SINDH MADRESSATUL ISLAM UNIVERISTY, KARACHI

DEPARTMENT OF SOFTWARE ENGINEERING

FALL 2022

CSC103 - PROGRAMMING FUNDAMENTALS

ZUBAIR-UDDIN SHAIKH

SECTION SE1A/SE1B/SE1C/CS1D[e]

LAB MANUAL 08

ARRAYS AND STRINGS

ARRAYS AND STRINGS

1. Array

- An array is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier.

2. Declaring Arrays

- In order to declare our 4 variables, more specifically - integers, above using an array we write:

int nums[4];

- This sets up in memory 4 integers, i.e. 4 consecutive boxes in memory, each box holding an integer.
- We say that nums is an array of type integer or simply, an integer array.
- Each integer in the array is called an element of the array. In this array, each individual element is simply an integer.

3. Accessing Array Elements:

- Each array element is located at a unique index and hence we can access a particular element using the corresponding index. For an array of size n, the indices (plural of index) always go from 0 to n-1.
- Consider the same e.g. of nums which is an integer array of 4 elements.
- Since there are 4 elements (i.e. n=4) so the indices will be 0, 1, 2 and 3 and at each index there will be an integer value:

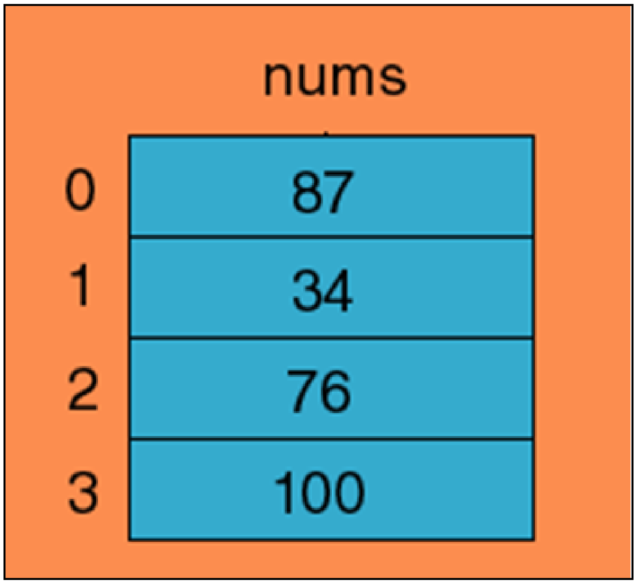
nums[0] = 87;

nums[1] = 34;

