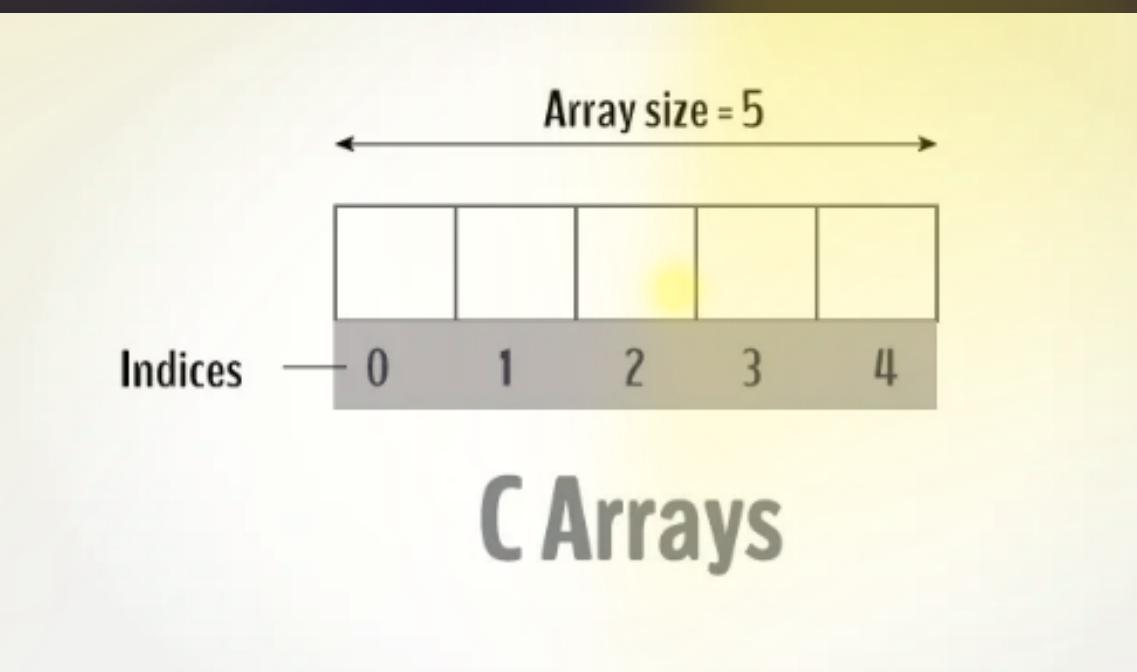


ARRAY AND FUNCTION



ARRAYS

ARRAYS



Array គឺជាតួន្យេរក្សាទីកែងបញ្ជីមុលដែលមានទំហំ
ដែលត្រូវបានបង្កើតឡើងដោយការផ្តល់ឈ្មោះទំនាក់ទំនង
នៃការបង្កើតឡើងនៅក្នុងបញ្ជីមុល។ ទំនាក់ទំនងនេះ
ត្រូវបានបង្កើតឡើងដោយការបង្កើតឡើងនៅក្នុងបញ្ជីមុល។

ARRAYS EXPLAIN



กล่องในที่ 0



กล่องในที่ 1



กล่องในที่ 2



กล่องในที่ 3



กล่องในที่ 4

ARRAYS EXPLAIN

10



กล่องในที่ 0

20



กล่องในที่ 1

30



กล่องในที่ 2

40



กล่องในที่ 3

50



กล่องในที่ 4

ARRAYS EXPLAIN

?



กล่องใบที่ 2

ARRAYS EXPLAIN

30



กล่องใบที่ 2

ARRAYS



```
1 int mark[5] = {19, 10, 8, 17, 9};
```

OUTPUT

mark[0] mark[1] mark[2] mark[3] mark[4]

--	--	--	--	--



mark[0] mark[1] mark[2] mark[3] mark[4]

19	10	8	17	9
----	----	---	----	---

```
#include <stdio.h>
int main(){
    int mark[5] = {19, 10, 8, 17, 9};
    mark[2] = -1;
    mark[4] = 0;

    printf("Mark[0] : %d\n", mark[0]);
    printf("Mark[1] : %d\n", mark[1]);
    printf("Mark[2] : %d\n", mark[2]);
    printf("Mark[3] : %d\n", mark[3]);
    printf("Mark[4] : %d\n", mark[4]);

    return 0;
}
```

CHANGE VARIABLES IN ARRAYS

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Mark[0] : 19
Mark[1] : 10
Mark[2] : -1
Mark[3] : 17
Mark[4] : 0
```

