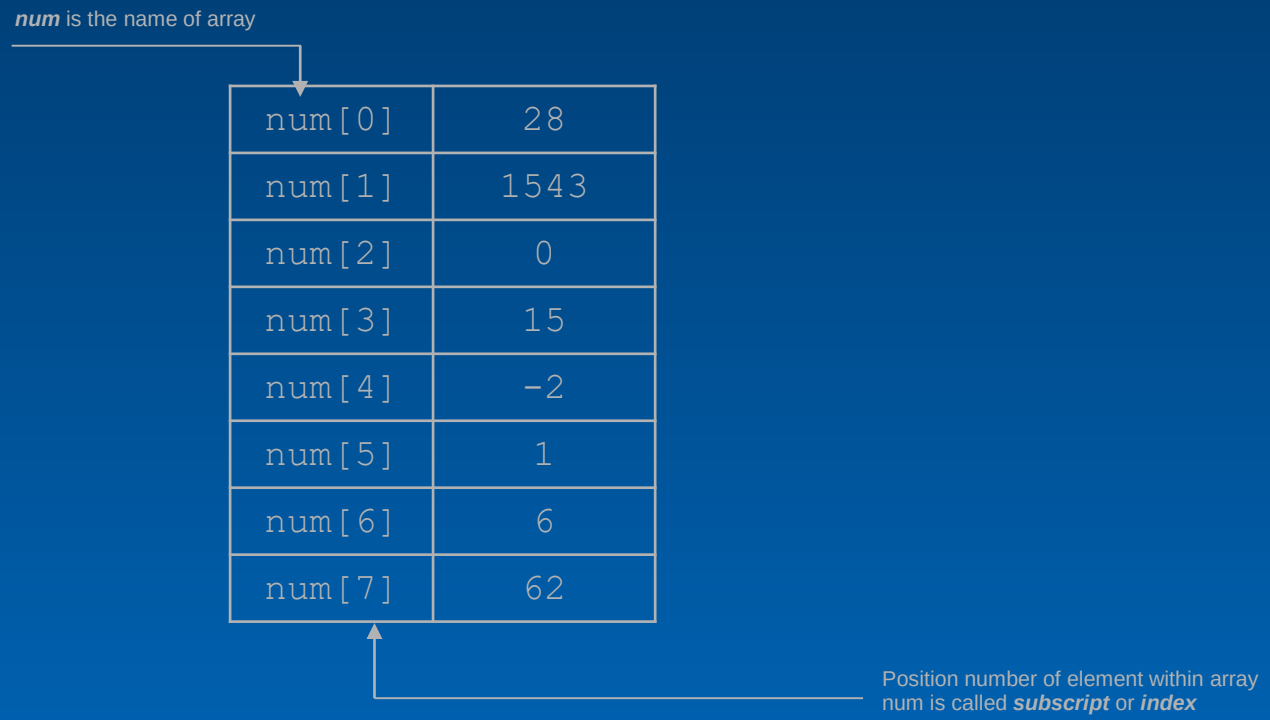
```
String s[] = {"He","She","It", "They", "We"}
int i, n=5;
for (i = n;   i > 0;   i--)
      cout << str[ i-1 ] << " ";
```

Answer?

3.2

```
String s[] = {"He","She","It", "They", "We"}
int i, n=5;
for (i = 0; i <= 3; i++)
      cout << str[ i++ ] << " ";
```

Answer?



C++

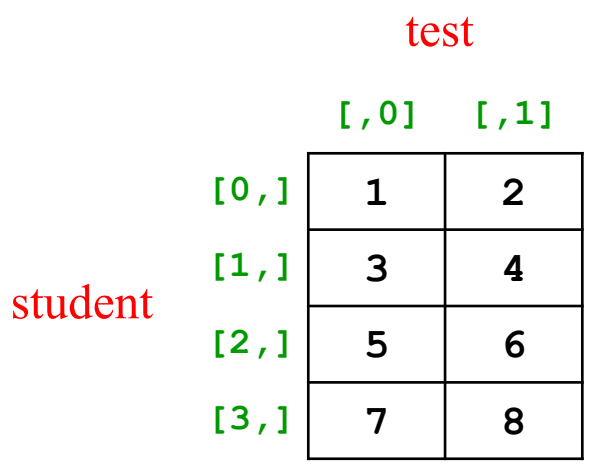
Data Structure

TWO-DIMENSIONAL ARRAYS

05 ARRAYS

Two-dimensional Arrays

- int score[4][2];
  - 8 variables (4*2)
  - score[0][0] keeps score for student 1 test 1
  - score[0][1] keeps score for student 1 test 2
  - …
  - score[3][1] keeps score for student 4 test 2
  - size of score[4][2] is 4 * 2 * 4 bytes (*size of int = 4 bytes*)



