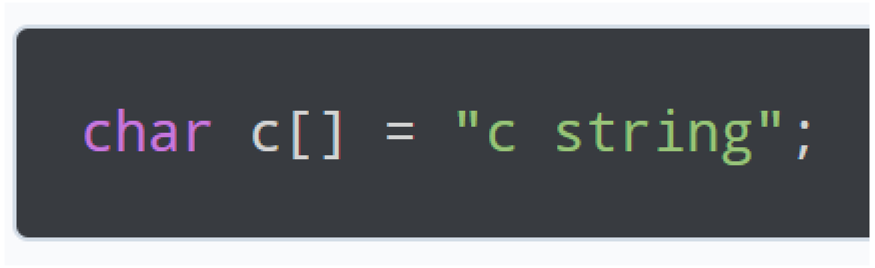
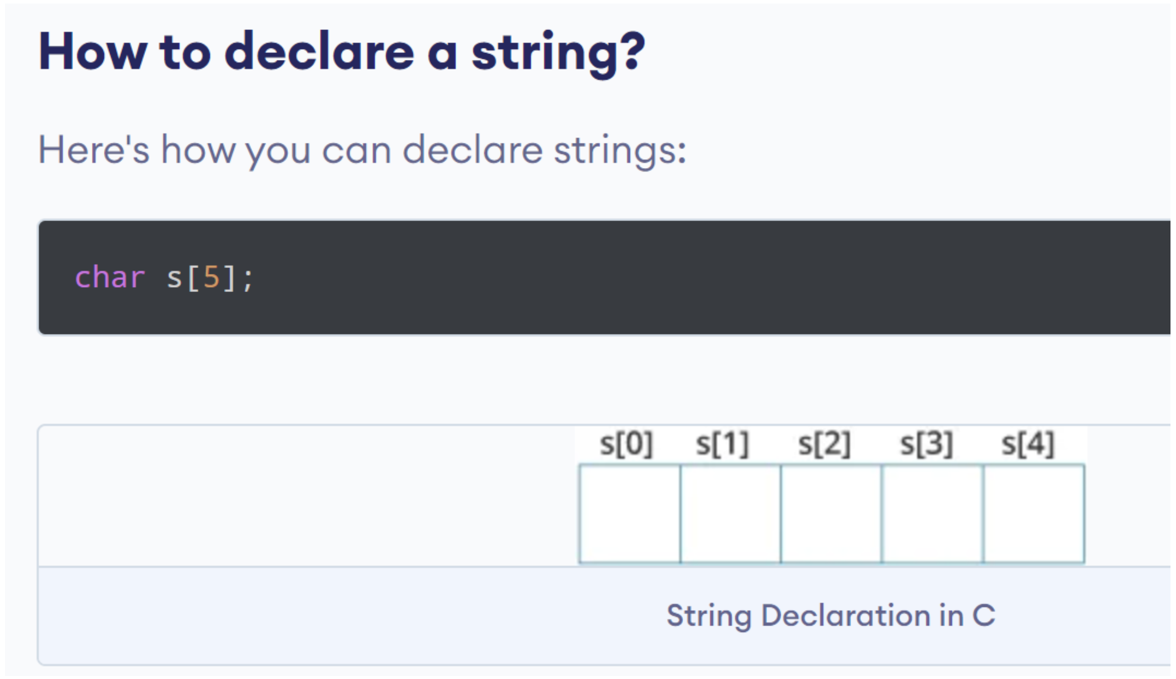
C Programming Strings

C Programming Strings



```c
char c[] = "c string";
```



How to declare a string?

Here's how you can declare strings:

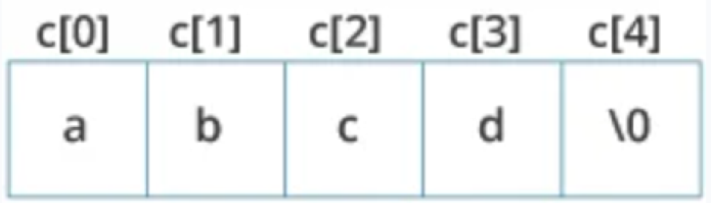
```c
char s[5];
```

String Declaration in C

How to initialize strings?

