C Language

Topic : Arrays & Strings

Summary

This session covers the below topics on Arrays & Strings.

1. Arrays:
    1. Why Arrays
    2. Use of Arrays
    3. Advantage of using arrays
    4. Syntax of Arrays
    5. Example Program and real time example
    6. Lab Programs
2. Strings:
    1. Why Strings
    2. Use of strings
    3. Advantage of using strings
    4. Syntax of strings (declaration and initialization(define)).
    5. Example Program
    6. Lab Programs
3. Comparison b/w the arrays and strings.
4. Revise the previous topics which depends in this session.
5. Questionnaires on this topic
6. Assessment
7. Lab Execution of Assessment

1. Arrays

a. Why Arrays:

Array is a collection of homogeneous values hold by the single variable and contains the single data type for all the values.

Array can be data structure that contains the sequential data of fixed array size of same data type.

Stores data into contiguous memory location in the stack.

Array index starts from “0” to “n-1”.

Data type of an Array can be “char”, “int”, “float”.

b. Use of Arrays
   i. Used to store large number of data information is of same data range.
   ii. Arrays are used for 1D, 2D, 3D and multi dimensional arrays.
   iii. Array variable name holds the address of the first index.
c. Advantage of using arrays
   i. Accessing of data is faster and simpler using the array index.
   ii. Can store string information as a single variable.
   iii. Arrays contains the static memory.
d. Syntax of Arrays
   i. Single Dimensional Array:
      1. DataType array_variablename [index];
         a. DataType is “int”, “char”, “float”.
         b. array_variablename is the name of the array to access in the program.
         c. index describes the size of the array to hold data.
         d. [] is single dimension.

EX: int employe_no [10]; /*Declaration*/

int employe_no [10] = {0, 2, 3, 4, 5, 6, 7, 8, 9, 10}; /*Initialization*/

Logical Representation:

Employe_no	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
	0	2	3	4	5	6	7	8	9	10

2. Array can be use for store string information.

EX1: char employe_name [13] = “employe_name”;

EX1: char employe_name [13] = {'e', 'm', 'p', 'l', 'o', 'y', 'e', '_', 'n', 'a', 'm', 'e'};

Note1: String ends with "\0" of last index element. So string name should not exceed the length more than "index-1".

ii. Two Dimensional Array:

1. DataType array_variablename [index1][index2]; /*[row][col]*/
   a. DataType is "int", "char", "float".
   b. array_variablename is the name of the array to access in the program.
   c. index describes the size of the array to hold data.
   d. [][] is two dimensions.

```
EX1:  int employe_details [7][2];  /*Declaration*/
      int employe_details [7][2] = {   /*Initialization */
                                    {2, 5},
                                    {3,4},
                                    {2,6},
                                    {4,3},
                                    {5,7},
                                    {8,4},
                                    {9,0}
                                };
EX1: int employe_details [7][2] = {2, 5,3,4,2,6,4,3,5,7,8,4,9,0};
```

Logical Representation:

employe_details

	[0]	[1]
[0]	2	5
[1]	3	4
[2]	2	6
[3]	4	3
[4]	5	7
[5]	8	4
[7]	9	0

iii. Muti Dimensional Array:

1. DataType array_variablename [index1][index2] ..... [indexn];
    a. DataType is “int”, “char”, “float”.
    b. array_variablename is the name of the array to access in the program.
    c. index describes the size of the array to hold data.
    d. [][] is two dimensions.

EX1:
```
int employe_details [3][2][3] = {
                                {
                                {2, 5,6},
                                {2,6,9}
                                },
                                {
                                {4,3,5},
                                {5,7,10}
                                },
                                {
                                {6,9,12},
                                {8,14,18}
                                },

                                };
```

EX2:
```
int employe_details [3][2][3] = {2,5,6,2,6,9,4,3,5,5,7,10,6,9,12,8,14,18};
```

Logical Representation:

employe_details

Table[0]:

	[0]	[1]	[2]
[0]	2	5	6
[1]	2	6	9

Table[1]

	[0]	[1]	[2]
[0]	4	3	5
[1]	5	7	10

Table[2]:

	[0]	[1]	[2]
[0]	6	9	12
[1]	8	14	18

e. Example Program and real time example
f. Lab Programs

2. Strings

3.