التاريخ: / /

الموضوع: _____

Lec 13

How to Pass an array as Parameter in C?

Parameters as a Pointers

Example:-

```
unsigned int Numbers[5]={11,12,13,14,15}
unsigned char Print_Number(const unsigned int *mynum,
                           const unsigned int length)
{
 unsigned char RetVal=0;
 unsigned int NumbersIndex=0;

 if(NULL==Mynum)
 {
   RetVal=1;
 }else{
 For(Numbers Index=0; NumbersIndex< length, NumberIndex++)
 {
  printf("%i\t", Mynum[NumberIndex]);
 }
 return Retval;
}
```

* في حالة لو أنا استخدمت الطريقة السابقة لازم اشوف هل محتويات الـ array هتتغير ولا لا لو مش هتتغير بكتب const قبل نوع الداتا تايب

* تاني حاجة لازم اعمل Validate على البوينتر لأنه الـ User ممكن يبعت NULL فعشان كدا استخدمت طريقة الـ errorstate

* لو أنا بستخدم طريقة الـ errorstate وبايز اعمل Return لحاجة من الفانكشن في الحالة دي بستخدم الـ Pointer في الـ Return

---

memset function:-

Task:- implement function memset:-

Void *my_memset(void* dest, int value, unsigned int bytelen)

Parameters:-

dest : Pointer to the destination object (start address)

value : value to be filled

bytelen : Number of bytes to be filled starting from dest to be filled.

* بستخدم memset لو أنا عايز اعمل fill لأراي متل بداتا معينة أو أعمل fill لجزء معين من الـ array بداتا معينة

memcpy function :-

* بستخدمها لو أنا عاوز اعمل copy بداية من start معين لحد مكان معين بعده بعمله copy في مكان تاني address أو address تاني

* على سبيل المثال لو عايز اعمل copy لـ array of char لـ array آخر

أو

Prototype :-

```
Void * my_memcpy (void *dest, const void *src,
                  unsigned int charCount);
```

Parameter :-

dest : Pointer to destination object.
src : Pointer to source object.
charCount: number of bytes to copy.

Task This lecture is implement All function of "string.h" Library

التاريخ: / /

الموضوع: _____

Pass 2D array as a Parameter to function:.

1- Passing 2D array to function Row and Column:-

```
Void Numbers_1( const int Numbers[size_10][size_20],
                const size_len_10 );
```

2- Passing 2D array to function omitting the Row:-

```
Void Numbers_2( const int Number[   ][size_20],
                const int size_len_10);
```

3- Passing 2D array to function using Pointers:-

```
unsigned char Numbers_3(const int (*Number)[size_20],
                const int size_len_10);
```

4- Passing 2D array to funetion using Pointers:-

```
unsigned char Numbers_4( const int (*Number)[size_10][size_20],
                const int size_len_10);
```

التاريخ: / / الموضوع: _____

How to find sizeof array in C?

1→ Size = size of (name of array) / sizeof (any element of array);

2- using Pointer Arithmetic

$$*(\&\text{name}_{\text{Array}} + 1) - \text{name}_{\text{Array}}$$

Ex:

```
int Numbers[5] = {1, 2, 3, 4, 5}
```

Address of the first element is:-

```
Numbers
&Numbers[0]
(Numbers+0)
```

```
&Numbers[0]+1
```
← هنا كدا بيشاور على تاني عنصر عشان الـ 1

```
&Numbers   //Pointer to array of 5 element

&Numbers+1 //Address of next memory block
             Address ahead of 5 integers
```
العنوان اللي بعد الـ 5 عناصر

```
int Arr[] = { 0x11, 0x22, 0x33 };

*(&Arr+1) - Arr   // Return size of Arr.
                  // number of elements.
```

Function Pointers concepts :-

- A Pointer to function in c is one of most important Pointer tools

What is a function Pointer or Pointer to function?

- Function Pointer is similar to the other Pointers but the only difference is that it stores the address of a function instead of a variable.

- In the Program whenever required we can invoke the Pointed function using the function Pointer.

- using the function Pointer we can Provide the run time binding in c Programing which resolves the many Problems.

