Programming Fundamental
Basic C Programming

Array

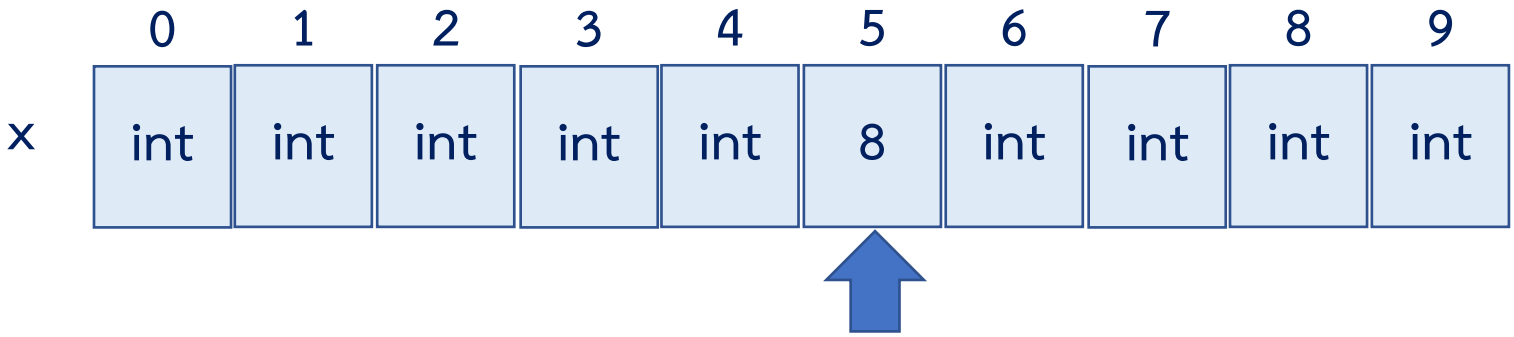
Function

Array = ชุดข้อมูลชนิดเดียวกัน ที่เรียงกันแบบมีลำดับ
ข้อมูลแต่ละตัวจะมีเลขอ้างอิงตำแหน่งของตนเอง

ผู้ใช้งานสามารถใช้ข้อมูลใน Array ได้โดยอ้างอิงลำดับในการเก็บข้อมูล

```
type     var_name[size];
```

```
int x[10];
```



x[5] = 8

เมื่อกำหนดให้ x เป็น Array

ตัวแปร x จะเก็บข้อมูล address ของข้อมูลตัวแรก

แต่เมื่อใช้วงเล็บพร้อมกับเลขตำแหน่ง x[?]

จะได้ข้อมูล integer ที่เก็บภายใน array x

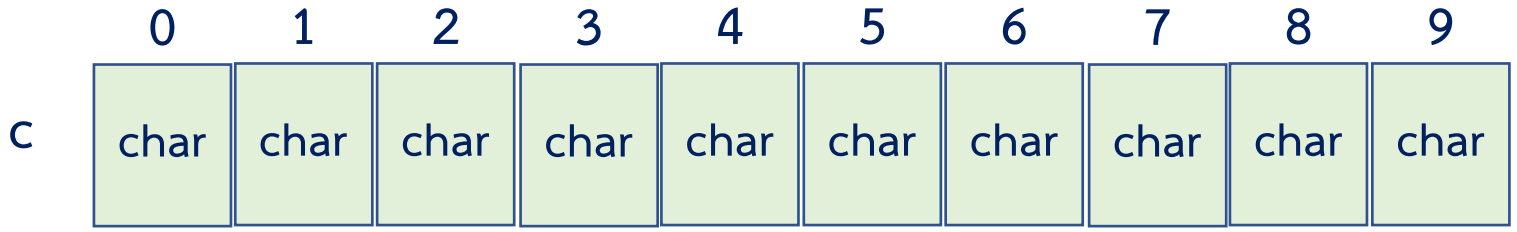
#Activity

ผลลัพธ์ของโปรแกรมคืออะไร

```
int main()
{
int p,x[5] = { 10,9,8,7,6 };
for (p = 0;p <= 4;p++)
        printf("%d\n", x[p]);
system("pause");
return 0;
}
```

String = array of characters

char c[10];



#Activity

ผลลัพธ์ของโปรแกรมคืออะไร

```
int main()
{
char c[20];
scanf("%s", &c);
printf("%s\n", c);
system("pause");
return 0;
}
```