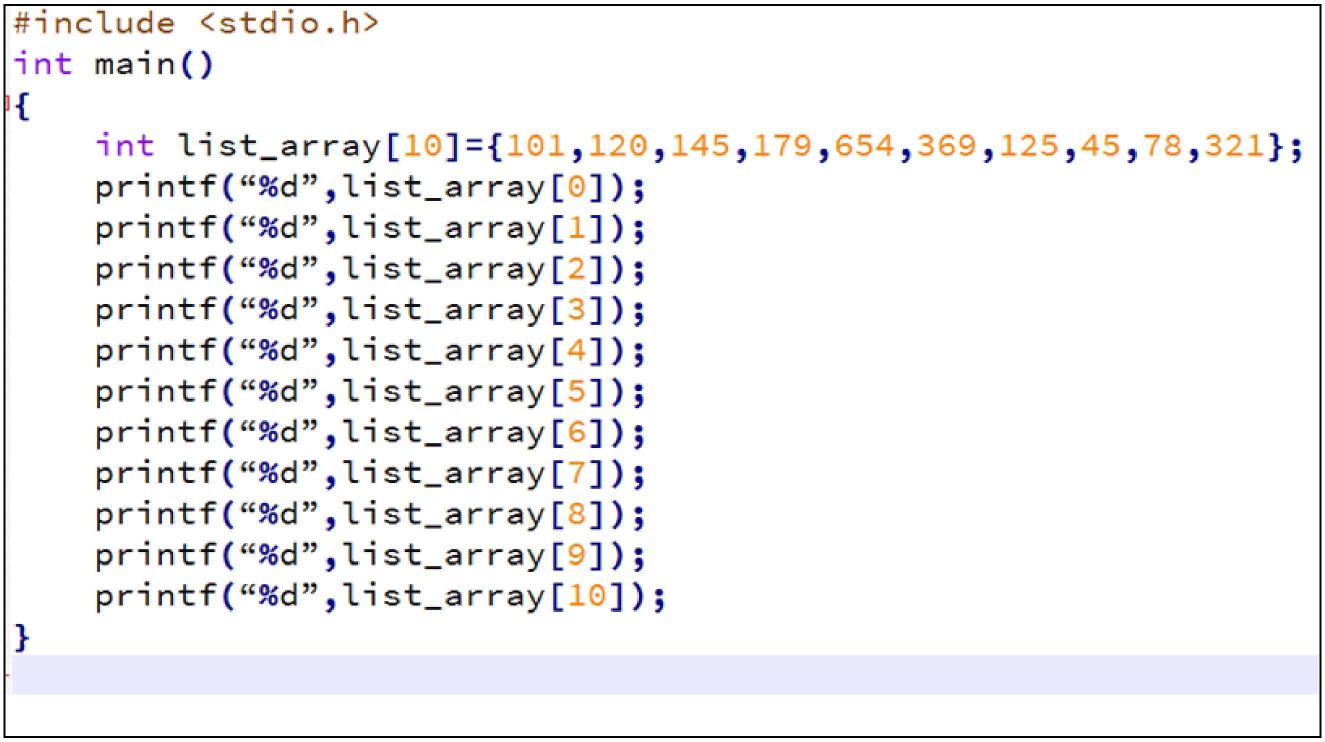
nums[2] = 76;

nums[3] = 100;

- Here, the number 87 is the first element of the array and is located at index 0. Similarly, 34 is the second element and is at index 1, and so on. So this is what'll be happening in memory:



Example 01:



```
#include <stdio.h>
int main()
{
    int list_array[10]={101,120,145,179,654,369,125,45,78,321};
    printf(“%d”,list_array[0]);
    printf(“%d”,list_array[1]);
    printf(“%d”,list_array[2]);
    printf(“%d”,list_array[3]);
    printf(“%d”,list_array[4]);
    printf(“%d”,list_array[5]);
    printf(“%d”,list_array[6]);
    printf(“%d”,list_array[7]);
    printf(“%d”,list_array[8]);
    printf(“%d”,list_array[9]);
    printf(“%d”,list_array[10]);
}
```

4. Loops and 1D Array:

- **Initializing Arrays:** Just like any other variable that is declared and not initialized, array elements also hold garbage values if not initialized. Initializing arrays can be done generally in two ways:

```
int nums[4] = {0,0,0,0};

                    OR

using a loop      // (for loop should be our choice, why?):

          int nums[4], i;
          for (i = 0 ; i < 4 ; i++)
          {
            nums[i] = 0;
          }
```

5. Taking Arrays as Input from the user:

- **Using for-loops to set arrays:** Set means to assign some value. Here we're taking it as input