You can initialize strings in a number of ways.

```
char c[] = "abcd";

char c[50] = "abcd";

char c[] = {'a', 'b', 'c', 'd', '\0'};

char c[5] = {'a', 'b', 'c', 'd', '\0'};
```



String Initialization in C

Assigning Values to Strings

Arrays and strings are second-class citizens in C; they do not support the assignment operator once it is declared. For example,

```c
char c[100];
c = "C programming";  // Error! array type is not assignable.
```

Note: Use the strcpy() function to copy the string instead.

Read String from the user

You can use the `scanf()` function to read a string.

The `scanf()` function reads the sequence of characters until it encounters whitespace (space, newline, tab, etc.).

Example 1: scanf() to read a string

```c
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter name: ");
    scanf("%s", name);
    printf("Your name is %s.", name);
    return 0;
}
```

Output

```
Enter name: Dennis Ritchie
Your name is Dennis.
```

How to read a line of text?

You can use the `fgets()` function to read a line of string. And, you can use `puts()` to display the string.

Example 2: fgets() and puts()

```c
#include <stdio.h>
int main()
{
    char name[30];
    printf("Enter name: ");
    fgets(name, sizeof(name), stdin);  // read string
    printf("Name: ");
    puts(name);    // display string
    return 0;
}
```

Output

```
Enter name: Tom Hanks
Name: Tom Hanks
```

Passing Strings to Functions

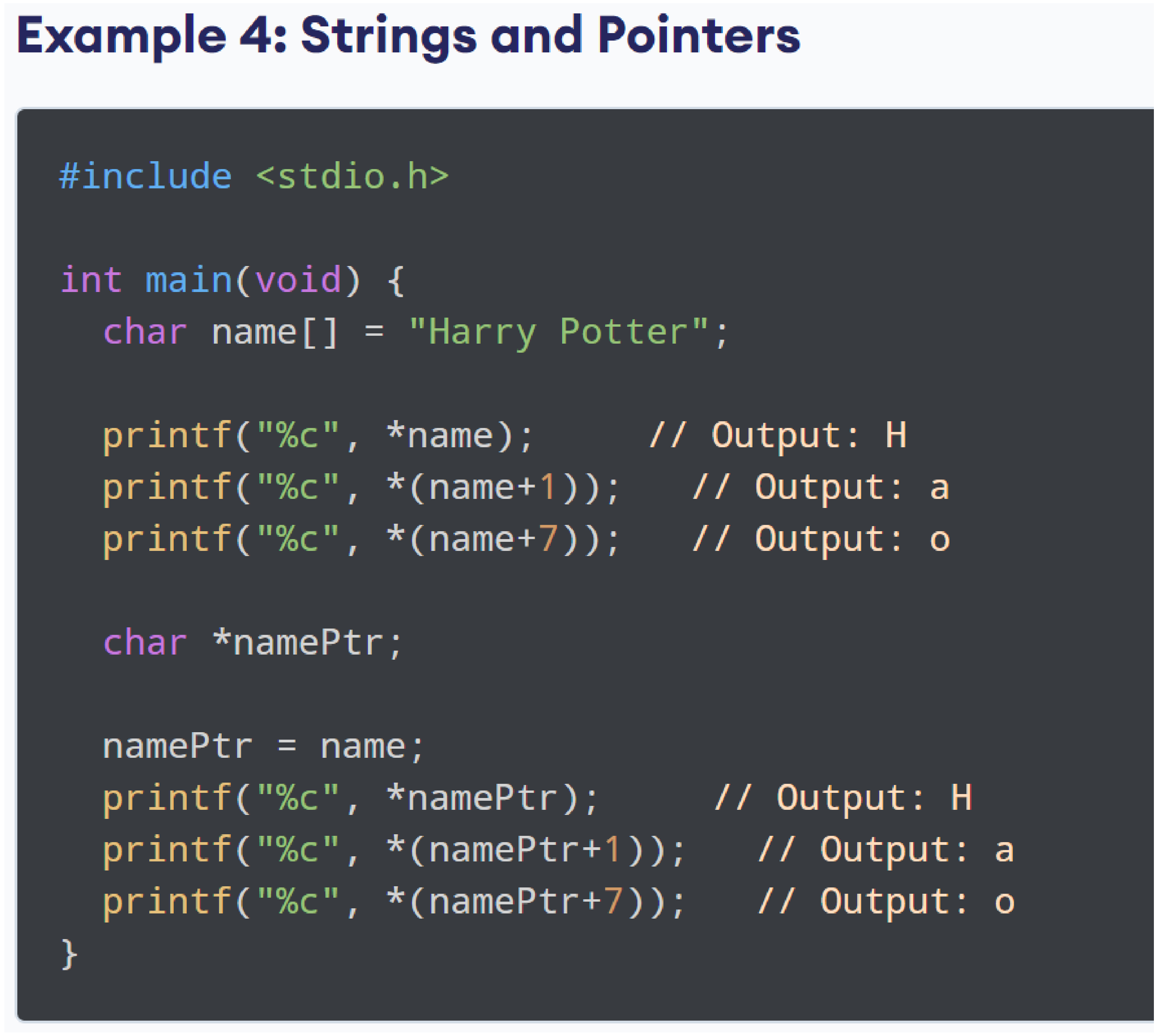
Strings can be passed to a function in a similar way as arrays. Learn more about passing arrays to a function.