Declaring and Initializing Two-dimensional Arrays

```
// Two-dimensional array.
int score1[][2] = { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };

// The same array with dimensions specified.
int score2[4][2] = { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

Two-dimensional Arrays (of Float)

- float temperature[12][31];
  - 372 variables (12*31)
  - temperature[0][0] keeps temperature in month 1 day 1
  - temperature[0][1] keeps temperature in month 1 day 2
  - …
  - temperature[11][30] keeps temperature in month 12 day 31
  - size of temperature[12][31] is 12*31*4 bytes (*size of float = 4 bytes*)

	[,0]	[,1]	[,2]	...	[,30]
[0,]	27.5	27.4	28.1	.	32.4
[1,]	29.9	30.0	31.4	.	24.7
...	.	.	.	.	.
[11,]	24.3	25.2	26.1	.	24.6

Declaring and Initializing Two-dimensional Arrays

```
// Two-dimensional array.
int temperature[12][31] = { {27.5,27.4,28.1,…,32.4},    /* row indexed by 0 */
                            {29.9,30.0,31.4,…,24.7},    /* row indexed by 1 */
                                        …,
                            {24.3,25.2,26.1,…,24.6}     /* row indexed by 11 */
                          };
```

Two-dimensional Arrays (of Characters)

- char student[10][31];
  - student[0], student[1],..., student[9]
  - each string can store up to 31 characters
  - size of student[10][31] is 10*31*2 bytes

    (*size of character = 2 bytes*)

	[,0]	[,1]	[,2]	[,3]	[,4]	...	[,30]
[0,]	'T'	'o'	'm'	…	…	…	'y'
[1,]	'J'	'o'	'h'	'n'	…	…	't'
...	…	…	…	…	…	…	…
[9,]	'N'	'a'	'n'	…	…	…	'c'

Declaring and Initializing Two-dimensional Arrays

```
// A similar array with string elements.
char student[10][31] = { {'T','o','m',…,'y'},     /* row indexed by 0 */
                         {'J','o','h',…,'t'},     /* row indexed by 1 */
                                    …,
                         {'N','a','n',…,'c'}      /* row indexed by 9 */
                       };
```

Two-dimensional Arrays (of Strings)