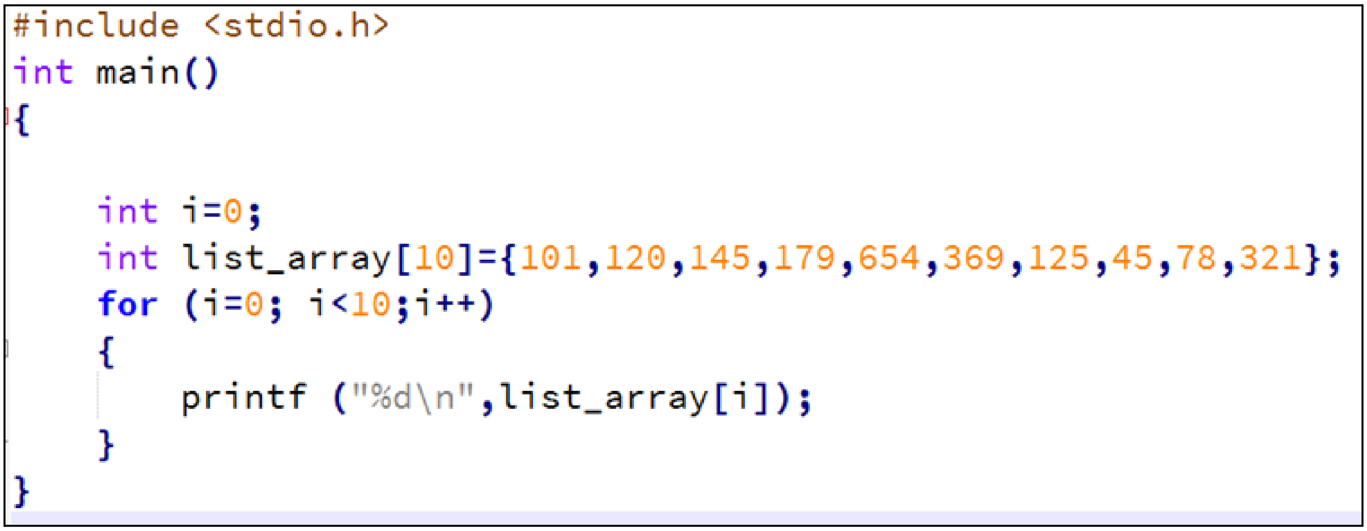
```
int nums[4], i;
for (i = 0 ; i < 4 ; i++)
{
  scanf("%d", &nums[i]);

}
```

- **Display/Print Arrays:** Using for-loops for printing arrays:

```
for (i = 0 ; i < 4 ; i++)
{
      printf ("%d ", nums[i]);
}
```

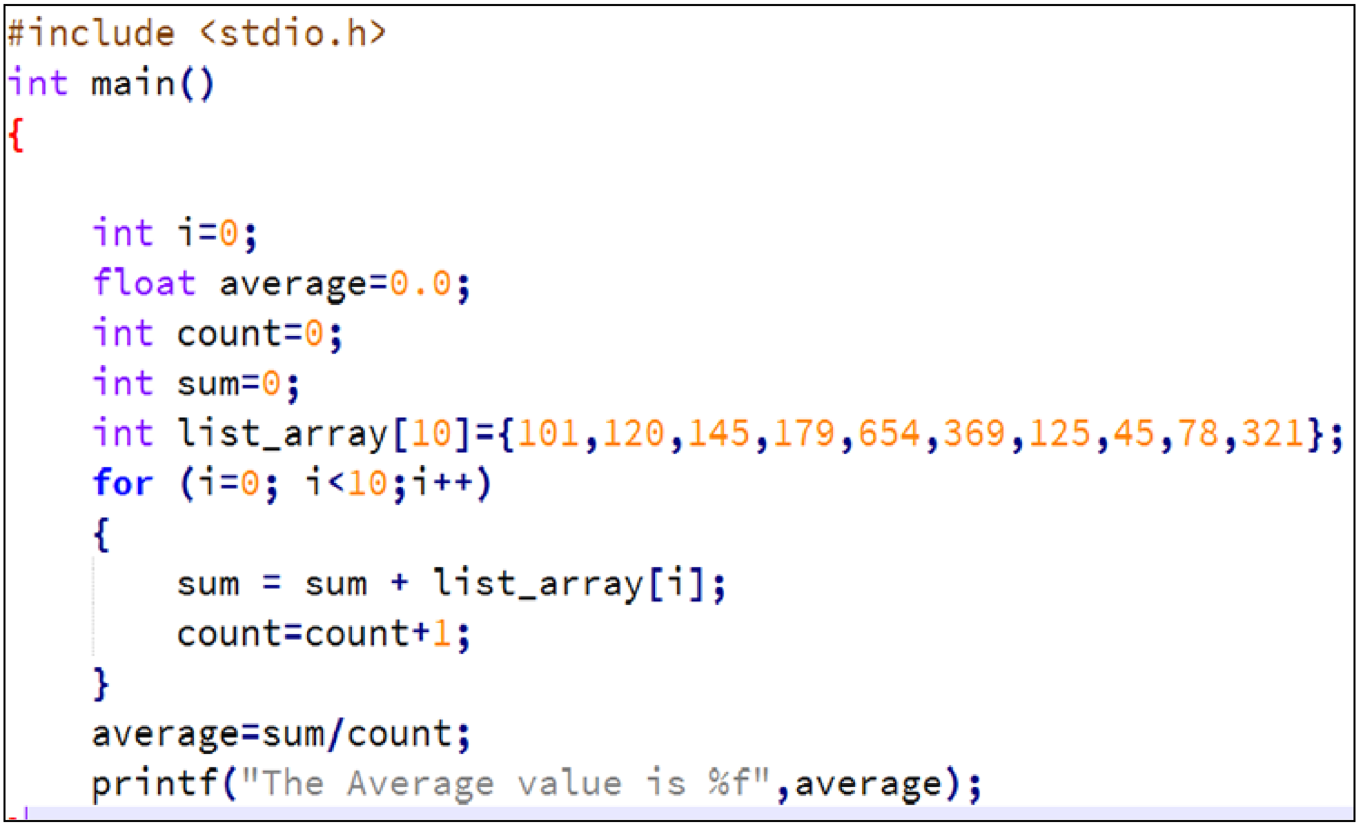
Example 02:



```c
#include <stdio.h>
int main()
{

    int i=0;
    int list_array[10]={101,120,145,179,654,369,125,45,78,321};
    for (i=0; i<10;i++)
    {
        printf ("%d\n",list_array[i]);
    }
}
```

