Bài tập 1: Viết hàm copy giá trị trong mảng này sang một mảng khác.

```
#include <stdio.h>
#define FOR(i, x, y) for(int i = x;i <=y; i++)
#define RE(i, x, y) for(int i = y; i>= x;i --)
#define maxN 100
int arr_coppy(int *arr, int *arr1, int arr_size);

int main()
{
    int n = 4;
    int a[maxN] = {1,2,2,3};
    int b[maxN] = {};
    arr_coppy(a, b, n);
    FOR(i, 0, n - 1)
    {
        printf("%d ", b[i]);
    }
    return 0;
}


/*
----------COPPY ARRAY---------------
Paramaters:
-   *arr1  : the source array
-   *arr2 : the destination array
-   arr_size : size of the array

Function:
-   Used for copying data from the source array to the destination array.
    It copies elements with indexes from 0 to arr_size - 1. The ouput is
0 if the function not error.
*/
int arr_coppy(int *arr1, int *arr2, int arr_size)
{
    FOR(i, 0, arr_size - 1)
    {
        arr2[i] = arr1[i];
    }
    return 0;
}
```

Bài tập 2: Viết hàm nhân các phần tử của 2 mảng theo từng vị trí tương ứng (mảng 1 chiều).

```
#include <stdio.h>
#define FOR(i, x, y) for(int i = x;i <=y; i++)
#define RE(i, x, y) for(int i = y; i>= x;i --)
#define maxN 100
int arr_multiply(int *arr1, int *arr2, int arr_size, int *res);

int main()
{
```

```
    int n = 3;
    int a[maxN] = {1, 2, 3}, b[maxN] = {1, 2, 3};
    int result[maxN] = {};
    arr_multiply(a, b, n, result);
    FOR(i , 0, n-1)
    {
        printf("%d ",result[i]);
    }
    return 0;
}

/*
 ----------MULTIPLY ARRAYS---------------
Parameters:
-   *arr1     : the first array
-   *arr2     : the second array
-   arr_size  : the size of the arrays
-   *res      : the result array

Function:
-    Multiplies each element of the first array with the corresponding
element
    in the second array (i.e., res[i] = arr1[i] * arr2[i]), from index 0
to arr_size - 1.
-   The resulting values are stored in *res. The ouput is 0 if the function
not error.
*/
int arr_multiply(int *arr1, int *arr2, int arr_size, int *res)
{
    FOR(i, 0, arr_size - 1)
    {
        res[i] = arr1[i] * arr2[i];
    }
    return 0;
}
```

Bài tập 3: Viết hàm nhân các phần tử của 2 mảng theo từng vị trí tương ứng (mảng 2 chiều).

```
#include <stdio.h>
#define FOR(i, x, y) for(int i = x;i <= y; i++)
#define RE(i, x, y) for(int i = y; i >= x;i --)
#define maxN 100
int arr_2d_mutiply(int arr1[][maxN], int arr2[][maxN], int arr_col, int
arr_row, int res[][maxN]);

int main()
{
    int row = 3, col = 3;
    int a[maxN][maxN] = {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
    }, b[maxN][maxN] = {
```

```
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
    } ;
    int result[maxN][maxN]={};
    arr_2d_mutiply( a,  b, col, row,  result);
    FOR(i, 0, row - 1)
    {
        FOR(j, 0, col - 1)
        {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }
    return 0;
}

/*
 ----------MULTIPLY 2D ARRAYS ---------------
Parameters:
-   *arr1     : the first array
-   *arr2     : the second array
-   arr_col  : the colums of the arrays
-   arr_row  : the rows of the arrays
-   *res      : the result array

Function:
-    Multiplies each element of the first array with the corresponding
element
    in the second array (i.e., res[i][j] = arr1[i][j] * arr2[i][j]). The
arr_col and arr_row indicate for the size include arr_col colums and
arr_row rows of res.
-   The resulting values are stored in *res.
*/
int arr_2d_mutiply(int arr1[][maxN], int arr2[][maxN], int arr_col, int
arr_row, int res[][maxN]){
    FOR(i, 0, arr_row - 1)
    {
        FOR(j, 0, arr_col - 1)
        {
            res[i][j] = arr1[i][j] * arr2[i][j];
        }
    }
    return 0;
}
```