```
// A similar array with string elements.
string members[3][2] = {{"one", "two"}, {"three", "four"}, {"five", "six"}};
```

String members[3][2];

members[0][0] = "one";

members[0][1] = "two";

members[1][0] = "three";

members[1][1] = "four";

members[2][0] = "five";

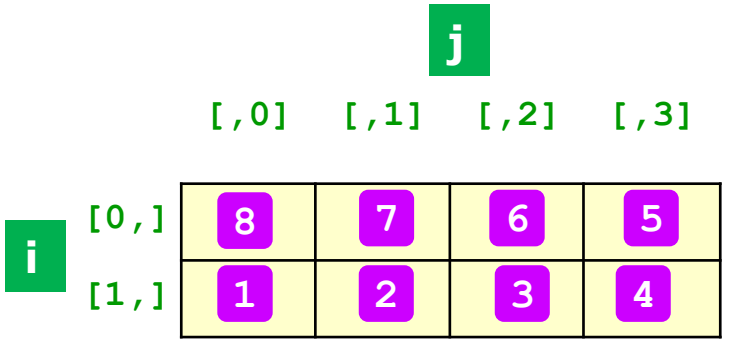
members[2][1] = "four";

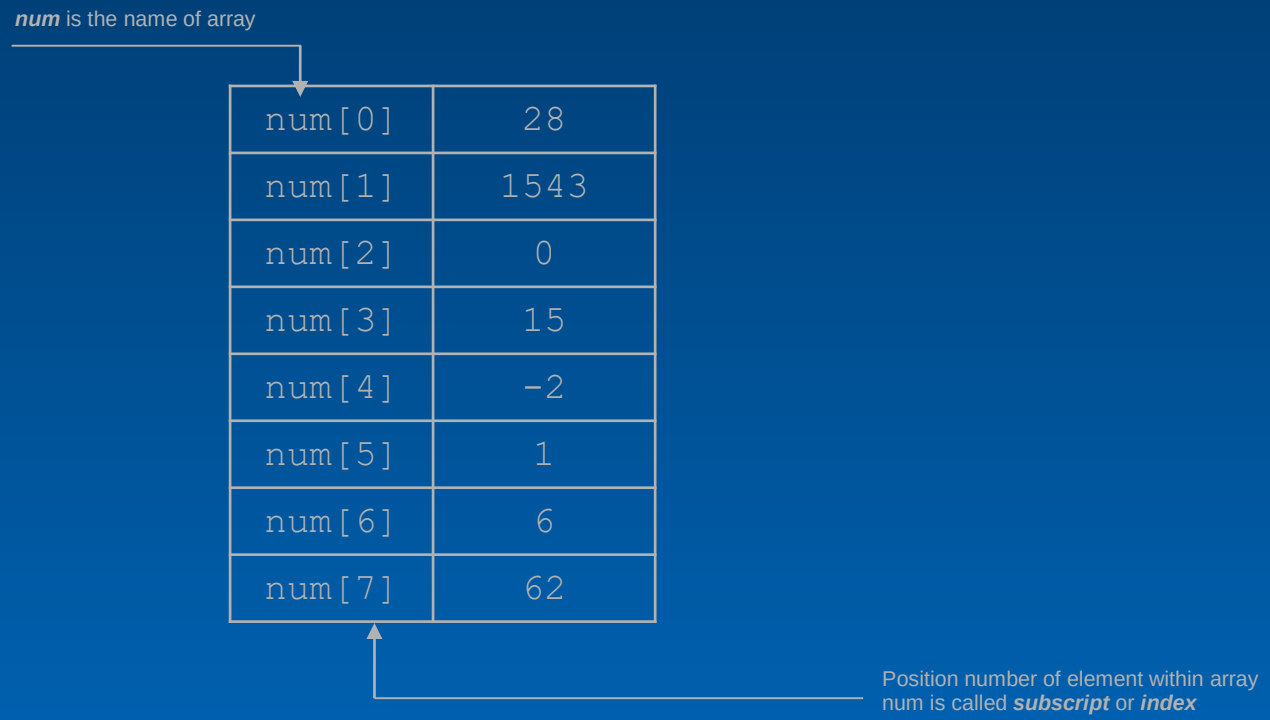
	[,0]	[,1]
[0,]	"one"	"two"
[1,]	"three"	"four"
[2,]	"five"	"six"

Example: Input a Matrix

```
1.    ...
2.    int main() {
3.       int  i, j;
4.       int scores[2][4];
5.
6.         cout << "Input elements in the matrix:" << endl;
7.         for (  i = 0;   i < 2;   i++ ) {
8.            for ( j = 0;  j < 4;   j++ ) {
9.               cout << "scores[" << i << "][" << j << "]: ";
10.              cin >> scores[i][j];
11.           }
12.        }
13.
14.      return 0;
15.   }
```

```
Input elements in the matrix:
scores[0][0]: 8      i = 0    j = 0
scores[0][1]: 7               j = 1
scores[0][2]: 6               j = 2
scores[0][3]: 5               j = 3
scores[1][0]: 1      i = 1    j = 0
scores[1][1]: 2               j = 1
scores[1][2]: 3               j = 2
scores[1][3]: 4               j = 3
```





C++

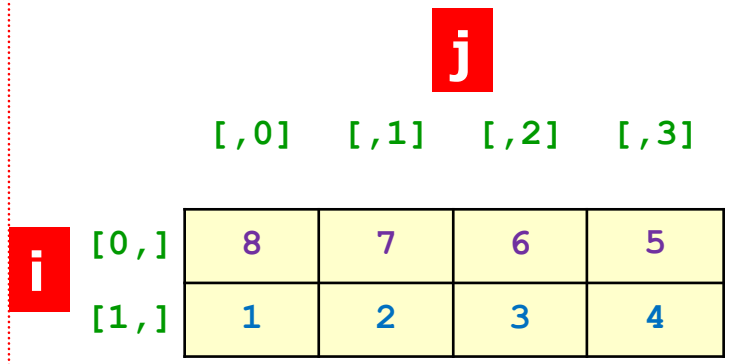
Problems

TWO-DIMENSIONAL ARRAYS

05 ARRAYS

Problem 1: Input and Print a 2D-Array (Matrix)