#include<string.h> function

function	การทำงาน
strlen(str)	ส่งค่าเป็นความยาวของ str
strcat(dst,src)	นำ src ไปต่อท้าย dst
strcpy(dst,src)	นำข้อมูล src ไปเก็บไว้ใน dst

#Activity

ให้นักศึกษาเขียนโปรแกรมรับ string 1 ชุด
แล้วแสดงผลเฉพาะตัวอักษรตัวพิมพ์ใหญ่ใน string ดังกล่าว เช่น

Input : abCDEfGHiJk

Output : CDEGHJ

#include<???>

Stdio.h , conio.h , string.h , math.h , ctype.h

Function

Standard Library function

User Defined function

Compiler ไม่มีให้ ... Programmer ต้องเขียนเอง

ตัวอย่างการใช้งานฟังก์ชันการคำนวณทางคณิตศาสตร์

```
#include<stdio.h>
#include<math.h>
#define PI 3.14
int main()
{
	float deg,rad;
	printf ("Enter Degree : ");
	scanf ("%f",&deg);
	rad = deg * PI / 180;
	printf ("sin(%.2f) = %.3f\n",deg,sin(rad));
	printf ("cos(%.2f) = %.3f\n",deg,cos(rad));
	printf ("tan(%.2f) = %.3f\n",deg,tan(rad));
	return 0;
}
```

ฟังก์ชันที่สร้างขึ้นเอง (User-defined Function)

- เนื่องจาก Standard Library Function ทั้งหมด เป็นฟังก์ชันมาตรฐานที่มีเฉพาะการทำงานพื้นฐานต่างๆ เท่านั้น

- หากต้องการฟังก์ชันที่มีการทำงานเฉพาะกิจ โปรแกรมเมอร์ต้องเขียนฟังก์ชันขึ้นมาเอง

ข้อกำหนดพื้นฐานของ User-defined Function

1. มีการประกาศ function prototype ที่ต้นโปรแกรมเสมอ จึงจะเรียกใช้งาน function นั้นๆ ได้ (เป็นการบอก Compiler ว่าคำสั่งดังกล่าวคือฟังก์ชัน ไม่ใช่ syntax error)

2. ต้องมีการเขียนฟังก์ชันตามโครงสร้างที่ได้ประกาศไว้ใน function prototype เท่านั้น

โครงสร้างโปรแกรมภาษาซี

```
#include<file.h>
type function_name(type);
type variable
int main()
{
     type variable;
     statement-1;
     ...
     statement-n;
     return 0;
}
type function_name(type variable)
{
     statement-1;
     ...
     statement-n;
     return(var);
}
```

พรีโปรเซสเซอร์ไดเร็คทีฟ
ฟังก์ชันโพรโทรไทพ์
ตัวแปรชนิดโกบอล
ตัวแปรชนิดโลคอล
คำสั่ง
ฟังก์ชันหลัก
ฟังก์ชันย่อย
ส่วนหัวโปรแกรม
ส่วนตัวโปรแกรม

โปรโตไทป์ฟังก์ชัน (Function Prototype)

เป็นการประกาศการใช้งานฟังก์ชั่นที่อยู่หลัง main()

```
type function_name (type-1,type-2,...,type-n);
```

type	คือ ชนิดของฟังก์ชัน ว่าฟังชันที่ทำการสร้างจะส่งข้อมูลชนิดใดกลับ
function_name	คือ ชื่อฟังก์ชันที่จะสร้างขึ้น
type-n	คือ ชนิดของข้อมูลที่จะส่งให้ฟังก์ชัน

Function Prototype

```
int max (int,int);
float min (float,float);
int isPrime (int);
```

Function Implementation

```
int max (int a,int b)
float min (float x,float y)
int isPrime (int n)
```

ฟังก์ชันที่สร้างขึ้นเอง (User-defined Function)