Example 3: Passing string to a Function

```c
#include <stdio.h>
void displayString(char str[]);

int main()
{
    char str[50];
    printf("Enter string: ");
    fgets(str, sizeof(str), stdin);
    displayString(str);     // Passing string to a function.
    return 0;
}
void displayString(char str[])
{
    printf("String Output: ");
    puts(str);
}
```

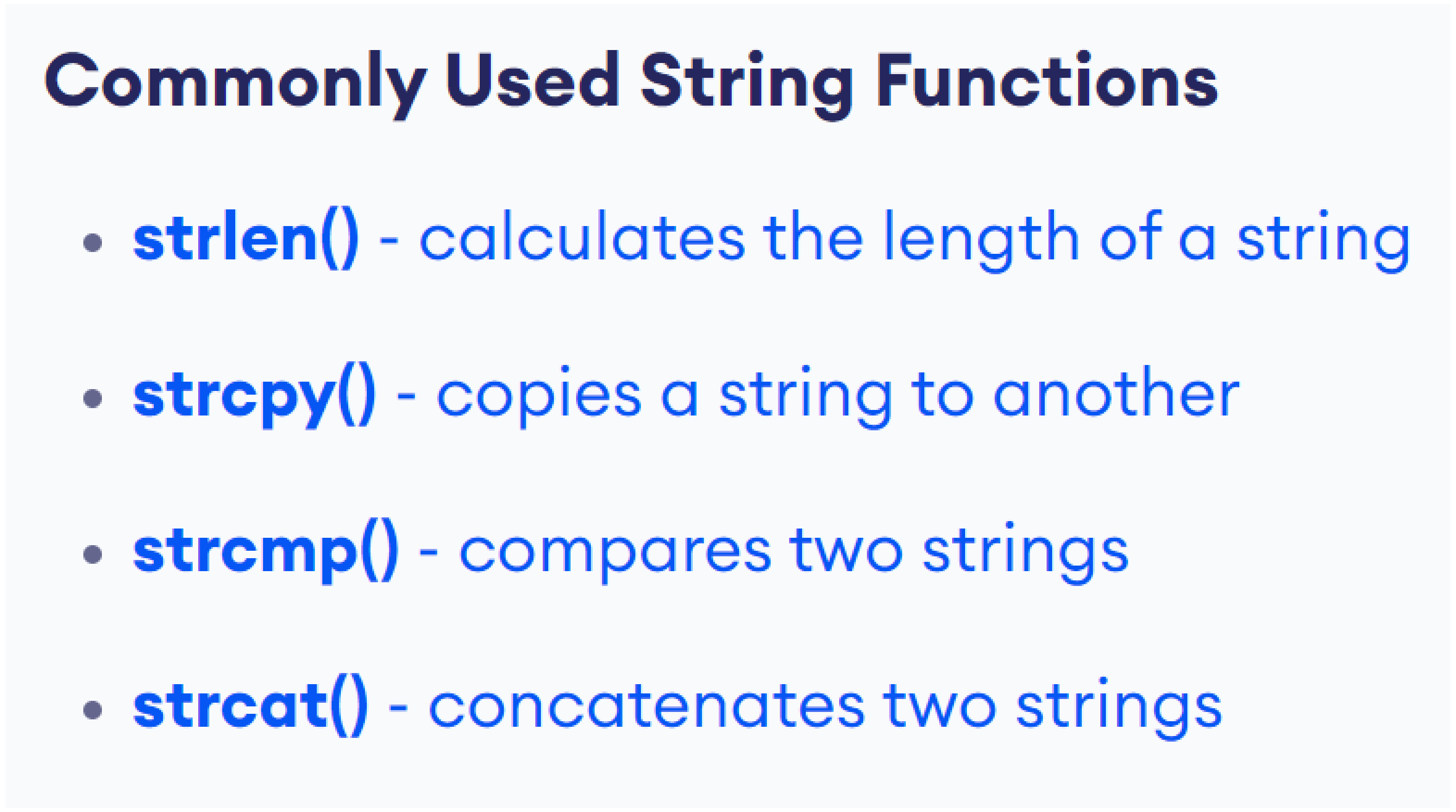
Strings and Pointers

Similar like arrays, string names are "decayed" to pointers. Hence, you can use pointers to manipulate elements of the string. We recommended you to check C Arrays and Pointers before you check this example.



Example 4: Strings and Pointers

```c
#include <stdio.h>

int main(void) {
  char name[] = "Harry Potter";

  printf("%c", *name);     // Output: H
  printf("%c", *(name+1));   // Output: a
  printf("%c", *(name+7));   // Output: o

  char *namePtr;

  namePtr = name;
  printf("%c", *namePtr);     // Output: H
  printf("%c", *(namePtr+1));   // Output: a
  printf("%c", *(namePtr+7));   // Output: o
}
```



Commonly Used String Functions

- **strlen()** - calculates the length of a string
- **strcpy()** - copies a string to another
- **strcmp()** - compares two strings
- **strcat()** - concatenates two strings

Example: C strlen() function

```c
#include <stdio.h>
#include <string.h>
int main()
{
    char a[20]="Program";
    char b[20]={'P','r','o','g','r','a','m','\0'};

    // using the %zu format specifier to print size_t
    printf("Length of string a = %zu \n",strlen(a));
    printf("Length of string b = %zu \n",strlen(b));

    return 0;
}
```

Output

```
Length of string a = 7
Length of string b = 7
```

Note that the `strlen()` function doesn't count the null character `\0` while calculating the length.

Example: C strcpy()

```c
#include <stdio.h>
#include <string.h>

int main() {
  char str1[20] = "C programming";
  char str2[20];

  // copying str1 to str2
  strcpy(str2, str1);

  puts(str2); // C programming

  return 0;
}
```

