**String Programs C**

Program 1:

The following program prompts user for the n numbers, once the user is done entering those numbers, this program sorts and displays them in ascending order. Here we have created a user defined function sort\_numbers\_ascending() for the sorting purpose.

/\*

\* C program to accept numbers as an input from user

\* and to sort them in ascending order.

\*/

#include <stdio.h>

void sort\_numbers\_ascending(int number[], int count)

{

int temp, i, j, k;

for (j = 0; j < count; ++j)

{

for (k = j + 1; k < count; ++k)

{

if (number[j] > number[k])

{

temp = number[j];

number[j] = number[k];

number[k] = temp;

}

}

}

printf("Numbers in ascending order:\n");

for (i = 0; i < count; ++i)

printf("%d\n", number[i]);

}

void main()

{

int i, count, number[20];

printf("How many numbers you are gonna enter:");

scanf("%d", &count);

printf("\nEnter the numbers one by one:");

for (i = 0; i < count; ++i)

scanf("%d", &number[i]);

sort\_numbers\_ascending(number, count);

}

Program 2:

In the following program we are initializing a variable (max\_element) with the first element of given array and then we are comparing that variable with all the other elements of the array using loop, whenever we are getting an element with the value greater than max\_element, we are moving that element to max\_element and moving further with the same approach to get the largest element in the max\_element variable.

#include <stdio.h>

/\* This is our function to find the largest

\* element in the array arr[]

\*/

int largest\_element(int arr[], int num)

{

int i, max\_element;

// Initialization to the first array element

max\_element = arr[0];

/\* Here we are comparing max\_element with

\* all other elements of array to store the

\* largest element in the max\_element variable

\*/

for (i = 1; i < num; i++)

if (arr[i] > max\_element)

max\_element = arr[i];

return max\_element;

}

int main()

{

int arr[] = {1, 24, 145, 20, 8, -101, 300};

int n = sizeof(arr)/sizeof(arr[0]);

printf("Largest element of array is %d", largest\_element(arr, n));

return 0;

}

Program 3:

This program calls the user defined function sum\_array\_elements() and the function calls itself recursively. Here we have hardcoded the array elements but if you want user to input the values, you can use a for loop and scanf function, same way as I did in the next section (Method 2: Using pointers) of this post.

#include<stdio.h>

int main()

{

int array[] = {1,2,3,4,5,6,7};

int sum;

sum = sum\_array\_elements(array,6);

printf("\nSum of array elements is:%d",sum);

return 0;

}

int sum\_array\_elements( int arr[], int n ) {

if (n < 0) {

//base case:

return 0;

} else{

//Recursion: calling itself

return arr[n] + sum\_array\_elements(arr, n-1);

}

}

Output:

Sum of array elements is:28

Program 4:

Here we are setting up the pointer to the base address of array and then we are incrementing pointer and using \* operator to get & sum-up the values of all the array elements.

#include<stdio.h>

int main()

{

int array[5];

int i,sum=0;

int \*ptr;

printf("\nEnter array elements (5 integer values):");

for(i=0;i<5;i++)

scanf("%d",&array[i]);

/\* array is equal to base address

\* array = &array[0] \*/

ptr = array;

for(i=0;i<5;i++)

{

//\*ptr refers to the value at address

sum = sum + \*ptr;

ptr++;

}

printf("\nThe sum is: %d",sum);

}

Output:

Enter array elements (5 integer values): 1 2 3 4 5

The sum is: 15

Program 5:

The formula that we are using to find the number of elements is common for all types of array. In this example, we have an array of double data type, however you can use the same logic for arrays of other data types like: int, float, long, char etc.

#include <stdio.h>

int main()

{

double arr[] = {11, 22, 33, 44, 55, 66};

int n;

/\* Calculating the size of the array with this formula.

\* n = sizeof(array\_name) / sizeof(array\_name[0])

\* This is a universal formula to find number of elements in

\* an array, which means it will work for arrays of all data

\* types such as int, char, float etc.

\*/

n = sizeof(arr) / sizeof(arr[0]);

printf("Size of the array is: %d\n", n);

return 0;

}

Output:

Size of the array is: 6