```
#include <stdio.h>
int main(){
    int values[5];

    printf("Enter 5 integers: ");

    for(int i = 0; i < 5; ++i){
        scanf("%d", &values[i]);
    }

    printf("Displaying integers: \n");

    for(int i = 0; i < 5; ++i){
        printf("%d\n", values[i]);
    }
    return 0;
}
```

EXAMPLES ARRAY WITH INPUT OUTPUT

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter 5 integers: 1 2 3 4 5
Displaying integers:
1
2
3
4
5
```

```
#include <stdio.h>
int main(){
    int marks[10], i, n, sum = 0;
    double average;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    for(i = 0; i < n; ++i){
        printf("Enter number%d:", i + 1);
        scanf("%d", &marks[i]);
        sum += marks[i];
    }

    average = (double)sum / n;

    printf("Average = %.2lf", average);

    return 0;
}
```

EXAMPLES

CALCULATE AVERAGE

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter number of elements: 5
Enter number1: 21
Enter number2: 22
Enter number3: 32
Enter number4: 12
Enter number5: 21
Average = 21.60
```

EXERCISE !!!

ឧប់ខែប្រជាមួយ

សង្ខេមនូវការបង្កើតកម្មវិធីដែលបានរាយការណ៍ឡើង។

OUTPUT

```
Enter a number to generate its multiplication table: 5
Multiplication table for 5:
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
5 x 11 = 55
5 x 12 = 60
```



```
#include <stdio.h>

#define MAX_SIZE 12

int main(){
    int num;
    int table[MAX_SIZE];

    printf("Enter a number to generate its multiplication table: ");
    scanf("%d", &num);

    printf("Multiplication table for %d:\n", num);

    for(int i = 0; i < MAX_SIZE; i++){
        table[i] = num * (i + 1);
    }

    for(int i = 0; i < MAX_SIZE; i++){
        printf("%d x %d = %d\n", num, i + 1, table[i]);
    }

    return 0;
}
```

FUNCTION

FUNCTION

How function works in C programming?

```
#include <stdio.h>

void functionName()
{
    ... ...
}

int main()
{
    ... ...
    functionName();
    ... ...
}
```

ฟังก์ชันคือ การแยกส่วนการทำงานของ Code ให้ทำงานอุ่นมาเป็นส่วนๆ เพื่อองค์การเขียนโปรแกรมให้อุ่นมาได้ គรุที่จะแยกการทำงานออกจากฟังก์ชัน หลักเพื่อให้ง่ายต่อการทำความเข้าใจและสามารถกลับมาแก้ไขการทำงานของแต่ละฟังก์ชันได้ง่ายมากยิ่งขึ้น

```
#include <stdio.h>

void sayHello()
{
    printf("Hello World");
}

int main()
{
    sayHello();
    return 0;
}
```

FUNCTION

OUTPUT

- PS C:\Users\olarn\C-Programming\c-programming-array-function>
- Hello World!

WHY
FUNCTION

```
● ● ●  
1 #include <stdio.h>  
2 main()  
3 {  
4     int a = 10;  
5     int b = 13;  
6     int c = 8;  
7  
8     if (a % 2 == 0)  
9     {  
10        printf("\n a is Even");  
11    }  
12    else  
13    {  
14        printf("\n a is Odd");  
15    }  
16  
17    if (b % 2 == 0)  
18    {  
19        printf("\n b is Even");  
20    }  
21    else  
22    {  
23        printf("\n b is Odd");  
24    }  
25  
26    if (c % 2 == 0)  
27    {  
28        printf("\n c is Even");  
29    }  
30    else  
31    {  
32        printf("\n c is Odd");  
33    }  
34 }
```

VS

```
● ● ●  
1 #include <stdio.h>  
2  
3 void checkEvenOdd(char name, int value)  
4 {  
5     if (value % 2 == 0)  
6     {  
7         printf("\n %c is Even", name);  
8     }  
9     else  
10    {  
11        printf("\n %c is Odd", name);  
12    }  
13 }  
14  
15 int main()  
16 {  
17     int a = 10;  
18     int b = 13;  
19     int c = 8;  
20  
21     checkEvenOdd('a', a);  
22     checkEvenOdd('b', b);  
23     checkEvenOdd('c', c);  
24 }
```

Code ที่ไม่ได้แยก Function

Code ที่แยก Function

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
```

```
○  
a is Even  
b is Odd  
c is Even
```

EXERCISE !!!