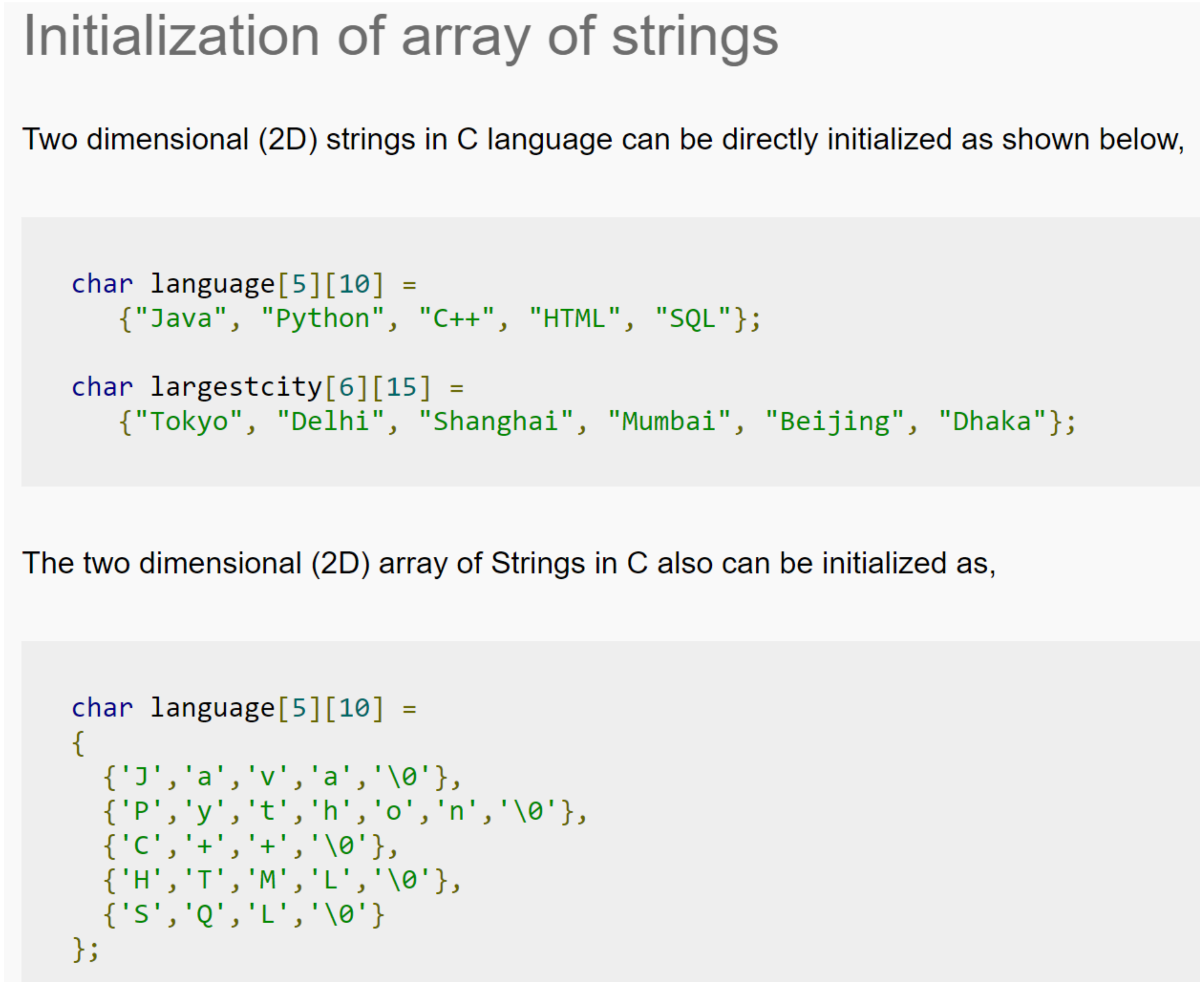
Output

```
C programming
```

Two Dimensional (2D) Array of Strings in C

Sometimes we often use lists of character strings, such as the list of names of students in a class, a list of names of employees in an organization, a list of places, etc. A list of names can be treated as a table of strings. To store the entire list we use a 2d array of strings in C language.

The array of characters is called a string. “Hi”, “Hello”, and e.t.c are examples of String. Similarly, the array of Strings is nothing but a two-dimensional (2D) array of characters. To declare an array of Strings in C, we must use the char data type.



Initialization of array of strings

Two dimensional (2D) strings in C language can be directly initialized as shown below,

```
char language[5][10] =
   {"Java", "Python", "C++", "HTML", "SQL"};

char largestcity[6][15] =
   {"Tokyo", "Delhi", "Shanghai", "Mumbai", "Beijing", "Dhaka"};
```

The two dimensional (2D) array of Strings in C also can be initialized as,

```
char language[5][10] =
{
  {'J','a','v','a','\0'},
  {'P','y','t','h','o','n','\0'},
  {'C','+','+','\0'},
  {'H','T','M','L','\0'},
  {'S','Q','L','\0'}
};
```

Since it is a two-dimension of characters, so each String (1-D array of characters) must end with null character i.e. ‘\0’

Each String in the above array can be accessed by using its index number. The index of the array always starts with 0.

```
language[0] => "Java";
language[1] => "Python";
language[2] => "C++";
language[3] => "HTML";
language[4] => "SQL";
```

Note1:- the number of characters (column-size) must be declared at the time of the initialization of the two-dimensional array of strings.

```
// it is valid
char language[ ][10] = {"Java", "Python", "C++", "HTML", "SQL"};
```

But that the following declarations are **invalid**.