:- Pointer to function

- هو عبارة عن بوينتر يخزن فيه عنوان الفانكشن .
- ممكن استعمل البوينتر في استدعاء الفانكشن بدل معمل استدعاء للفانكشن .

خطوات إنشاء Pointer to function ؟

١- باخد الـ Prototype بتاع الفانكشن الـ عاوز اعملها Pointer يشاور عليها بعملها copy

٢- بغير إسم الفانكشن لـ إسم PtrFun مثلاً ليدل على المعنى

٣- بحط إسم الفانكشن بين قوسين ( )

٤- بحط قبل إسم الفانكشن (*)

* لو حابب استعمل الـ Pointer ←

* هنا أنا خزنت عنوان الفانكشن في البوينتر Pointer = Name of function;

و استعمل البوينتر كأنه فانكشن Pointer ();

* لازم البوينتر يكون نفس الفانكشن اللي بيشاور عليها في الدخل و الخرج

* ممكن احط & قبل اسم الفانكشن وممكن اشيلها عادي مش هيأثر

* لازم احط اسم البوينتر بين ( ) لو شيتهم كدا اصبح فانكشن و بترجع بوينتر على حسب النوع الكائنة

* إسم الفانكشن هو عنوان الفانكشن

* لو عندي بوينتر ← بيشاور على فانكشن بتاخد Parameter ممكن مكتش أساسي لـ Parameter

Array of function Pointer:

* لو عايز اعمل array كل عنصر فيها عبارة عن Pointer كل Pointer يشاور على فانكشن أو بيخزن عنوان فانكشن

Ex:

```
Void (*Ptr[2])(Void) = {NULL, NULL}
```

* مملكن اشيل الـ NULL واحط اسامي الفانكشن.
* الفانكشن لازم تكون شبه بعض، بحيث الدخل متشابه والخرج متشابه

→ Ptr is an array of two element each element is a Pointer to function which No input and No output

هل ينفع أباصي function لـ function تانية as Parameter؟

← نعم ينفع باستخدام الـ Pointer to function وليكن مثلا داهيبقا شكل الـ function بتاع الـ Prototype.

```
Void Print_Summing(int Num1, int Num2,
                   Void (*Ptr_sum)(int ,int ));
```

Ex:- ولما أجي اعمل call للفانكشن دي هكتبها كدا:

```
Print_Summing(2, 3, Name of function);
```

→ اسم الفانكشن فقط

Binding concept:

* Associate the call function with the definiation of that function.

* There are 2 kinds of Binding: Static Binding.
  Dynamic Binding.

1-Static Binding/Compile time Binding/Early Binding

- All information necessary in order to perform that association is available at compile time.
- Association happening at compile time.
- Happens by default → Normal function call.
- Makes our program run faster.

2-Dynamic Binding/Run time Binding/Late Binding

- Association happening during run time.
- All information necessary in order to perform that is not available at compile time, it is availbe run time.
- Makes our program a little bit slower.
- Very flexible → Decide which function definition to invoke at run time.

التاريخ: / / الموضوع: _____

Application of function Pointers:

* used to resolve the runtime binding.

Call-back function:-

- we can implement the call-back function using function pointer in the c programming.

- A call back function is any executable code that is passed as an argument to other code that is expected to call back (execute) the argument at a given time.

