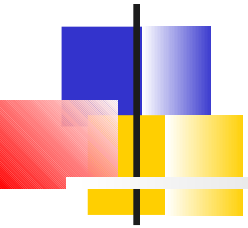
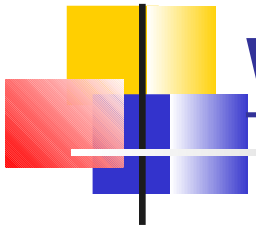
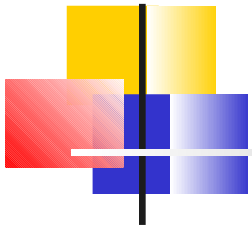


Arrays



What is Array?

- An array is a fixed-size sequential collection of elements of same data types that share a common name.
- It is simply a group of data types.
- An array is a derived data type.
- An array is used to represent a list of numbers , or a list of names.



Example of Arrays:

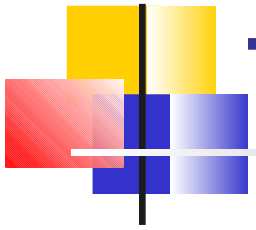
- For Example :

1. List of employees in an organization.
2. Test scores of a class of students.
3. List of customers and their telephone numbers.
4. List of students in the college.

- For Example, to represent 100 students in college , can be written as

student [100]

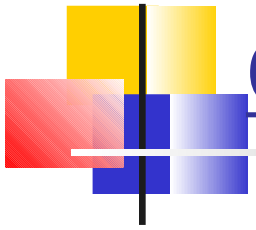
- Here student is a array name and [100] is called index or subscript.



Types of Arrays

- Types of Array :

1. One-dimensional arrays
2. Two-dimensional arrays
3. Multidimensional arrays



One-dimensional Arrays.

- A variable which represent the list of items using only one index (subscript) is called one-dimensional array.

- For Example , if we want to represent a set of five numbers say(35,40,20,57,19), by an array variable number, then number is declared as follows

```
int number [5] ;
```

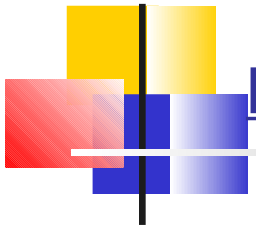
One-dimensional Arrays.

- and the computer store these numbers as shown below :

  number [0]
  number [1]
  number [2]
  number [3]
  number [4]

- The values can be assigned to the array as follows :

  number [0] = 35;
  number [1] = 40;
  number [2] = 20;
  number [3] = 57;
  number [4] = 19;

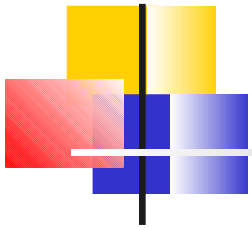


DECLARATION OF ONE-DIMENSIONAL ARRAYS :

- The general form of array declaration is :

  type *array-name[size];*

- Here the type specifies the data type of elements contained in the array, such as int, float, or char.
- And the size indicates the maximum numbers of elements that can be stored inside the array.
- The size should be either a numeric constant or a symbolic constant.

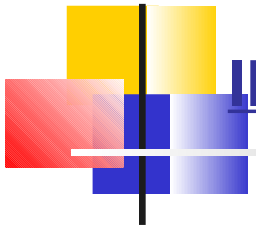


Example

- For example :

  int group [10] ;

- here int is type, group is a variable name , 10 is a size of array and the subscripts (index) is start from 0 to 9.



INITIALIZATION OF ONE-DIMENSIONAL ARRAYS :

- An array can be stored in following stages :

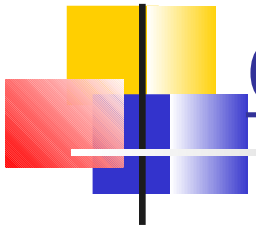
1. At compile time
2. At run time

Compile time initialization :

- In compile time initialization, the array is initialized when they are declared.
- The general form of initialization of array is :

  type array-name[size] = { list of values };

- The list of values are separated by commas.



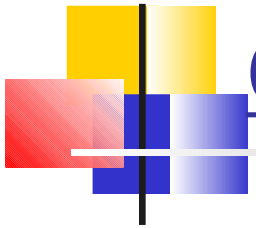
Compile time initialization

Example :

```
int number[3] = {4,5,9};
```

- Here array number of size 3 and will assign 4 to first element(number[0]), 5 is assign with second element(number[1]) and 9 is assign with third element(number[2]).

- If the number of values in the list is less than the number of elements, then only that many elements will be intialized. The remaining elements will be set to zero automatically.