```
1.   #include <iostream>;
2.   #include <string>;
3.   using namespace std;
4.   int main() {
5.     int  i, j;
6.     int scores[2][4];
7.
8.       cout << "Input elements in the matrix:" << endl;
9.       for (i = 0; i < 2; i++) {
10.        for (j = 0; j < 4; j++) {
11.          cout << "scores[" << i << "][" << j << "]: ";
12.          cin >> scores[i][j];
13.        }
14.      }
15.
16.      cout << "Elements in the matrix:" << endl;
17.      for (i = 0; i < 2; i++) {
18.        for (j = 0; j < 4; j++) {
19.          cout << scores[i][j] << " ";
20.        }
21.        cout << endl;
22.      }
23.    return 0;
24.  }
```




```
Input elements in the matrix:
scores[0][0]: 8
scores[0][1]: 7
scores[0][2]: 6
scores[0][3]: 5
scores[1][0]: 1
scores[1][1]: 2
scores[1][2]: 3
scores[1][3]: 4
```

Answer?

Problem 2: Input and Print a *n* x *m* Matrix

```
1.   #include <iostream>;
2.   #include <string>;
3.   using namespace std;
4.   int main() {
5.     int i, j, n, m;
6.     float scores[4][9];
7.     cin >> n >> m;
8.       cout << "Input elements in the matrix:" << endl;
9.       for (i = 0; i < n; i++) {
10.        for (j = 0; j < m; j++) {
11.          cout << "scores[" << i << "][" << j << "]: ";
12.          cin >> scores[i][j];
13.        }
14.      }
15.
16.      cout << "Elements in the matrix:" << endl;
17.      for (i = 0; i < n; i++) {
18.        for (j = 0; j < m; j++) {
19.          cout << scores[i][j] << " ";
20.        }
21.        cout << endl;
22.      }
23.    return 0;
24.  }
```

Maximum number

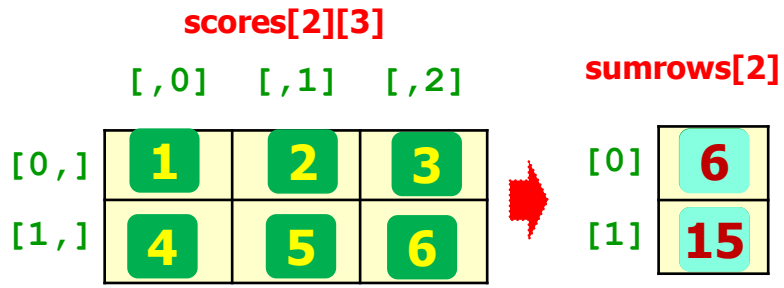
Where $0 < n < 5$, $0 < m < 10$

	[,0]	[,1]	[,2]	...	[,*m-1*]
[0,]	27.5	27.4	28.1	.	32.4
[1,]	29.9	30.0	31.4	.	24.7
...	.	.	.	.	.
[*n-1*,]	24.3	25.2	26.1	.	24.6

How to change this program?

Problem 3: Sum of Rows of a 2x3 Matrix

```
1.    ...
2.    int main() {
3.      int  n = 0, i, j, sumrows[2];
4.      int scores[2][3];
5.
6.        cout << "Input elements in the matrix:" << endl;
7.        for (i = 0; i < 2; i++) {
8.            for (j = 0; j < 3; j++) {
9.                cout << "scores[" << i << "][" << j << "]: ";
10.               cin >> scores[i][j];
11.           }
12.           sumrows[i] = 0;
13.       }
14.
15.       cout << "Total Scores:" << endl;
16.       for (i = 0; i < 2; i++) {
17.           for (j = 0; j < 3; j++) {
18.               cout << scores[i][j] << " ";
19.               sumrows[i] += scores[i][j];
20.           }
21.           cout << "= " << sumrows[i] << " " << endl;
22.       }
23.     return 0;
24.   }
```




```
Input elements in the matrix:
scores[0][0]: 1
scores[0][1]: 2
scores[0][2]: 3
scores[1][0]: 4
scores[1][1]: 5
scores[1][2]: 6
```

Answer?

Problem 4: Sum of Rows of a *n* x3 Matrix

```
1.    int main() {
2.       string student[100];
3.       int  n = 0, i, j;
4.       int scores[100][4];
5.
6.          do {
7.            cout << "Enter a student name: ";
8.            cin >> student[n];
9.          } while (student[n++] != "END");
10.
11.         if (n-- != 1) {
12.            for (i = 0; i < n; i++) {
13.                scores[i][3] = 0;
14.                cout << "\nEnter scores for " << student[i] << ":" << endl;
15.
16.                for (j = 0; j < 3; j++) {
17.                     cout << "Test[" << i << "][" << j << "]: ";
18.                     cin >> scores[i][j];
19.                     scores[i][3] += scores[i][j];
20.                }
21.             }
22.
23.             cout << "\nTotal Scores: " << endl;
24.             for (i = 0; i < n; i++)
25.                  cout << student[i] << " : " << scores[i][3] << endl;
26.          }
27.
28.       return 0;
29.    }
```