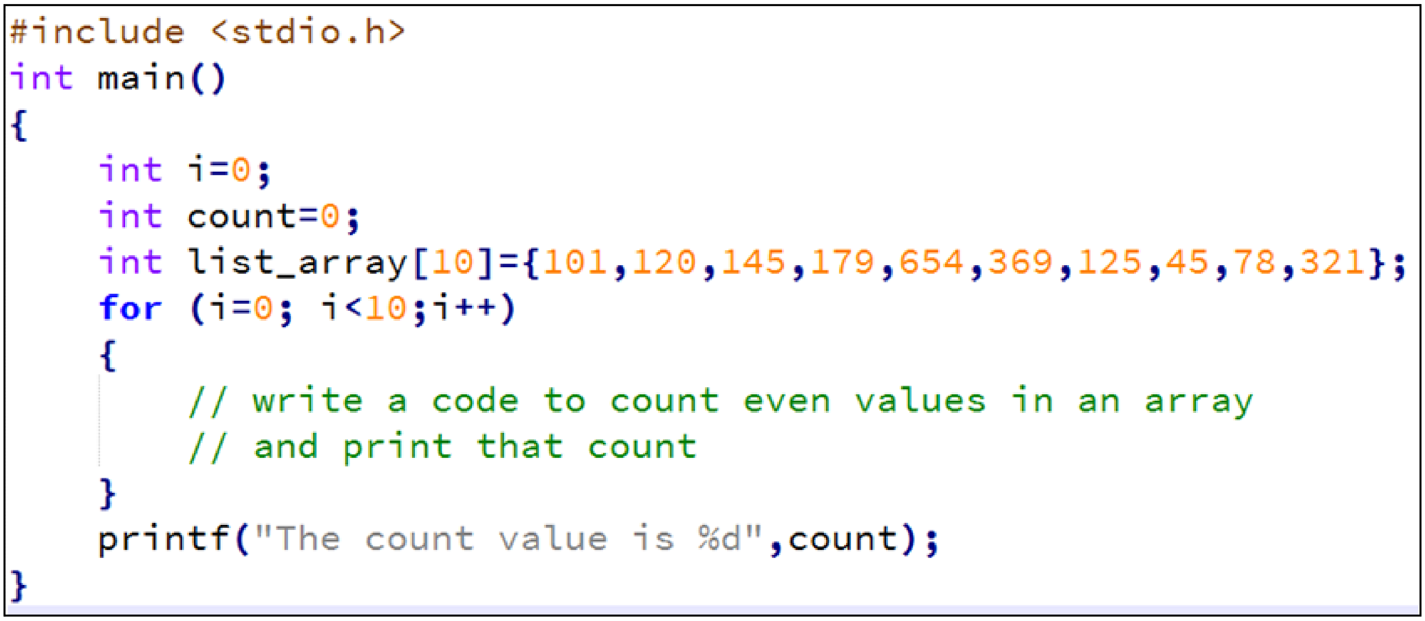
Example 03:



```c
#include <stdio.h>
int main()
{

    int i=0;
    float average=0.0;
    int count=0;
    int sum=0;
    int list_array[10]={101,120,145,179,654,369,125,45,78,321};
    for (i=0; i<10;i++)
    {
        sum = sum + list_array[i];
        count=count+1;
    }
    average=sum/count;
    printf("The Average value is %f",average);
```