Bài tập 4: Viết hàm tìm kiếm vị trí xuất hiện lần đầu tiên của một ký tự trong chuỗi.

```
#include <stdio.h>
#include <string.h>
#define FOR(i, x, y) for(int i = x;i <=y; i++)
#define RE(i, x, y) for(int i = y; i>= x;i --)
#define maxN 1000
int char_first(char *s, char character);
```

```
int main ()
{
    char c[maxN];
    fgets(c, sizeof(c), stdin);
    c[strlen(c)-1] = '\0';
    FOR(i, 0, strlen(c) - 1)
    {
        if(!char_first(c, c[i]))
        {
            printf("%c ", c[i]);
        }

    }
    return 0;
}
/*
----------CHAR FIRST---------------
Paramaters:
-   *s        : the string contain the character
-   character : the character that you need to check

Function:
-   The output is '1' if character appear 1 time on the string s
    and output is '0 if the character appear more than 1 time. The ouput
is 0 if the function  not error.
*/
int char_first(char *s, char character)
{
    int cnt = 0;
    FOR(i, 0, strlen(s) - 1)
    {
        if(s[i] == character)++ cnt;
        if(cnt == 2) return 1;
    }
    return 0;
}
```

Bài tập 5: Viết hàm nhận vào một chuỗi và 2 ký tự lưu trong tham số tên a và b, tìm kiếm và thay thế tất cả các vị trí xuất hiện ký tự a bằng ký tự b.

```
#include <stdio.h>
#include <string.h>
#define FOR(i, x, y) for(int i = x;i <=y; i++)
#define RE(i, x, y) for(int i = y; i>= x;i --)
#define maxN 1000

int char_find_a_re_b(char *s,char char1, char char2);

int main ()
{
    char c[maxN] , a, b;
    scanf("%c %c", &a, &b);
```

```
    getchar();
    fgets(c, sizeof(c), stdin);
    c[strlen(c) - 1] = '\0';
    char_find_a_re_b(c, a, b);
    printf("%s", c);
    return 0;
}
/*
----------FIND THE CHAR A AND REPLACE BY THE CHAR B---------------
Paramaters:
-   *s      : the string contain the character a
-   char a  : the character a
-   char b  : the character b

Function:
-    find the char a in the string s and replace it by the char b. The
ouput is 0 if the function not error.
*/
int char_find_a_re_b(char *s,char char1, char char2)
{
    FOR(i, 0, strlen(s))
    {
        if(s[i] == char1)
        {
            s[i] = char2;
        }
    }
    return 0;
}
```

Bài tập 6: Viết hàm đảo thứ tự các phần tử trong mảng.

```
#include <stdio.h>
#include <string.h>
#define FOR(i, x, y) for(int i = x;i <=y; i++)
#define RE(i, x, y) for(int i = y; i>= x;i --)
#define maxN 1000
int char_reverse(char *s);

int main()
{
    char c[maxN];
    fgets(c, sizeof(c), stdin);
    c[strlen(c) - 1] = '\0';
    char_reverse(c);
    printf("%s", c);
    return 0;
}
/*
----------FIND THE CHAR A AND REPLACE BY THE CHAR B---------------
Paramaters:
-   *s      : the string
Function:
-    The output is the revese of the string s . The ouput is 0 if the
function not error.
```

```
*/
int char_reverse(char *s)
{
    char temp;
    int left = 0;
    int right = strlen(s) - 1;
    while (left < right)
    {
        temp = s[left];
        s[left] = s[right];
        left ++;
        right --;
    }
    return 0;
}
```

Bài tập 7: Viết hàm so sánh 2 chuỗi để xác định xem 2 chuỗi có giống nhau hoàn toàn hay không.

```
#include <stdio.h>
#include <string.h>
#define FOR(i, x, y) for(int i = x;i <=y; i++)
#define RE(i, x, y) for(int i = y; i>= x;i --)
#define maxN 1000
bool str_cmp(char *string1, char* string2);
/*
----------COMPARING THE STRING---------------
Paramaters:
-   *string1   : the string 1
-   *string 2  : the string 2

Function:
-   The output is '1' if string 1 and string 2 are equal
    and output is '0 if string 1 and string 2 not equal
*/
bool str_cmp(char *string1, char* string2)
{
    int n = strlen(string1);
    int m = strlen(string2);
    if( n == m)
    {
        FOR(i, 0, n - 1)
        {
            if(string1[i] != string2[i])
            {
                return 1;
            }
        }
    }
    else
    {
        return 1;
    }
    return 0;
}
```

```
int main ()
{
    char c[maxN]="hello", p[maxN]="Hello";
    if(str_cmp(c, p))
    {
        printf("Error");
    }
    else
    {
        printf("Yup");
    }
    return 0;
}
```

Bài tập 8: Cho đoạn code bên dưới, điền vào vị trí [TODO] để hoàn thành hàm split có chức năng tìm kiếm vị trí xuất hiện đầu tiên của ký tự ' ' và cắt chuỗi thành 2 phần tại vị trí có kỹ tự ' '. Nếu ký tự ' ' nằm ở cuối hoặc khôn có trong chuỗi thì trả về NULL.

