1. Data type của scanf là 32 bit, vậy có khi nào là 64 bit hay 16 bit không?

2. Anh Hải có ví dụ tham số có lúc lưu ở thanh ghi, lúc tham số lưu ở stack (lúc gọi hàm là dùng có 1 tham số chứ ko phải lớn hơn 4 tham số trở lên) thì có đúng ko?

3. kỹ năng đọc hiểu nhanh 1 data sheet của 1 con vi điều khiển, ví dụ MCP50, nên đọc theo thứ tự từ đầu hay đọc từ các thanh ghi

4. Tại sao anh Hải cho phép break trong while và for và có những giảng viên khác chỉ cho phép break trong switch case ?

6. vì sao x[++i,++j] lại được convert thành *(x+(++j))

7. nên viết là int* p hay int * p hay int *p và vì sao slide của anh Đạt lại viết là int * p mà không phải là int *p

8. tại sao cho nguyên hàm bool allocate10Bytes(uint8_t *outPtr); mà không cho luôn bool allocate10Bytes(uint8_t **outPtr); cho tối ưu bộ nhớ

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <stdint.h>

bool allocate10Bytes( uint8_t **outPtr )
{
   const size_t N = 10;
```

```
    *outPtr = (uint8_t*)malloc( N * sizeof( uint8_t ) );

    return *outPtr == NULL;
}

int main(void)
{
    uint8_t *p;

    bool success = allocate10Bytes( (uint8_t**)&p );

    if ( success )
    {
        printf("success");
        free( p );
    }
    else
    {
        printf("false");
    }
    return 0;
}
```

Thay vì:

Header.h

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
```

```
#include <stdint.h>

#ifndef HEADER_H_
#define HEADER_H_
/*************************************
*Prototype
*************************************/
bool allocate10Bytes(uint8_t *outPtr );

#endif/*HEADER_H*/
```

Allocate.c

```
#include "header.h"

/*Function to allocate 10 Bytes (memory) for a pointer*/
bool allocate10Bytes(uint8_t *outPtr )
{
   /*uint8_t *outPtr = (uint8_t*)&p*/
   const size_t N = 10;/*10 Bytes*/
   /*p2p is pointer to pointer to value of type uint8_t*/
/*-----------------------------------------------------------------------------------------*/
   /*The parameter (*outPtr) contains address of pointer p
   We want allocate dynamic memory for this address
   --> declare a pointer to pointer to dedicate that pointer is allocated dynamic memory */
   uint8_t **p2p = (uint8_t**)outPtr;/*cast type of pointer outPtr to pointer to pointer*/
   /*    After this, uint8_t **p2p = (uint8_t**)&p    */
/*-----------------------------------------------------------------------------------------*/
```

```
    /*&p is allocated by using malloc function, address of *p2p is address of p*/
    *p2p = (uint8_t*)malloc( N*sizeof( uint8_t ) );/*datatypes is 1 byte --> 10*1 byte = 10 bytes */
    /*memory has been allocated to the pointer then this function will return TRUE*/
    return *p2p != NULL;
}
```

Main.c

```
#include "header.h"


int main(void)
{
    /*declare pointer, this pointer just has a address but doesn't have memory,
    data type of this pointer (uint8_t*) must same data type of variable which this pointer point to.*/
    uint8_t *p = 0;/*NULL --> this pointer doesn't point anywhere*/
    /*argument is address --> casting address to (uint8_t) because the propotype is constant*/
    bool success = allocate10Bytes( (uint8_t*)&p );
    /*Must be successful, then free dynamic memory*/
    if ( success )
    {
        printf("allocate successful then free malloc\n");
        printf("address &p =%d\n",&p);
        printf("value p = %d\n", p);/*Trash value because this pointer didn't pointe to anywhere*/
        free( p );
    }
}
```

9. có cách nào khác để bảo toàn biến mà mình muốn retain lại khi optimize bằng tool mà ko cần dùng volatile không

10. quy tắc đặt tên cho 1 hàm API, ví dụ file tên là ManipulateFunction.c

1 hàm trong đó là hàm cho phép người dùng nhập một mảng có N phần tử thì nên đặt tên là Enter_Array hay EnterArray hay enterArray hay ManipulateFunctionEnter

con trỏ struct

11. Khi truy cập vào bộ nhớ không hợp lệ thì bị lỗi segment fault, làm thế nào mà có thể gây chết được chip hoặc chip bị systemdown (micro chip)

và PC đang chạy 1 chương trình như vậy có bị hỏng chip của PC không.

16. int *p; /*Lúc này mới khai báo 1 con trỏ và nó mới chỉ có địa chỉ chứ chưa có vùng nhớ

Cấp phát vùng nhớ cho con trỏ *p đó thì nên để ở stack hay là heap (mặc dù có vẻ heap được dùng nhiều hơn do người dùng tự do quản lý)*/

17. Sử dụng cấp phát động để cấp phát vùng nhớ lưu trữ giá trị được nhập từ người dùng, sử dụng con trỏ aptr để quản lý.

thường thì các bài tập là cấp phát vùng nhớ tĩnh (stack, data, bss) cho các giá trị được nhập từ người dùng,

và bài này yêu cầu cấp phát vùng nhớ động (heap) thì hiểu như vậy có đúng không?

con trỏ aptr trong bài này được cấp phát ở vùng nhớ động hay tĩnh đều được có đúng không?

18. Ngoài lỗi về ko quản lý bộ nhớ 1 cách chuẩn xác (ví dụ không giải phóng bộ nhớ)

thì còn những lỗi nào khiến cho hệ thống nhúng không bị lỗi trong quá trình test

mà phải sau 1 thời gian mới xảy ra lỗi lag (đơ, chậm) hoặc không hoạt động ổn định khi chạy lâu dài không?

19.tham chiếu (reference) và tham trị/ truyền kiểu tham chiếu và truyền kiểu tham trị