Example 04:



```
#include <stdio.h>
int main()
{
    int i=0;
    int count=0;
    int list_array[10]={101,120,145,179,654,369,125,45,78,321};
    for (i=0; i<10;i++)
    {
        // write a code to count even values in an array
        // and print that count
    }
    printf("The count value is %d",count);
}
```

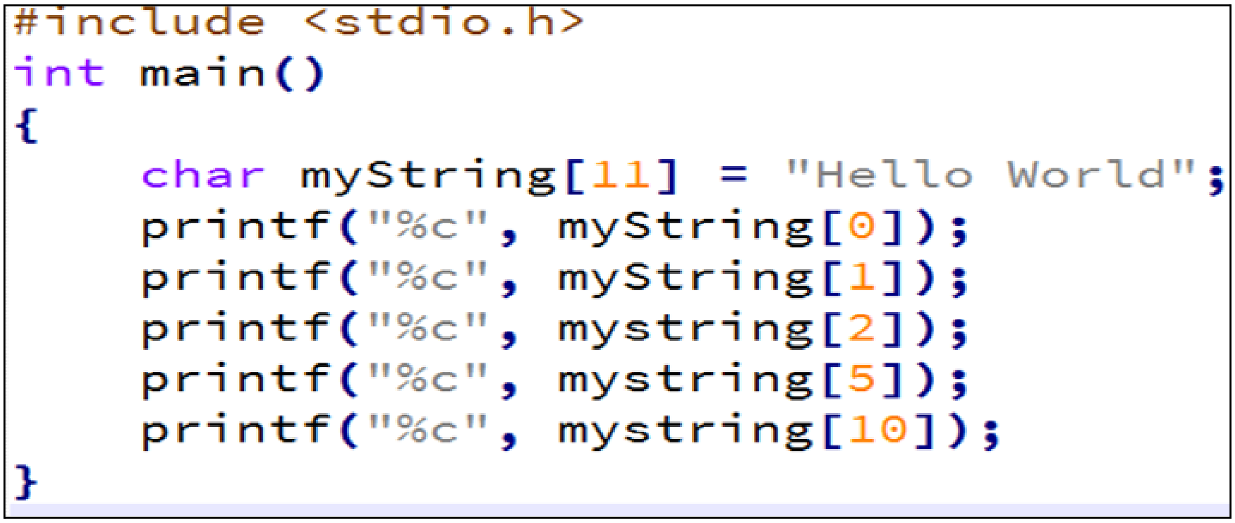
6. Strings:

- String is a collection of characters. We use strings to ease our work.
- There are ready-made functions, which will be allowed once you will have a complete understanding of how strings work.

char myString[11] = "Hello World";

