université de science et technologie houari boumediene

TP 5 : Arrays in C

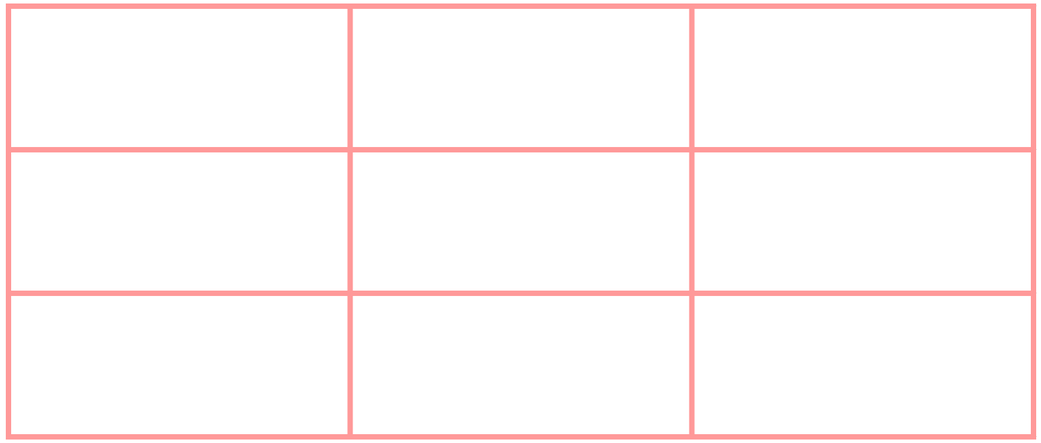
beldjouziasmaa@gmail.com

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Arrays

- An array is a finite set of elements of the same type, stored in memory at contiguous addresses.



suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

One-Dimensional Array - Vector

- **Declaration**

```
<type> <arrayname> [<array-length>];
```

- where <array-length> must be a positive integer constant expression , and t <type> is the type of the array elements.
- **Examples:**
- int V[10]; /* V: array of 10 integers */
- long V1[10] ; /* V1: array of 10 long integers */
- float V2[100]; /* V2: array of 100 reels float */
- double V3[100]; /* V3: array of 100 double reels */
- char ch[20] ; /* ch: array of 20 characters */

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

One-Dimensional Array - Vector

- **Initialization :**

Examples **:** int V[5] = {10, 20, 30, 40, 50};
float V2[4] = {-1.05, 3.33, 87e-5, -12.3E4};
int V3[10] = {1, 0, 0, 1, 1, 1, 0, 1, 0, 1};

- **Noticed :**
- The number of values in the list must correspond to the size of the array. If the list does not contain enough values for all components, the remaining components are initialized with zero.
- If the dimension is not explicitly specified during initialization, then the necessary number of bytes is reserved automatically .
- **Example :**
- Int V[ ] = {10, 20, 30, 40, 50}; ==> Reservation of 5*size of an integer

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

One-Dimensional Array - Vector

- **Access to Array Elements :**

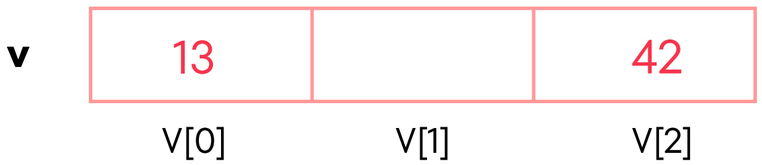
Access to the elements of the array is:

```
<arrayname>[index]
```

- By declaring an array as: int V[100];
- We defined an array V with 100 elements, which can be accessed by: V[0], V[1], ..., V[99]

An array in C is always indexed from index 0.

- Example :
- int V[3];
- V[0] = 13; // put value 13 in the 1st element of the array
- V[3] = 42; // put value 42 in the last element of the array



suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

One-Dimensional Array - Vector

Application

/ Displaying and Reading elements of a one-dimensional array */*

```c
main()
{
   int V[5];
   int i;

   for (i=0; i<5; i++)
       scanf("%d", &V[i]);

   for (i=0; i<5; i++)
      printf("%d ", V[i]);
}
```