Compile time initialization

Example :

```
int number[ ] = {1,2,3,4};
```

- The character array can be initialized as follows :

```
char name[ ] = {'j','o','h','n','\0'};
```

- The character array can also be initialized as follows :

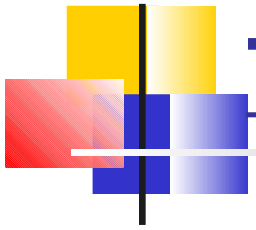
```
char name[ ] = "john";
```

Run time initialization :

- In run time initialization, the array is explicitly initialize at run time.
- This concept generally used for initializing large arrays.
- Example:

```
for(i=0; i < 100; i++)
{
        if( i < 50)
        sum[i] = 0.0;
        else
        sum[i] = 1.0;
}
```

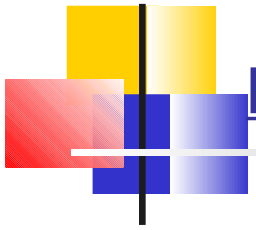
- Here first 50 elements of the array sum are initialized to 0 and the remaining 50 elements are initialized to 1 at run time.



Two-dimensional Arrays

- A variable which represent the list of items using two index (subscript) is called two-dimensional array.
- In Two dimensional arrays, the data is stored in rows and columns format.
- For example:

```
int table[2][3];
```

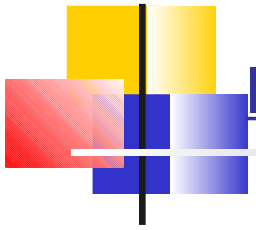


DECLARATION OF TWO-DIMENSIONAL ARRAYS :

- The general form of two dimensional array declaration is :

  type *array-name[row_size][column_size];*

- Here the type specifies the data type of elements contained in the array, such as int, float, or char.
- The size should be either a numeric constant or a symbolic constant.



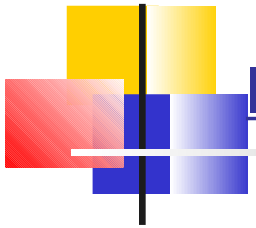
INITIALIZATION OF TWO-DIMENSIONAL ARRAYS :

- The general form of initializing two-dimensional array is :
  type array-name[row_size][column_size] = {list

  of values};
- Example :

  int table[2][3] = {0,0,0,1,1,1};
- Here the elements of first row initializes to zero and the elements of second row initializes to one.
- This above statement can be written as :

  int table[2][3] = {{0,0,0}, {1,1,1}};
- In two-dimensional array the row_size can be omitted.



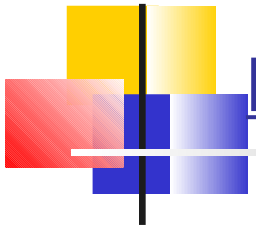
INITIALIZATION OF TWO-DIMENSIONAL ARRAYS :

- Example :

  int table[ ][3] = {{0,0,0}, {1,1,1}};
- If the values are missing in an initializer, they are automatically set to zero.

- Example :

  int table[2][3] = {1,1,2};
- Here first row initialize to 1,1 and 2, and second row initialize to 0,0 and 0 automatically.



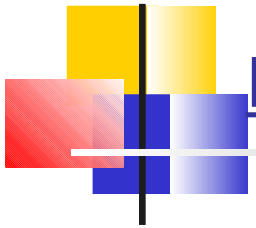
Memory Layout of Two-dimensional array :

- In Two dimensional arrays, the data is stored in rows and columns format.
- For example:

```
int table[2][3] = {1,2,3,4,5,6};
```

- The memory layout of above example :

```
table[0][0] = 1;
table[0][1] = 2
table[0][2] = 3;
 table[1][0] = 4;
table[1][1] = 5;
 table[1][2] = 6;
```

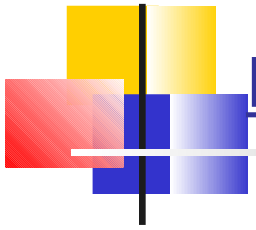


multi-dimensional Arrays

- A variable which represent the list of items using more than two index (subscript) is called multi-dimensional array.

- The general form of multi dimensional array is :

  type *array-name[s1][s2][s3]…….[sn];*

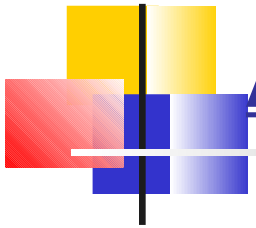


multi-dimensional Arrays

- Where S is the size. Some examples are :

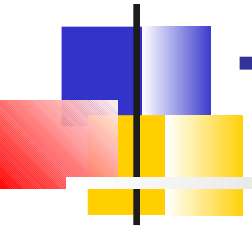
```
int  survey[3][5][6];

float  table[5][4][5][3];
```

- Here survey is a three-dimensional array And table is a four-dimensional array.



Applications Of arrays

- Using pointers for accessing arrays.
- Passing arrays as function parameters.
- Arrays as members of structures.
- Using structure type data as array elements.
- Arrays as dynamic data structures.
- Manipulating character arrays and strings.



THANK YOU