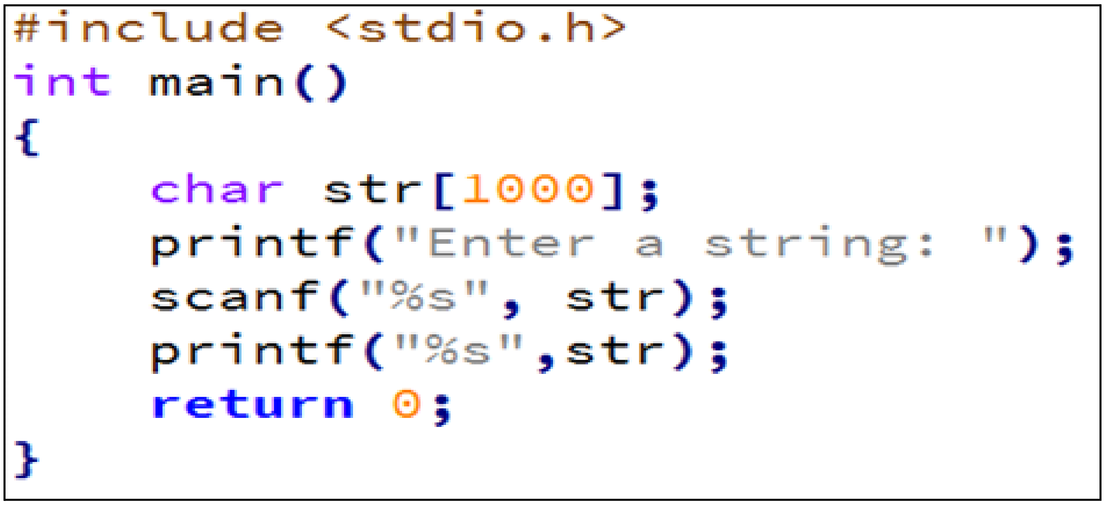
- myString is simply an array of type char whose value of first index(i.e. index 0) is ‘H’, and value of the element at index 10 is ‘d’.
- If you print myString[0] by using printf function then you will see H on your screen. Don’t forget that each string ends with the null character i.e. ‘\0’ whose decimal value is 0.
- Hence, myString[11] has that null character placed once we’ve initialized our string as shown above.

Example 05:



```
#include <stdio.h>
int main()
{
	char myString[11] = "Hello World";
	printf("%c", myString[0]);
	printf("%c", myString[1]);
	printf("%c", mystring[2]);
	printf("%c", mystring[5]);
	printf("%c", mystring[10]);
}
```

Example 06:

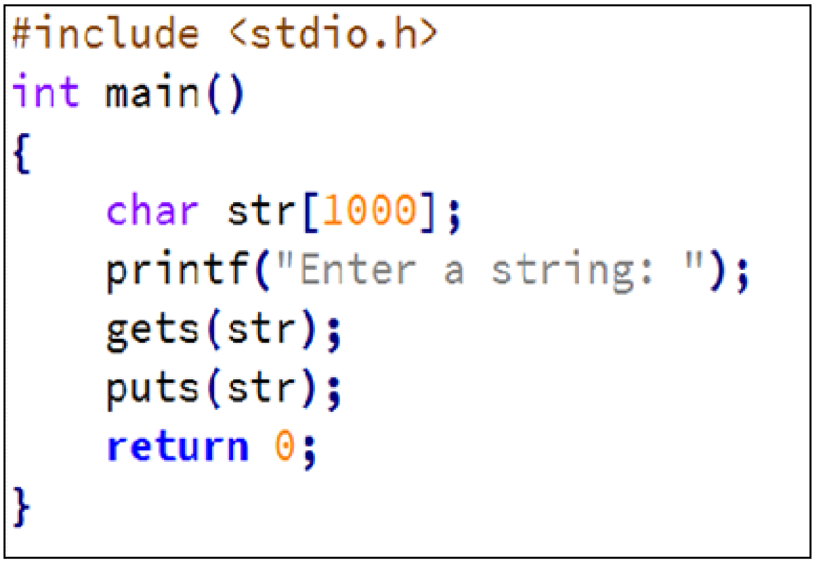


```
#include <stdio.h>
int main()
{
	char str[1000];
	printf("Enter a string: ");
	scanf("%s", str);
	printf("%s",str);
	return 0;
}
```

- As you can notice in the execution of the above code, string is read only until any whitespace.

