C Programming

Lesson 4: Arrays

Lesson Objectives

- **To understand the following topics:**
  - Single-Dimension Array
  - Two-Dimension Array
  - Strings
  - Common Best Practices

Memory Representation Of Two Dimensional Array

- **A two-dimensional array a[i][j] can be visualized as a table or a matrix of i rows and j column**
- **All the elements in a row are placed in contiguous memory locations**

	col 1	col 2		col colid-1	col colid
row 1	Emp[0][0]	Emp[0][1]	----	Emp[0][colid-2]	Emp[0][colid-1]
row 2	Emp[1][0]	Emp[1][1]	----	Emp[1][colid-2]	Emp[1][colid-1]
row rowid-1	Emp[rowid-2][0]	Emp[rowid-2][1]	----	Emp[i-2][colid-2]	Emp[rowid-2][colid-1]
row rowid	Emp[rowid-1][0]	Emp[rowid-1][1]	----	Emp[rowid-1][colid-2]	Emp[rowid-1][colid-1]

IGATE
Speed.Agility.Imagination

Two-Dimension Array Processing

- **Processing of two-dimensional array is same as that of single dimensional arrays.**
- **The common way to process a two-dimension array is by means of nested loops.**

Two-Dimensional Arrays: Sample Program

```
/*students in a test by accepting roll number and marks of each student */
 # include <stdio.h>
void main(void){
            int index;
            float sum=0;
            int student[25][2];
            for(index=0; index<25; index++)        {
                        printf(“Enter Roll no and marks : “);
                        scanf(“%d%d”, &student[index][0], &student[index][1]);
                        sum += student[index][1];
      /* Roll no will get stored in students[index][0] and marks in student[index][1]
          */
          }
          printf(“\n Average marks : %.2f\n”, sum/25);
      }
```

Two-Dimensional Array of Characters

```
char namelist[3][10] ={
                                "akshay",
                                "parag",
                                "raman"
                    };
```

Declares an Array of 3 strings of 10 characters.

The memory representation of the above array is

1001	a	k	s	h	a	y	\0	▨	▨	▨
1011	p	a	r	a	g	\0	▨	▨	▨	▨
1021	r	a	m	a	n	\0	▨	▨	▨	▨

Even though 10 bytes are reserved for storing the name 'akshay', it occupies only 7 bytes. Thus, 3 bytes go waste.

Built-in Library Functions

Function	Description
strlen (string)	Finds the length of a string.
strlwr(string)	Converts a string to lowercase.
strupr (string)	Converts a string to uppercase.
strcat (str1,str2)	Appends one string at the end of another.
strncat (str1,str2,n)	Appends first n character of a string at the end of another.
strcpy (str1,str2)	Copies a string into another.
strncpy(str1,str2,n)	Copies first n character of one string into another.
strcmp(str1,str2)	Compares two strings.
strncmp(str1,str2,n)	Compares first n characters of two strings.

Strings

- **The size of the array should be one byte more than the actual space occupied by the string since the compiler appends a null character at the end of the string.**
- **Example:**
  - char city[9] = “NEW YORK”;
    - Length of the character buffer shall be expected as length of the string+1.

Strings

- **strncpy function is not null terminated so always add null character after doing string manipulation**

  - Example

    char source[MAX] = "123456789"; char destination[MAX] = "abcdefg";

    strcpy(destination,source) ; // Output:destination is originally = 'abcdefg'

    After strcpy, destination becomes '123456789'

    strncpy(destination,source) ; // Output:destination is originally = 'abcdefg'

    After strcpy, destination becomes '12345fg‘

    /*the result will not be null-terminated. */

Strings

- **Characters and string are very different so strings function will not work for character.**
  - strcat(string, ‘|’) //not valid
  - strcat(string, “|”) //valid

Arrays

- **Cannot use const values in array dimensions, as in:**

```
const int size = 5;
    int mark[size];
```

  - The const qualifier really means ‘read-only’; an object so qualified is a run-time object that cannot be assigned to.
  - The value of a const qualified object is therefore not a constant expression in the full sense of the term and cannot be used for array dimensions.
  - When you need a true compile time constant, use a preprocessor #define.

Arrays

- **An extern array defined in one file and used in another.**
  - file1.c
    - int array[] = {1, 3, 5};
  - file2.c
    - extern int array[];

- **The sizeof operator will not work on extern array in file2.c. An extern array of unspecified size is an incomplete type. You cannot apply sizeof to it because sizeof operates at compile time and is not able to learn the size of an array that is defined in another file.**

Lab Session

- **Lab 4**

IGATE
Speed.Agility.Imagination

Summary

- **In this lesson, you have learnt:**
  - Arrays, whose elements are specified by one subscript, are called one-dimensional arrays.
  - The array size may be omitted during declaration.
  - Multidimensional Arrays are defined in much the same manner as single dimensional arrays, except that a separate pair of square brackets is required for each subscript.

Review Questions

- **Complete the following statement:**
  - Arrays, whose elements are specified by one subscript, are called as ______________.
- **True or false?**
  - The location of an array is the location of its last element.
    - True/False
  - The type of array is the data type of its elements.
    - True/False

Review – Match the Following

1. strncat	1. Converts a string to lowercase
2. strlen	2. Compares two strings
3. strcmp	3. Finds the length of a string
4. strlwr	4. Append first n character of a string at the end of another

IGATE
Speed.Agility.Imagination