

# Introduction to C++

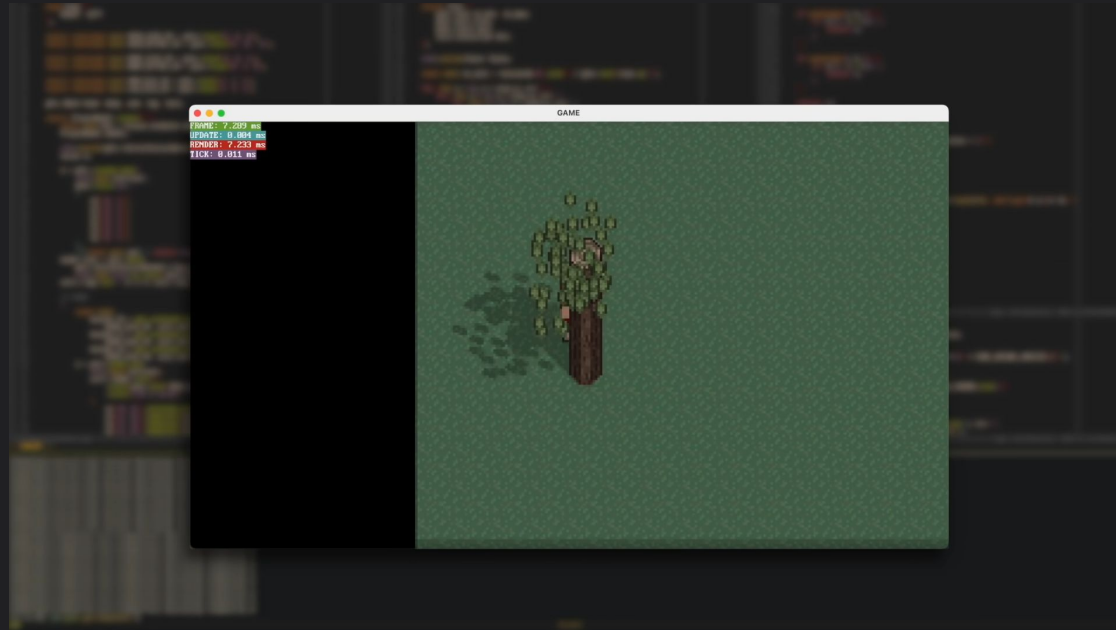
By Rufflogix 

# Outline

- What is C++?
- Why should we learn C++?
- C++ Template
- Input / Output
- Conditional Statement
- Loops
- Functions
- Pointer
- Array
- String
- Structure & Union

# Why should we learn C++?

ภาษา C++ นั้นเป็นภาษาที่เร็วและมีประสิทธิภาพสูง เราจึงมักใช้ภาษา C++ ในการทำงานต่างๆ เช่น การพัฒนาเกม การพัฒนาซอฟต์แวร์ต่างๆ หรือกระทั่งในการเขียนโปรแกรมเชิงแข่งขัน



# What is C++?

C++ ถูกพัฒนาโดย Bjarne Stroustrup ใน Bell Laboratories ในช่วงต้นของปี 1979 ภาษา C++ มีการเพิ่ม object-oriented features และความสามารถอื่นๆ เพิ่มเติมจากภาษา C โดยในช่วงแรกภาษา C++ ถูกเรียกว่า "C with Objects" ถัดมาในปี 1983 Stroustrup ได้ตั้งชื่อใหม่ว่า C++ โดยชื่อ C++ มีความหมายว่า "C incremented"



# C++ Template : Version 1.0



C++ Template version 1.0

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     // Code here
7
8     return 0;
9 }
```

นำเข้า library พื้นฐาน

# C++ Template : Version 2.0



C++ Template version 2.0

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7     // Code here
8
9     return 0;
10 }
```

เพิ่มความเร็วคำสั่ง input / output

# C++ Template : Version 3.0



C++ Template version 3.0

```
1 #include <bits/stdc++.h>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7     // Code here
8
9     return 0;
10 }
```