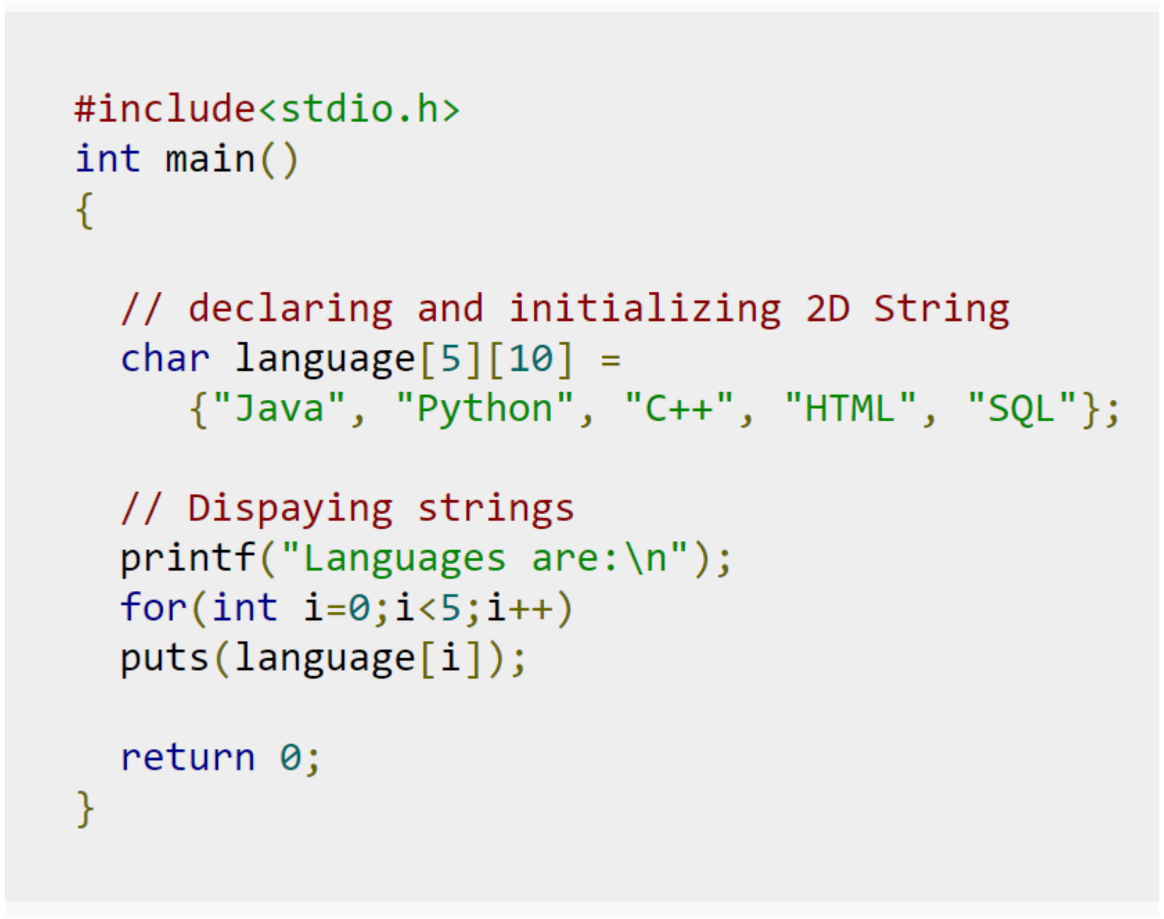
```
// invalid
char language[ ][ ] = {"Java", "Python", "C++", "HTML", "SQL"};

// invalid
char language[5][ ] = {"Java", "Python", "C++", "HTML", "SQL"};
```

```c
char language[5][10] = {"Java", "Python", "C++", "HTML", "SQL"};

// now, we can't directly assign a new String
language[0] = "Kotlin"; // invalid

// we must copy the String
strcpy(language[0], "Kotlin"); // valid
// Or,
scanf(language[0], "Kotlin"); // valid
```