Lưu ý: Hàm split có thể thay đổi chuỗi đầu vào.

```
#include <stdio.h>
#include <string.h>
char *split(char *str)
{
 // [TODO]

    for(int i = 0; i < strlen(str); i++)
    {
        if(str[i] == ' ')
        {
            if( i == 0 || i == strlen(str) - 1)
            {
                return NULL;
            }
            else
            {
                str[i]='\0';
                return &str[i+1];
            }

        }

    }
    return NULL;
}
int main()
{
 char *res_str;

 // Case 1:
 char str_1[] = "Hello World";
 printf("\n=====\n");
 printf("Case 1: str_1 = %s\n", str_1);
 res_str = split(str_1);
```

```
 printf("str_1 = %s\n", str_1);
 if (res_str != NULL)
 printf("res_str = %s\n", res_str);
 else
 printf("No more string behind ' ' or string do not have ' '");
 // Case 2:
 char str_2[] = "Hello ";
 printf("\n=====\n");
 printf("Case 2: str_2 = %s\n", str_2);
 res_str = split(str_2);
 printf("str_2 = %s\n", str_2);
 if (res_str != NULL)
 printf("res_str = %s\n", res_str);
 else
 printf("No more string behind ' ' or string do not have ' '\n");
 // Case 2:
 char str_3[] = "Hello ";
 printf("\n=====\n");
 printf("Case 3: str_3 = %s\n", str_3);
 res_str = split(str_3);
 printf("str_3 = %s\n", str_3);
 if (res_str != NULL)
 printf("res_str = %s\n", res_str);
 else
 printf("No more string behind ' ' or string do not have ' '\n");
 return 0;
}
```

Bài tập 9: Cho đoạn code bên dưới, điền vào vị trí dấu ... để kết quả in ra terminal là:

Code:

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

char *str_list[3] =
{
   "This is 1st string",
   "This is 2nd string",
   "This is 3rd string",
};

int main()
{
   for (...)
   {
      printf(...);
   }

   return 0;
}
```

```
#include <stdio.h>
#include <string.h>
char *str_list[3] =
{
 "This is 1st string",
```

```
 "This is 2nd string",
 "This is 3rd string",
};
int main()
{
 for (int i = 0;i < 3; i++)
 {
    printf("%s\n", str_list[i]);
 }
 return 0;
}
```

Bài tập 10: Cho đoạn code bên dưới, điền vào vị trí [TODO] để hoàn thành hàm command_line_parse(). Cho biết hàm nhận vào một chuỗi ký tự, sau đó tách chuỗi ra thành các chuỗi nhỏ hơn tại vị trí xuất hiện ký tự ' '. Các chuỗi nhỏ sẽ được trả về thông qua tham số cli_command_args, tham số cli_command_num_args sẽ trả về số lượng các chuỗi nhỏ.

Gợi ý: Tìm hiểu hàm strtok() của thư viện <string.h>.

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

#define CLI_COMMAND_MAX_NUM_ARGS 10

/**
 * @brief Split a command string into tokens using ' '
 *
 * @param[in]  cli_command_buff   The input command string (modifiable)
 * @param[in]  cli_buff_size      Length of the input buffer
 * @param[out] cli_command_args   Array of char* to store each token (must
have enough space)
 * @param[out] cli_command_num_args Pointer to store number of tokens
 */
void command_line_parse(char *cli_command_buff, uint8_t cli_buff_size,
    char **cli_command_args, uint8_t *cli_command_num_args)
{
    char *token = strtok(cli_command_buff, " ");

     while(token != NULL){
        cli_command_args[*cli_command_num_args] = token;
        *cli_command_num_args = *cli_command_num_args + 1;
        token = strtok(NULL, " ");
    }

}

int main()
{
    char command_str[] = "SET GPIO A0 1";
    char *args_list[CLI_COMMAND_MAX_NUM_ARGS];  // mảng con trỏ
    uint8_t num_args = 0;
```

```
    command_line_parse(command_str,    sizeof(command_str),    args_list,
&num_args);

    for (int i = 0; i < num_args; i++) {
        printf("Argument %d: %s\n", i, args_list[i]);
    }

    return 0;
}
```