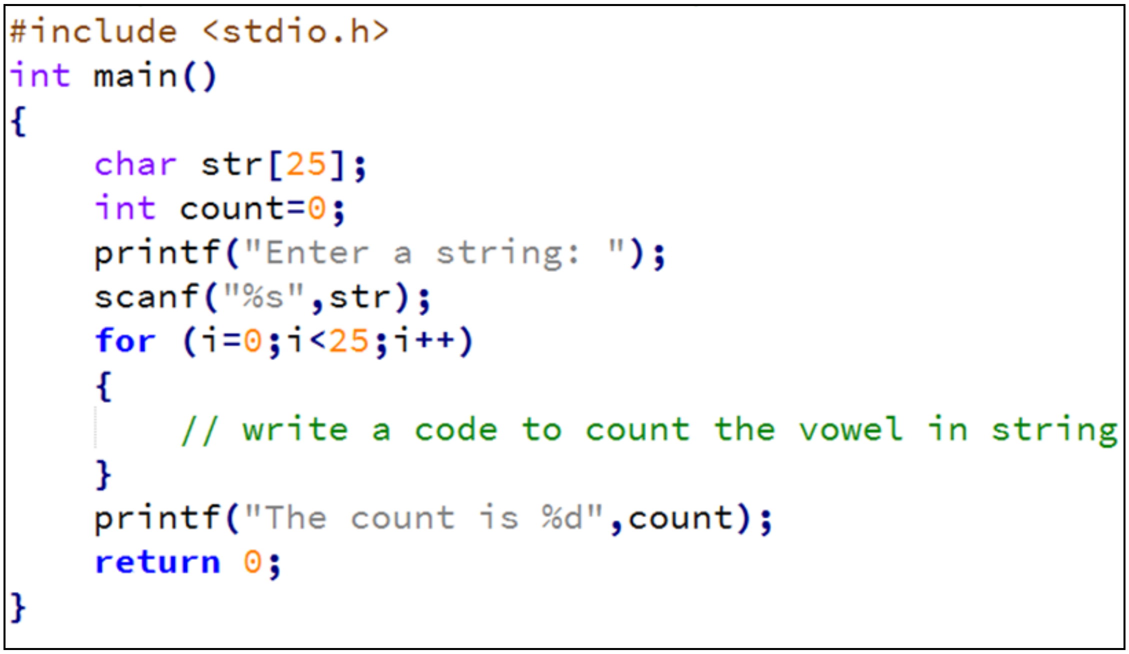
- To tackle that situation, commands named gets() and puts() are used for input and output, respectively.

Example 07:



```
#include <stdio.h>
int main()
{
    char str[1000];
    printf("Enter a string: ");
    gets(str);
    puts(str);
    return 0;
}
```

Example 08:



```
#include <stdio.h>
int main()
{
    char str[25];
    int count=0;
    printf("Enter a string: ");
    scanf("%s",str);
    for (i=0;i<25;i++)
    {
        // write a code to count the vowel in string
    }
    printf("The count is %d",count);
    return 0;
}
```

Lab Task01: Write a C code to calculate the length of a user-entered string.

Lab Task 08:

1. **Write a program that takes 10 integer numbers as array input and print the median of that array list.**
2. **Write a program that takes 10 numbers (double data type) as array input. Perform the following tasks.**
   a. **Find Maximum Number**
   b. **Find Minimum Number**
3. **Write a C program that inputs a line from the user and print it in uppercase. (Assume all characters are in lowercases. Only give input in lower case)**
4. **Write a program that inputs email id from users and finds whether the email address includes the '@' character or not.**

Submission Instructions:

Due Date: Nov 17, 2022

1. **For C files, name your C files as questionNumber_yourRollNum_yourSection_LTNumber.c (e.g. Q1_BSE-22F-123_SE1A_LT1.c).**
2. **Place all files in a folder and name the folder as yourRollNum_yourSection_LTNumber (e.g. BSE-22F-123_SE1A_LT1).**
3. **Compress the folder by using either Winrar or 7Zip with the same name.**
4. **Go to tiny.cc/pffall2022smiu and in the "Coordination Document Folder" open the "PF-Activity Submission Form".**
5. **Fill out all the details with your correct password and submit the form by the due date.**