```
Enter a student name : Somchai
Enter a student name : Somsri
Enter a student name : END
Enter scores for Somchai:
Test[0][0]: 9
Test[0][1]: 10
Test[0][2]: 8
Enter scores for Somsri:
Test[1][0]: 10
Test[1][1]: 10
Test[1][2]: 10
```

Scores

	[,0]	[,1]	[,2]	[,3]
[0,]	9	10	8	27
[1,]	10	10	10	30
:				
[n,]				

j

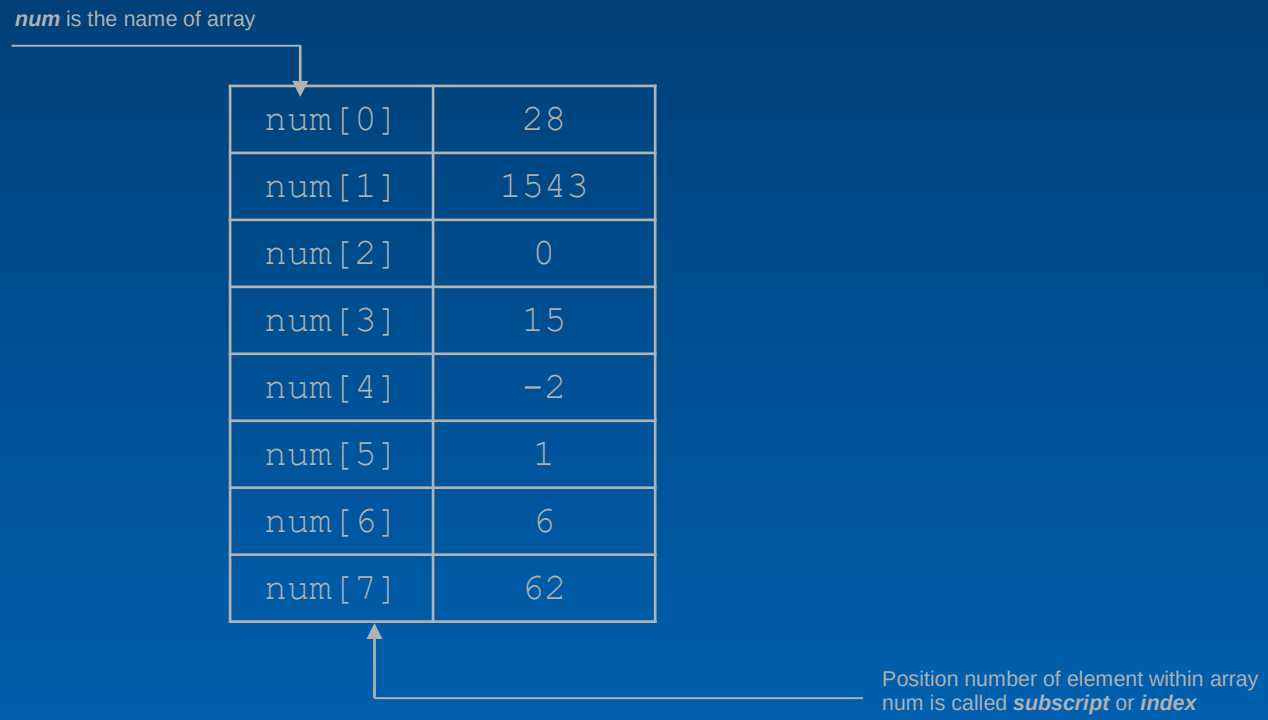
i

Problem 5: Addition of two matrices

```
1.      ...
2.      int main() {
3.        int  n = 0, i, j;
4.        int matrix1[2][3],matrix2[2][3];
5.
6.          cout << "The first matrix:" << endl;
7.          for (i = 0; i < 2; i++) {
8.            for (j = 0; j < 3; j++) {
9.              cout << "matrix1[" << i << "][" << j << "]: ";
10.             cin >> matrix1[i][j];
11.           }
12.         }
13.         cout << "The second matrix:" << endl;
14.         for (i = 0; i < 2; i++) {
15.           for (j = 0; j < 3; j++) {
16.             cout << "matrix2[" << i << "][" << j << "]: ";
17.             cin >> matrix2[i][j];
18.           }
19.         }
20.
21.          cout << "Summation in the matrix:" << endl;
22.         for (i = 0; i < 2; i++) {
23.           cout << "row[" << i << "][..] = ";
24.           for (j = 0; j < 3; j++) {
25.             cout <<          Code?          << " ";
26.           }
27.           cout << endl;
28.         }
29.
30.       return 0;
31.     }
```