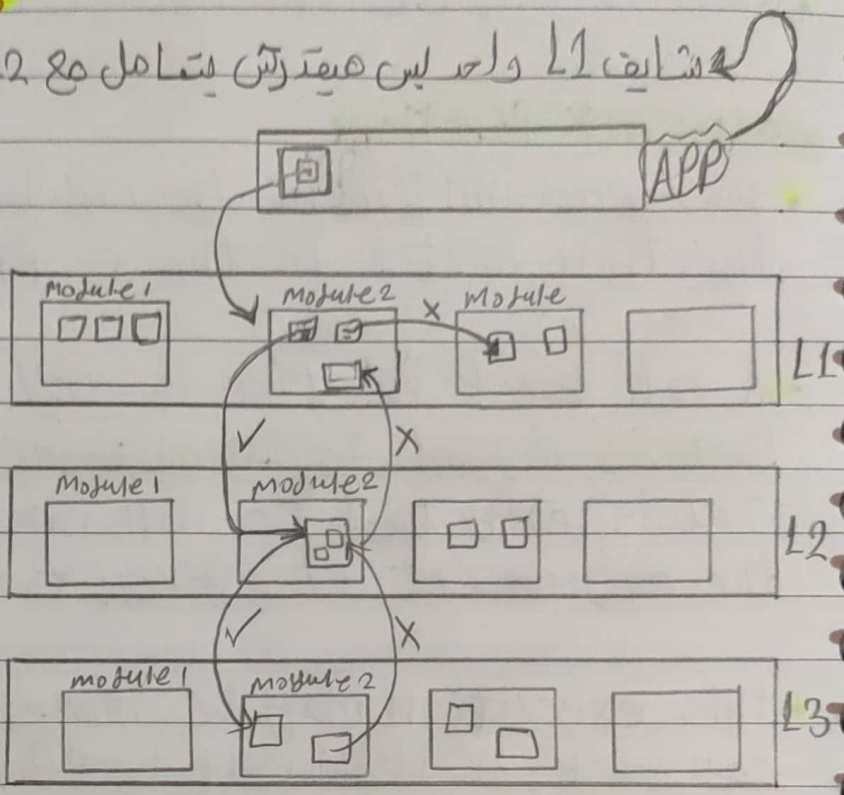
- this execution may be immediate as in synchronous call back, or it might happen at a later time as in a asynchronous call back

- if an API call is synchronous, it means that code execution will block (or wait) for API call to return before contining.
- This means that until a response is returned by the API, your application will not execute any function, which could be perceived by the user as latency or performace lag in your App

- Asynchronous calls do not block (or wait) for the API call to return from the server.
- Execution continues on in your program, and when the call returns from the server, a "call back" function is executed.

- Software Design APPRoach
Layers → Modules

ضايف L1 واحد بس هيقدرش يتعامل مع L3 & L2



* layer APPRoach software Design *

- عبارة عن طريقة معينة للناس اللي بتكتب الكود في الإيمبديد علشان لما أجي اعمل برنامج مثلا البرنامج دا بيتقسم لمجموعة من الـ layers كل layer جواها مجموعة من الـ Modules كل Module بيتكون أو جواه مجموعة من الفولدرز كل فولدر بيحتوي على الفايلات فايل .c فايل .h هي عملية تنظيمية للبرنامج

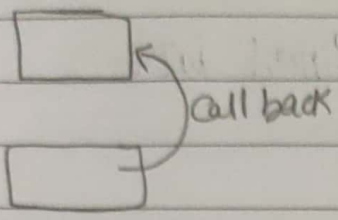
* في مبدأ من النوع دا من الـ Design بيقول إن ممنوع فانكشن تعمل call لفانكشن تانية في نفس الـ layer ولكن ينفع تعمل call من layer أقل منها مثلا زي فانكشن في layer1 تعمل call لـ module2 في layer2 وده برضه تعمل call لـ module2 في layer3

* لكن العكس ممنوع بطريقة مباشرة

التاريخ: / /

الموضوع: _____

* لو عايز فانكشن في layer2 مثلا تنادي على فانكشن في layer1 هينفعش بطريقة direct وكده هستعمل Call-back Function



* عشان استعمل call back مفيش حل غير إن استعمل Pointer to function

* الـ call-back بنشوف منها نوعين :-

Synchronous :-

لما أنا هعمل request من layer أقل مثلا هينفذ الـ functionality ولما تخلص هيطلب الـ call-back في اللحظة الأبليكشن هيعمل stak لحد هيخلص الـ function و الـ call back

asynchronous :-

موجودة في الـ Embedded APP عادة وهي إن الـ APP مش بيعمل stak وكده يروح ينفذ function تانية على عكس synchronous

* لازم أعمل check على Pointer to fun لو أنا مستعمل call back عشان ميحصلش crash للبرنامج