การเขียนโปรแกรมโดยมีฟังก์ชันที่สร้างขึ้นเองมี 2 รูปแบบ

- สร้างฟังก์ชัน ก่อน ฟังก์ชันหลัก

  ฟังก์ชันหลักสามารถเรียกใช้งานฟังก์ชันที่สร้างขึ้นได้

- สร้างฟังก์ชัน หลัง ฟังชันหลัก

  ต้องประกาศ Function Prototype ก่อนเพื่อให้ฟังก์ชันหลักรู้ว่ามีฟังก์ชันที่สร้างขึ้น

```
#include<file.h>
type variable
type function_name(type variable)
{
    statement-1;
    ...
    statement-n;
    return(var);
}
int main()
{
    type variable;
    statement-1;
    ...
    statement-n;
    return 0;
}
```

```
#include<file.h>
type function_name(type);
type variable
int main()
{
    type variable;
    statement-1;
    ...
    statement-n;
    return 0;
}
type function_name(type variable)
{
    statement-1;
    ...
    statement-n;
    return(var);
}
```

ประเภทของฟังก์ชันที่สร้างขึ้นเอง

- ฟังก์ชันที่ไม่มีการรับส่งค่า
- ฟังก์ชันที่มีการรับค่า แต่ไม่ส่งค่ากลับ
- ฟังก์ชันที่มีการรับค่า และมีการส่งค่ากลับ
- ฟังก์ชันที่ไม่มีการรับค่า แต่มีการส่งค่ากลับ

** สามารถแยกประเภทได้โดยดูจาก function prototype

ตัวอย่าง ฟังก์ชันที่ไม่มีการรับส่งค่า

```
#include<stdio.h>
void    PrintHello(void);
int main()
{
        PrintHello();
        printf("Hello,in function main.\n");
        PrintHello();
        return 0;
}
void    PrintHello(void)
{
   printf("Hello,in function PrintHello ..\n\n");
}
```

ตัวอย่างฟังก์ชันที่มีการรับค่า แต่ไม่ส่งค่ากลับ

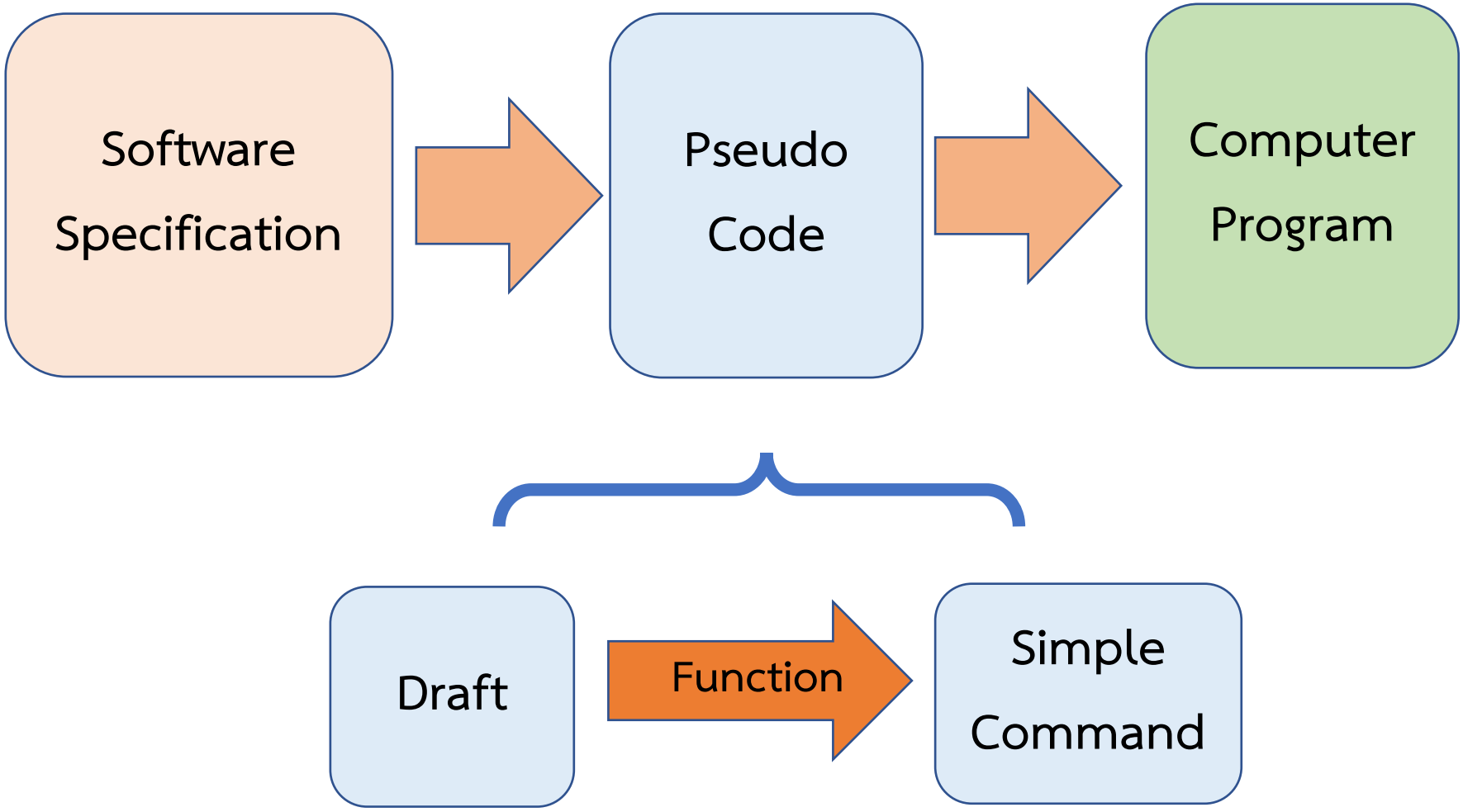
```
#include<stdio.h>
void PrintHello( int );
int main()
{       int n;
        printf("input number of hello : ");
        scanf("%d",&n);
        PrintHello( n );
        return 0;
}
void PrintHello( int i )
{ int count;
  for ( count=1; count<=i ; count++ )
        printf("%d HELLO\n",count);
}
```