```
The first matrix:
matrix1[0][0]: 8
matrix1[0][1]: 7
matrix1[0][2]: 6
matrix1[1][0]: 5
matrix1[1][1]: 4
matrix1[1][2]: 3

The second matrix:
matrix2[0][0]: 1
matrix2[0][1]: 2
matrix2[0][2]: 3
matrix2[1][0]: 4
matrix2[1][1]: 5
matrix2[1][2]: 6

Summation in the matrix:
row[0][..] = 9 9 9
row[1][..] = 9 9 9
```

i = 0	j = 0	Addition[0][0] = 8+1 = 9
	j = 1	Addition[0][1] = 7+2 = 9
	j = 2	Addition[0][2] = 6+3 = 9
i = 1	j = 0	Addition[1][0] = 5+4 = 9
	j = 1	Addition[1][1] = 4+5 = 9
	j = 2	Addition[1][2] = 3+6 = 9



C++

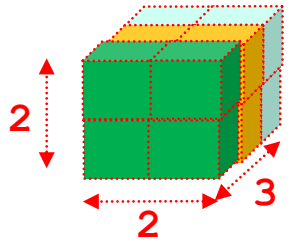
Data Structure

MULTI-DIMENSIONAL ARRAYS

05 ARRAYS

Multi-dimensional Arrays

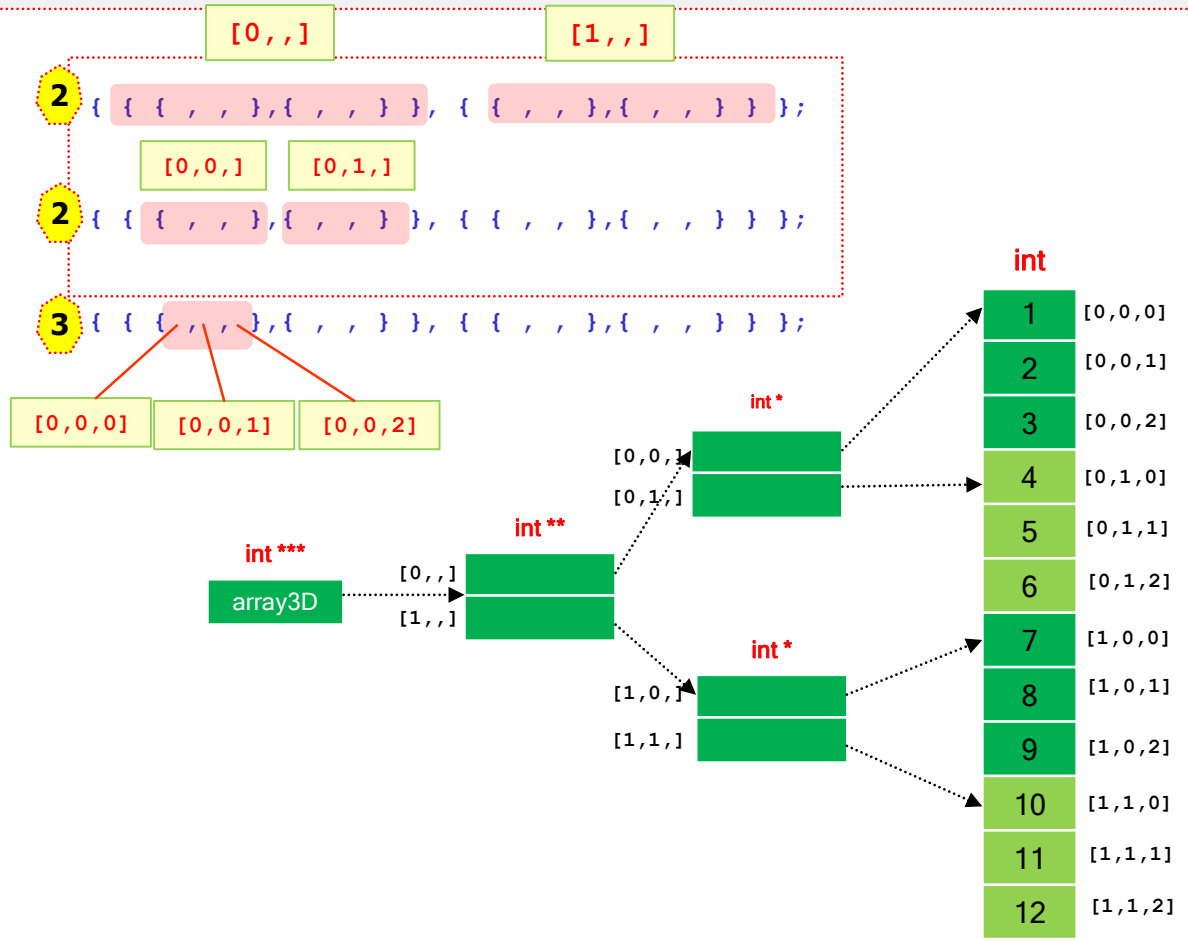
```
// Three-dimensional array.
int array3D[][2][3] = { { { 1, 2, 3 }, { 4, 5, 6 } }, { { 7, 8, 9 }, { 10, 11, 12 } } };