ຂໍ້ມູນ

จงเขียน Function ที่รับ Input เข้ามาแล้วนำไป Input ที่รับเข้ามาไปคูณ กับ 5 (รับ Input ใน Function)

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function> Enter you number : 5  
Result : 25  
PS C:\Users\olarn\C-Programming\c-programming-array-function> Enter you number : 21  
Result : 105
```



```
#include <stdio.h>

int multiplyFive()
{
    int number, result;
    printf("Enter your name : ");
    scanf("%d", &number);
    result = number * 5;
    printf("Result : %d", result);
    return 0;
}

int main()
{
    multiplyFive();
    return 0;
}
```

**DEFINED
FUNCTION**

```
#include <stdio.h>

int addNumbers(int a, int b); ← DEFINED FUNCTION

int main()
{
    int n1, n2, sum;
    printf("Enter two numbers : ");
    scanf("%d %d", &n1, &n2);

    sum = addNumbers(n1, n2);
    printf("sum : %d", sum);

    return 0;
}

int addNumber(int a, int b)
{
    int result;
    result = a + b;
    return result;
}
```

การประกาศฟังก์ชันเป็นการประกาศชื่อและประเภทของฟังก์ชันนั้นๆ เพื่อให้เราสามารถเขียน Function ในบรรทัดต่อไปของฟังก์ชัน main ได้ เมื่อจากหากเราเขียน Function ปกติเราจะต้องทำการเขียน Function ของเราริบ้หนึ่อ Function main Function ของเราจึงจะสามารถใช้งานได้หากในกรณีที่เราต้องการที่จะเขียนริบ้ให้ Function main ก็ให้เราทำการประกาศชื่อฟังก์ชันของเราในหนึ่อ Function main และเราจะสามารถเขียน Function ของเราให้ Function main ได้ เพื่อให้ง่ายต่อการใช้งานและกลับมาทบทวนในภายหลัง

```
#include <stdio.h>

int addNumbers(int a, int b);

int main()
{
    int n1, n2, sum;
    printf("Enter two numbers : ");
    scanf("%d %d", &n1, &n2);

    sum = addNumbers(n1, n2);
    printf("sum : %d", sum);

    return 0;
}

int addNumber(int a, int b)
{
    int result;
    result = a + b;
    return result;
}
```

DEFINED FUNCTION

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter two numbers : 1 1
○ sum : 2
```

EXERCISE !!!

วิธีแบบดูเฉลยหน้า

จะเขียน Function ที่รับ Input เข้ามาแล้วทำการตรวจสอบว่า Input นี้เป็นเลขคู่หรือเลขคี่ ถ้าเป็นเลขคู่ให้แสดงคำว่า "Even" และหากเป็นเลขคี่ให้แสดงคำว่า "Odd" โดยใช้หลักการของ Defined Function(รับ Input ใน Function)

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter your number : 21
21 is Odd
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter your number : 22
22 is Even
```



```
#include <stdio.h>

int checkEvenOddNumbers();

int main()
{
    checkEvenOddNumbers();
    return 0;
}

int checkEvenOddNumbers()
{
    int number;

    printf("Enter your number : ");
    scanf("%d", &number);
    if(number % 2 == 0)
    {
        printf("%d is Even", number);
    }
    else
    {
        printf("%d is Odd", number);
    }
}
```

ARGUMENTS

How to pass arguments to a function?

```
#include <stdio.h>

int addNumbers(int a, int b);

int main()
{
    ...
    ...
    sum = addNumbers(n1, n2);
    ...
}

int addNumbers(int a, int b)
{
    ...
    ...
}
```

ARGUMENTS

ค่าที่รับเข้ามาภายใน Function นั้นๆเพื่อนำค่าเหล่านั้นไปใช้งานภายใน Function ที่เราได้ทำการเขียนขึ้น อาจจะรับมาในรูปแบบของตัวเลข (int) หรือตัวอักษร (char) หรืออื่นๆ ก็ได้

```
#include <stdio.h>

int checkPrimeNumber(int n);

int main()
{
    int n, flag;

    printf("Enter a positive integer: ");
    scanf("%d", &n);

    flag = checkPrimeNumber(n);

    if(flag == 1)
        printf("%d is not a prime number", n);
    else
        printf("%d is a prime number", n);

    return 0;
}

int checkPrimeNumber(int n)
{
    if (n == 0 || n == 1)
        return 1;

    for(int i = 2; i <= n / 2; ++i)
    {
        if(n % i == 0)
            return 1;
    }

    return 0;
}
```

ARGUMENTS

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter you number : 21
21 is not a prime number
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter a positive integer: 57
57 is not a prime number
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter a positive integer: 19
19 is a prime number
```