ตัวอย่าง ฟังชั่นที่มีรับค่า มีส่งค่า

```
#include<stdio.h>
float  CircleArea( int );
int  main()
{ int  radius;
   printf("input radius : ");     scanf("%d",&radius);
   printf(" Circle area = %f\n",CircleArea(radius));
   return 0;
}
float  CircleArea( int rad )
{ float  answer=0;
   answer  =  22.0/7.0*rad*rad ;
   return  answer;
}
```

ตัวอย่างฟังก์ชันที่ไม่มีการรับค่า แต่มีการส่งค่ากลับ

```
#include<stdio.h>
int  RoundInput( void );
int  main()
{  int  i,round;
   round = RoundInput();
   for( i=1; i<=round;i++)  printf("hello #%d\n",i);
   return 0;
}
int  RoundInput(void)
{  int answer;
   printf( "Please input number of hello : ");
   scanf( "%d", &answer );
   return  answer;
}
```


Software Specification
Pseudo Code
Computer Program
Draft
Function
Simple Command

เขียนการทำงานเพื่อตรวจสอบว่าตัวเลขที่ป้อนเป็น
ตัวเลขจำนวนเฉพาะหรือไม่
ถ้าเป็นจำนวนเฉพาะ แสดงผล “Prime Number”
ถ้าไม่ใช่จำนวนเฉพาะ แสดงผล “Not Prime Number”

Draft Pseudo-Code

รับตัวเลขจำนวนเต็ม เก็บใน x
ถ้า x เป็นจำนวนเฉพาะ ให้แสดงผล “Prime Number”
ถ้าไม่ใช่ให้แสดงผล “Not Prime Number”

รับตัวเลขจำนวนเต็ม เก็บใน x
ถ้า x เป็นจำนวนเฉพาะ ให้แสดงผล “Prime Number”
ถ้าไม่ใช่ให้แสดงผล “Not Prime Number”

สร้าง function isPrime(int) ที่เมื่อ int เป็นเลขจำนวนเฉพาะ
isPrime() จะ return 1 ถ้าไม่ใช่จะ return 0

รับตัวเลขจำนวนเต็ม เก็บใน x
ถ้า isPrime(x) == 1 ให้แสดงผล “Prime Number”
ถ้าไม่ใช่ให้แสดงผล “Not Prime Number”

Activity

ให้นักศึกษาเขียนโปรแกรมเพื่อตรวจสอบว่าตัวเลขที่ป้อนเป็นตัวเลขจำนวนเฉพาะหรือไม่
ถ้าเป็นจำนวนเฉพาะ แสดงผล “Prime Number”
ถ้าไม่ใช่จำนวนเฉพาะ แสดงผล “Not Prime Number”
โดยสร้าง และใช้งานฟังก์ชั่น isPrime()