// The same array with dimensions specified.
int array3Da[2][2][3] = { { { 1, 2, 3 }, { 4, 5, 6 } },{ { 7, 8, 9 }, { 10, 11, 12 } } };
```



int members[2][2][3]

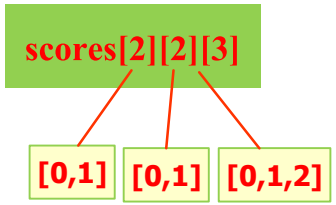
members[0][0][0] = 1;
members[0][0][1] = 2;
members[0][0][2] = 3;
members[0][1][0] = 4;
members[0][1][1] = 5;
members[0][1][2] = 6;

members[1][0][0] = 7;
members[1][0][1] = 8;
members[1][0][2] = 9;
members[1][1][0] = 10;
members[1][1][1] = 11;
members[1][1][2] = 12;



Example: Input and Print a 2x2x3 Matrix

```
1.  #include <iostream>;
2.  #include <string>;
3.  using namespace std;
4.  int main() {
5.    int  i, j;
6.    int scores[2][2][3];
7.
8.    cout << "Input elements in the matrix:" << endl;
9.    for (i = 0; i < 2; i++) {
10.     for (j = 0; j < 2; j++) {
11.       for (k = 0; k < 3; k++) {
12.         cout << "scores[" << i << "][" << j << "][" << k << "]: ";
13.         cin >> scores[i][j][k];
14.       }
15.     }
16.   }
17.   cout << "Elements in the matrix:" << endl;
18.   for (i = 0; i < 2; i++) {
19.     for (j = 0; j < 4; j++) {
20.       cout << scores[i][j] << " ";
21.     }
22.     cout << endl;
23.   }
24.   return 0;
    }
```




```
Input elements in the matrix:
scores[0][0][0]: 1
scores[0][0][1]: 2
scores[0][0][2]: 3
scores[0][1][0]: 4
scores[0][1][1]: 5
scores[0][1][2]: 6
scores[1][0][0]: 7
scores[1][0][1]: 8
scores[1][0][2]: 9
scores[1][1][0]: 10
scores[1][1][1]: 11
scores[1][1][2]: 12