นำเข้า library แบบครอบจักรวาล

# Input / Output

คำสั่ง `cout` นั้นเป็นคำสั่งที่ใช้แสดงค่าผ่านทางหน้าจอ เช่น `cout << "Hello World";`



Input / Output

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     cout << "Hello World";
9
10    return 0;
11 }
```



# Input / Output : Variables

ประเภทข้อมูลที่ใช้หลักๆ ในภาษา C++ ได้แก่

- Integer [ `int` => 3 | 102 ]
- floating number [ `float` => 1.2 | 24.0 ]
- boolean [ `bool` => true | false ]
- character [ `char` => 'a' | 'b' ]
- string [ `string` => "Hi" | "12" ]



Input / Output

```
1 int main() {  
2     string username = "RuffLogix";  
3     cout << username;  
4  
5     return 0;  
6 }
```

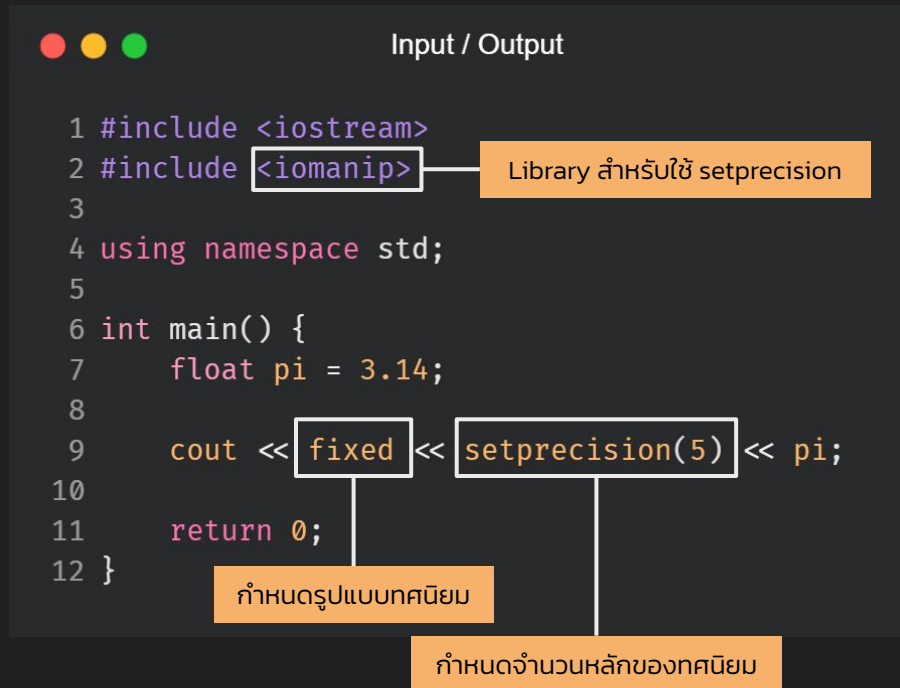


Input / Output

```
1 #include <iostream>  
2  
3 using namespace std;  
4  
5 int main() {  
6     ios_base::sync_with_stdio(0), cin.tie(0);  
7  
8     int a = 10;  
9     float b = 20;  
10    bool c = true;  
11    char d = 'x';  
12    string e = "Hello World\n";  
13  
14    return 0;  
15 }
```

# Input / Output

เราสามารถแสดงค่าทศนิยมโดยการกำหนดจำนวนหลักของเลขทศนิยมได้  
ด้วยคำสั่ง `fixed` และ `setprecision()`