**TYPES OF
DEFINED
FUNCTION**

void function

```
●●●  
1 #include <stdio.h>  
2  
3 void sayCE()  
4 {  
5     printf("Welcome to Computer Engineering!!");  
6 }  
7  
8 int main()  
9 {  
10    sayCE();  
11    return 0;  
12 }
```

int function

```
●●●  
1 #include <stdio.h>  
2  
3 int sayCE()  
4 {  
5     printf("Welcome to Computer Engineering!!");  
6     return 0;  
7 }  
8  
9 int main()  
10 {  
11    sayCE();  
12    return 0;  
13 }
```

TYPES OF DEFINED FUNCTION

ในการเขียน Function ในภาษา C ที่เราจะได้ใช้กันบ่อยๆ คือ Void กับ Int ซึ่งน้องๆ หลายคนอาจจะสงสัยว่าก็ Function กั้ง 2 ชนิดนี้ต่างกันอย่างไร ? void กับ int จะต่างกันตรงที่ function ที่มีชนิดเป็น void จะไม่จำเป็นจะต้อง return ค่าภายใน function และ function int จะจำเป็นที่ต้อง return ค่าภายใน function นั้นๆ

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>  
Welcome to Computer Engineering!!
```

ฟังก์ชันที่ไม่ได้มีการ return ค่ากลับ

```
● ● ●  
1 #include <stdio.h>  
2  
3 void checkPrimeNumber();  
4  
5 int main() {  
6     checkPrimeNumber();  
7     return 0;  
8 }  
9  
10 void checkPrimeNumber() {  
11     int n, i, flag = 0;  
12  
13     printf("Enter a positive integer: ");  
14     scanf("%d",&n);  
15  
16     if (n == 0 || n == 1)  
17         flag = 1;  
18  
19     for(i = 2; i <= n/2; ++i) {  
20         if(n%i == 0) {  
21             flag = 1;  
22             break;  
23         }  
24     }  
25  
26     if (flag == 1)  
27         printf("%d is not a prime number.", n);  
28     else  
29         printf("%d is a prime number.", n);  
30 }  
31
```

TYPES OF DEFINED FUNCTION

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>  
Enter a positive integer: 21  
21 is not a prime number.
```

ฟังก์ชันที่มีการ return ค่ากลับ

```
● ● ●
1 #include <stdio.h>
2 int getInteger();
3
4 int main()
5 {
6     int n, i, flag = 0;
7     n = getInteger();
8
9     if (n == 0 || n == 1)
10        flag = 1;
11
12    for (i = 2; i <= n / 2; ++i)
13    {
14        if (n % i == 0)
15        {
16            flag = 1;
17            break;
18        }
19    }
20
21    if (flag == 1)
22        printf("%d is not a prime number.", n);
23    else
24        printf("%d is a prime number.", n);
25
26    return 0;
27 }
28
29 int getInteger()
30 {
31     int n;
32
33     printf("Enter a positive integer: ");
34     scanf("%d", &n);
35
36     return n;
37 }
38
```

TYPES OF DEFINED FUNCTION

OUTPUT

```
PS C:\Users\olarn\C-Programming\c-programming-array-function>
Enter a positive integer: 21
21 is not a prime number.
```

RECURSIVE FUNCTION

```
void recurse() {  
    ... ... ...  
    recurse(); ————— recursive call  
    ... ... ...  
}  
  
int main()  
{  
    ... ... ...  
    recurse(); —————  
    ... ... ...  
}
```

ฟังก์ชันเวียนบังเกิด (Recursive Function)
ฟังก์ชันที่เรียกตัวเอง หลักการฟังก์ชันเวียนบังเกิดคือ การเขียนโปรแกรมวนซ้ำเพื่อลดปัญหาของโปรแกรมที่ซับซ้อน โดยรวมแล้วคือ การเขียนฟังก์ชันมาตั้วหนึ่ง ถ้ายังหาคำตอบไม่ได้ก็ให้เรียกตัวเองซ้ำไปเรื่อยๆ จนกว่าจะเจอคำตอบ

```
● ● ●  
1 #include <stdio.h>  
2 int sum(int n);  
3  
4 int main()  
5 {  
6     int number, result;  
7  
8     printf("Enter a positive integer: ");  
9     scanf("%d", &number);  
10  
11    result = sum(number);  
12  
13    printf("sum = %d", result);  
14    return 0;  
15 }  
16  
17 int sum(int n)  
18 {  
19     if (n != 0)  
20         return n + sum(n - 1);  
21     else  
22         return n;  
23 }
```