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

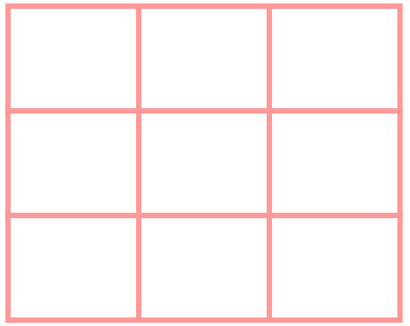
ADRESSE EMAIL
beldjouziasmaa@gmail.com

Two-Dimensional Array - Matrix

Declaration

```
<type> <arrayname> [<number-of-rows>][<number-of-columns>];
```

- where <number-of-rows> and <number-of-columns> must be positive integer constant expressions , and t <type> is the type of the elements of the matrix .
- **Examples:**
- int M[5][10]; /* M: matrix of 5 rows and 10 columns of integers */
- long M1[10][10]; /* M1: matrix of 10 rows and 10 columns of long integers */
- float M2[20][100]; /* M2: matrix of 20 rows and 100 columns of float reels */
- double M3[15][10]; /* M3: matrix of 15 rows and 10 columns of double reels */
- char ch[12][20]; /* ch: matrix of 12 lines and 20 columns of characters */



suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Two-Dimensional Array - Matrix

Memorization :

- The components of a two-dimensional array are stored line by line in memory.
- **Initialization :**
- ***Examples* :**

```
int M[3][10] ={ { 0, 10, 20, 30, 40, 50, 60, 70, 80, 90},
                {10, 11, 12, 13, 14, 15, 16, 17, 18,19},
                    {1, 12, 23, 34, 45, 56, 67, 78, 89, 90}
                };
```

suivant →

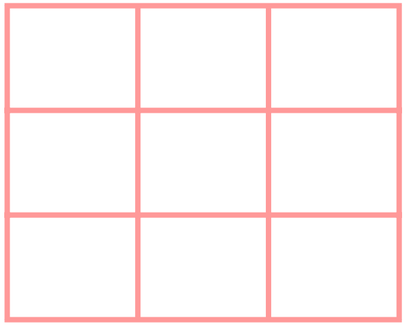
BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Two-Dimensional Array - Matrix

- **Access to the Elements of a Matrix**
- Access to the elements of the matrix is:

```
<matrixname> [rowindex][columnindex];
```



- By declaring a matrix by: int M[5][10];
- We have defined an array (matrix) M with 5 rows and each row contains 10 components, which can be accessed by:
- M[0][0], M[0][1], ..., M[0][9], ... , M[4][0],...,M[4][9]

Example :

int M[5][10];
M[0][0] = 15; // put the value 15 in the 1st element of the first row of the matrix.
M[2][9] = 42; // put the value 42 in the last element of the 3rd row of the matrix

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Two-Dimensional Array - Matrix

Application:: Displaying the elements of a two-dimensional array (Matrix)

```c
main()

{

    long A[10][20];
    int i,j;
    /* For each line ... */
    for (i=0; i<10; i++)
    {
        for (j=0; j<20; j++) /* consider each
 component*/
                printf("%d ", A[i][j]);
        printf("\n"); /* Line break */
    }
 }
```

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Two-Dimensional Array - Matrix

Application:: Displaying the elements of a two-dimensional array (Matrix)

```c
main()

{

    long A[10][20];
    int i,j;
    /* For each line ... */
    for (i=0; i<10; i++)
    {
        for (j=0; j<20; j++) /* consider each
 component*/
                printf("%d ", A[i][j]);
        printf("\n"); /* Line break */
    }
 }
```

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Character Strings

There is no special string type in C. A character string is treated as a one-dimensional array of characters (character vector).

Declaration

```
char <StringName> [<length>];
```

Examples :
char NAME[20];
char FIRST NAME[20];
char PHRASE[300];

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Character Strings

Note: When declaring, we need to indicate the space to be reserved in memory for storing the string. The internal representation of a character string is terminated by the symbol '\0' (NULL). So, for a text of n characters, we must provide n+1 bytes.
The length of the character string is the maximum size (minus 1) of the string. If we put fewer characters than the length in the character string, the character '\0' is put after the last character (see initialization below)

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Character Strings

Initialization :

Examples:

char firstname1 [8] = "OMAR";

will be represented in Memory as shown below:

'O'	'M'	'A'	'R'	'\0'	x	x	x

char firstname2 [ ] = "OMAR";

will be represented in Memory as shown below:

'O'	'M'	'A'	'R'	'\0'

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Character Strings

When initialized with [ ] , the computer automatically reserves the number of bytes necessary for the string, i.e.: the number of characters + 1 for the end of string mark (here: 5 bytes). We can also explicitly indicate the number of bytes to reserve, if this is greater than or equal to the length of the initialization string.
We can declare a string with a certain length and then initialize it using the strcpy function as shown below:
char firstname3[8]
strcpy(firstname3, "ALI");
and in this case firstname3 will be represented in memory as:

'A'	'L'	'I'	'\0'	x	x	x	x

strcpy() is a function that is part of the **string.h** library

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Character Strings

Library of string.h character strings :

Fonction	Description	Exemple	Résultat
strlen(ch)	Renvoie la taille (nombre de caractères) de la chaîne ch (sans inclure le caractère nul).	char string[10]; strcpy(string, "Omar"); int len = strlen(string);	len = 4
strcpy(dest, src)	Copie la chaîne src dans la chaîne dest.	char src[10] = "example"; char dest[10]; strcpy(dest, src);	dest = "example"
strcmp(s1, s2)	Compare deux chaînes s1 et s2. Retourne 0 si elles sont identiques, >0 si s1 > s2, <0 sinon.	char *s1 = "abc"; char *s2 = "abd"; int value = strcmp(s1, s2);	value = -1 (car "abc" est inférieur à "abd").
strstr(s1, s2)	Vérifie si s2 est une sous-chaîne de s1. Retourne un pointeur vers le début de s2 dans s1.	char s1[10] = "example"; char s2[5] = "amp"; if (strstr(s1, s2)) printf("s2 est une sous-chaîne");	Affiche "s2 est une sous-chaîne".
strncpy(dest, src, n)	Copie au plus n caractères de src dans dest.	char src[10] = "example"; char dest[5]; strncpy(dest, src, 4); dest[4] = '\0';	dest = "exam"
strncmp(s1, s2, n)	Compare au plus n caractères des chaînes s1 et s2.	char *s1 = "abc"; char *s2 = "abd"; int value = strncmp(s1, s2, 2);	value = 0 (car les 2 premiers caractères des deux chaînes sont identiques).

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Character Strings

Example :

Program that reverses a character string given by the user.

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

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

int main()
{
    char string[1000]; // string is a local 1000 char array
    int i, j, len;
    char temp;
    printf("Please enter a string ...\n");
    scanf("%s", string);
    len = strlen(string);
    /*
        i starts from the beginning and increments
        j start from the end and decrease
        i and j interchange their characters until they meet
     */
     j = len - 1;
     for (i = 0; i < j; i++)
     {
        temp = string[i];
        string[i] = string[j];
        string[j] = temp;
        j--;
     }

     printf("The reverse string is: %s \n", string);

     system("pause");
     return 0;
}
```

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

String Arrays

Example :
Program that reverses a character string given by the user.
A string array corresponds to a two-dimensional array of type char , where *each row contains a character string* .
Declaration :
The char mois declaration[12][10]; Reserves space in memory for 12 words containing 10 characters (including 9 significant characters).

Initialization

```
char month[12][10] = {"January", "February", "March", "April", "May", "June",
                      "July", "August", "September", "October", "November", "December"};
```

Access to Character Strings
It is possible to access the different *character strings* in an array, by simply indicating the corresponding line:
month[4]->"may"

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Applications

Application 1

Consider a Tab vector (one-dimensional array) containing N integers (N<=100). Write a program that calculates the product of all the elements of Tab as well as the number of strictly positive values.

Application 2

Given a square matrix A(N, N) of integers (N<=25), write a program which allows you to calculate the trace of the matrix A. The trace is the sum of the elements of the main diagonal.

Application 3

Given a matrix A(N, M) of integers (N<=20 and M<=30), write a program which:

- calculates and saves the sum of each column,
- determines the jmin position of the minimum sum and the jmax position of the maximum sum.
- permutes the two columns of indices jmin and jmax of matrix A if jmin>jmax.

suivant →

BELDJOUZI ASMAA
ingénieure en informatique

ADRESSE EMAIL
beldjouziasmaa@gmail.com

Applications

Application 1

Write a program that checks if a character string is a palindrome.

A palindrome is a word that is read left to right and right to left (that is, the word is read both ways).

Example of palindrome words: RADAR, HERE, SHE,

Application 2

Consider a CH character string terminated by point '.'. Write a program that:

- displays the number of characters of CH, i.e. the size of CH;
- displays the number of lowercase alphabetic characters;
- displays the number of uppercase alphabetic characters.

(We assume, in this exercise, that we do not have the size function which gives the size of a character string)

Application 3

Let T1(N<=10) and T2(M<=15) be two arrays of character strings.

Write a program that displays words from T1 that do not exist in T2.

suivant →