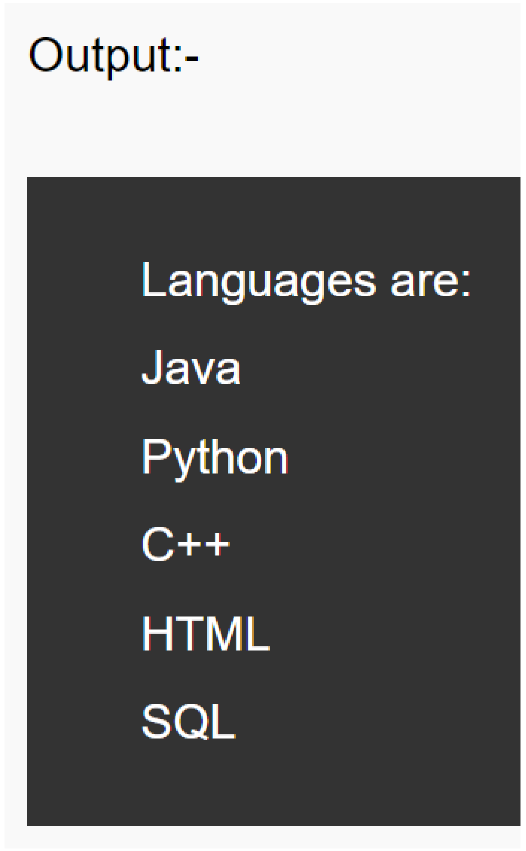
Reading and displaying 2d array of strings in C

- Reading and displaying 2d array of strings in C
- The two-dimensional array of strings can be read by using loops. To read we can use scanf(), gets(), fgets() or any other methods to read the string.

Sample program to read the 2D array of characters or array of String in C



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

  // declaring and initializing 2D String
  char language[5][10] =
     {"Java", "Python", "C++", "HTML", "SQL"};

  // Dispaying strings
  printf("Languages are:\n");
  for(int i=0;i<5;i++)
  puts(language[i]);

  return 0;
}
```



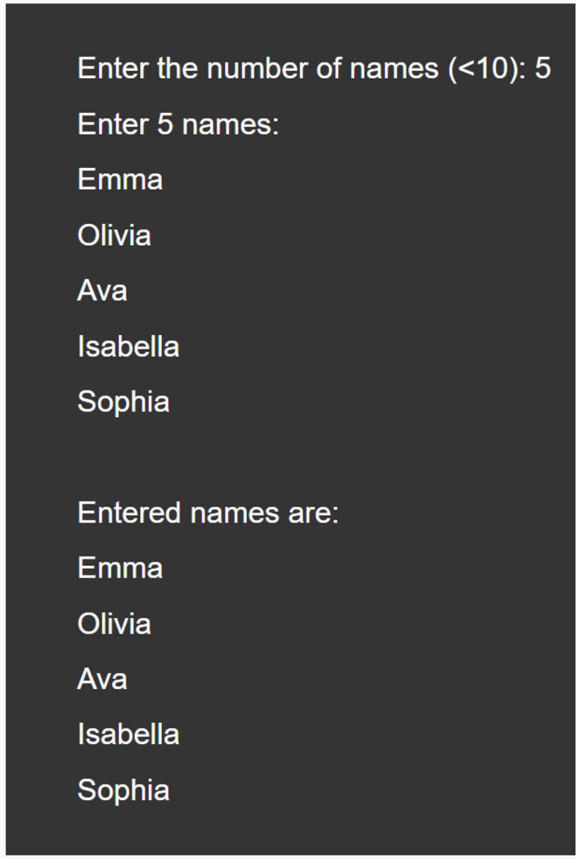
Output:-

```
Languages are:
Java
Python
C++
HTML
SQL
```

Program:- Write a program to read and display a 2D array of strings in C language.

```c
#include<stdio.h>
int main()
{
     char name[10][20];
     int i,n;

     printf("Enter the number of names (<10): ");
     scanf("%d",&n);

     // reading string from user
     printf("Enter %d names:\n",n);
     for(i=0; i<n; i++)
     scanf("%s[^\n]",name[i]);

     // dispaying strings
     printf("\nEntered names are:\n");
     for(i=0;i<n;i++)
     puts(name[i]);

     return 0;
}
```

Output:-



```
Enter the number of names (<10): 5
Enter 5 names:
Emma
Olivia
Ava
Isabella
Sophia

Entered names are:
Emma
Olivia
Ava
Isabella
Sophia
```