Elements in the matrix:
scores[0][][]: 1 2 3 4 5 6
scores[1][][]: 7 8 9 10 11 12
```

i = 0 j = 0 k = 0 k = 1 k = 2
j = 1 k = 0 k = 1 k = 2
i = 1 j = 0 k = 0 k = 1 k = 2
j = 1 k = 0 k = 1 k = 2

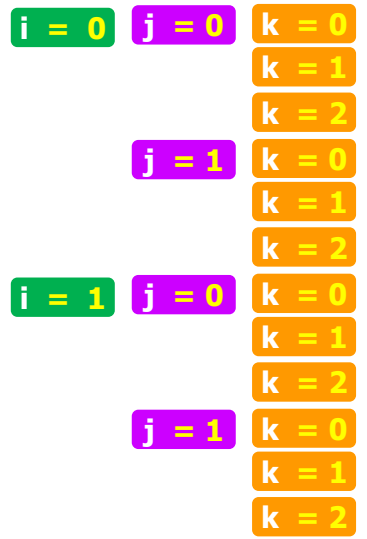
```
cout << "Elements in the matrix:" << endl;
for (i = 0; i < 2; i++) {
  cout << "scores[" << i << "][][]: ";
  for (j = 0; j < 2; j++) {
    for (k = 0; k < 3; k++) {
    cout << scores[i][j][k] << " ";
    }
  }
  cout << endl;
}
```

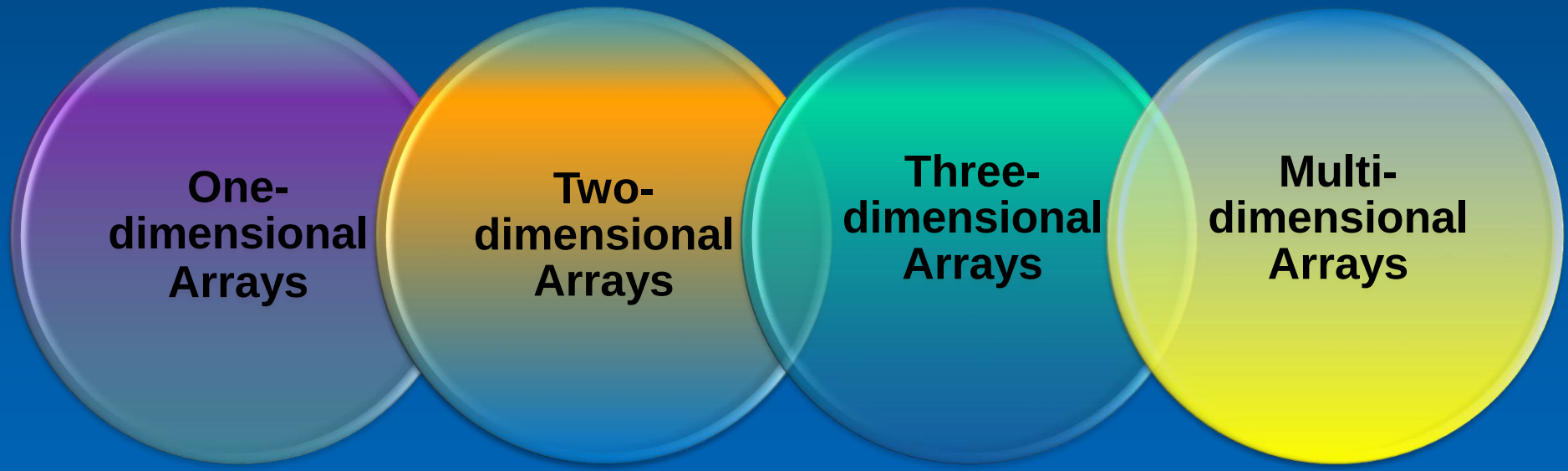
Example: Input and Print a 2x2x3 Matrix

```
1.    int main() {
2.      int  i, j, k;
3.      float scores[2][2][3];
4.
5.       cout << "Input elements in the matrix:" << endl;
6.      for (i = 0; i < 2; i++) {
7.        for (j = 0; j < 2; j++) {
8.          for (k = 0; k < 3; k++) {
9.              cout << "scores[" << i << "][" << j << "][" << k << "]: ";
10.             cin >> scores[i][j][k];
11.         }
12.       }
13.     }
14.     cout << "Elements in the matrix:" << endl;
15.     for (i = 0; i < 2; i++) {
16.       cout << "scores[" << i << "][][]: ";
17.       for (j = 0; j < 2; j++) {
18.         for (k = 0; k < 3; k++) {
19.            cout << scores[i][j][k] << " ";
20.         }
21.       }
22.       cout << endl;
23.     }
24.
25.     .....
26.
27.     return 0;
28.   }
```

```
Input elements in the matrix:
scores[0][0][0]: 1
scores[0][0][1]: 2
scores[0][0][2]: 3
scores[0][1][0]: 4
scores[0][1][1]: 5
scores[0][1][2]: 6
scores[1][0][0]: 7
scores[1][0][1]: 8
scores[1][0][2]: 9
scores[1][1][0]: 10
scores[1][1][1]: 11
scores[1][1][2]: 12

Elements in the matrix:
scores[0][][]: 1 2 3 4 5 6
scores[1][][]: 7 8 9 10 11 12

Elements in the matrix:
scores[0][0][]: 1 2 3
scores[0][1][]: 4 5 6
scores[1][0][]: 7 8 9
scores[1][1][]: 10 11 12
```

```
cout << "Elements in the matrix:" << endl;
 for (i = 0; i < 2; i++) {
   for (j = 0; j < 2; j++) {
       cout << "scores[" << i << "][" << j << "][]: ";
       for (k = 0; k < 3; k++) {
           cout << scores[i][j][k] << " ";
       }
       cout << endl;
   }
 }
```





Data Structure
ARRAYS