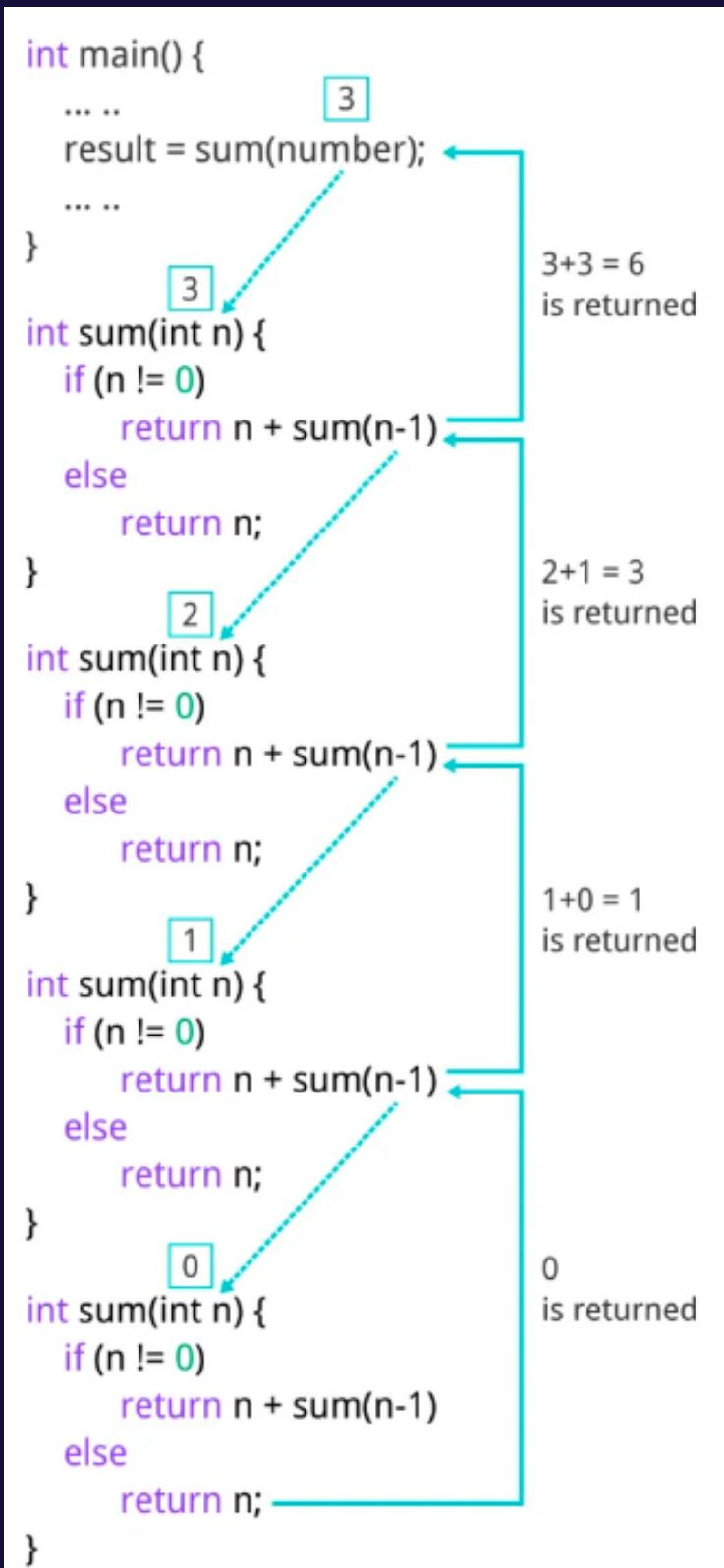
RECURSIVE FUNCTION

OUTPUT

- ▶ PS C:\Users\olarm\C-Programming\c-programming-array-function>
- ▶ Enter a positive integer: 3
- ▶ sum = 6

RECURSIVE FUNCTION EXPLAINED

โดยที่ Function ดังกล่าวคือ Recursive Function
ที่ทำการส่งค่า Argument จากฟังก์ชัน main ไปยัง
function sum ภายใน function sum ก็จะทำการตรวจสอบ
ส่วนว่า argument ที่เราเข้ามามีค่าเท่ากับ 0 หรือไม่หาก
ยังไม่เท่ากับ 0 ก็จะทำการนำค่านั้นไปบวกกับค่าของ
function sum ที่รับ argument เดิม ลดตัวย 1 และก็จะ
ทำงานแบบนั้นซ้ำไปเรื่อยๆจนกว่าค่าที่ส่งเข้าไปยัง
function sum จะมีค่าเท่ากับ 0 Recursive function ก็
จะหยุดทำงาน



THANK YOU



borntoDev