```
1 #include <iostream>
2 #include <iomanip>
3
4 using namespace std;
5
6 int main() {
7     float pi = 3.14;
8
9     cout << fixed << setprecision(5) << pi;
10
11     return 0;
12 }
```

Input / Output

Library สำหรับใช้ setprecision

กำหนดรูปแบบทศนิยม

กำหนดจำนวนหลักของทศนิยม

# Input / Output : Ascii Table

ASCII Table ย่อมาจาก American Standard Code for Information Interchange เป็น ตัวเลขที่ใช้แทน character เช่น 97 แทน 'a' หรือ 65 แทน 'A' เป็นต้น

Dec	Oct	Hex	C	Dec	Oct	Hex	C	Dec	Oct	Hex	C	Dec	Oct	Hex	C
0	0	0	^@	32	40	20		64	100	40	@	96	140	60	`
1	1	1	^A	33	41	21	!	65	101	41	A	97	141	61	a
2	2	2	^B	34	42	22	"	66	102	42	B	98	142	62	b
3	3	3	^C	35	43	23	#	67	103	43	C	99	143	63	c
4	4	4	^D	36	44	24	\$	68	104	44	D	100	144	64	d
5	5	5	^E	37	45	25	%	69	105	45	E	101	145	65	e
6	6	6	^F	38	46	26	&	70	106	46	F	102	146	66	f
7	7	7	^G	39	47	27	'	71	107	47	G	103	147	67	g
8	10	8	^H	40	50	28	(	72	110	48	H	104	150	68	h
9	11	9	^I	41	51	29	)	73	111	49	I	105	151	69	i
10	12	a	^J	42	52	2a	*	74	112	4a	J	106	152	6a	j
11	13	b	^K	43	53	2b	+	75	113	4b	K	107	153	6b	k
12	14	c	^L	44	54	2c	,	76	114	4c	L	108	154	6c	l
13	15	d	^M	45	55	2d	-	77	115	4d	M	109	155	6d	m
14	16	e	^N	46	56	2e	.	78	116	4e	N	110	156	6e	n
15	17	f	^O	47	57	2f	/	79	117	4f	O	111	157	6f	o
16	20	10	^P	48	60	30	0	80	120	50	P	112	160	70	p
17	21	11	^Q	49	61	31	1	81	121	51	Q	113	161	71	q
18	22	12	^R	50	62	32	2	82	122	52	R	114	162	72	r
19	23	13	^S	51	63	33	3	83	123	53	S	115	163	73	s
20	24	14	^T	52	64	34	4	84	124	54	T	116	164	74	t
21	25	15	^U	53	65	35	5	85	125	55	U	117	165	75	u
22	26	16	^V	54	66	36	6	86	126	56	V	118	166	76	v
23	27	17	^W	55	67	37	7	87	127	57	W	119	167	77	w
24	30	18	^X	56	70	38	8	88	130	58	X	120	170	78	x
25	31	19	^Y	57	71	39	9	89	131	59	Y	121	171	79	y
26	32	1a	^Z	58	72	3a	:	90	132	5a	Z	122	172	7a	z
27	33	1b	^[	59	73	3b	;	91	133	5b	[	123	173	7b	{
28	34	1c	^\	60	74	3c	<	92	134	5c	\	124	174	7c	
29	35	1d	^]	61	75	3d	=	93	135	5d	]	125	175	7d	}
30	36	1e	^^	62	76	3e	>	94	136	5e	^	126	176	7e	~
31	37	1f	^_	63	77	3f	?	95	137	5f	_	127	177	7f	

Input / Output

```
1 int main() {  
2     int number = 65;  
3  
4     cout << char(number);  
5  
6     return 0;  
7 }
```

Type casting คือ การเปลี่ยนข้อมูล  
จากประเภทหนึ่งเป็นประเภทหนึ่ง

# Input / Output : Escape Sequences

Escape sequences เป็นรหัสที่แทรกลงไปใน string โดยผ่านเครื่องหมาย backslash

Escape Sequence	Meaning
\n	New Line
\t	Horizontal Tab
\b	BackSpace
\r	Carriage Return
\a	Audible bell
\'	Printing single quotation
\"	printing double quotation
\?	Question Mark Sequence
\\	Back Slash
\f	Form Feed
\v	Vertical Tab
\0	Null Value
\nnn	Print octal value
\xhh	Print Hexadecimal value

endl นั้นช้ากว่ากว่า '\n' เนื่องจากเบื้องหลังของคำสั่ง endl มีการเพิ่มคำสั่ง flush ต่อท้ายทุกครั้งที่เกิดการเรียกใช้

# Input / Output : Comments


เราสามารถใช้ Comment ในการเขียนอธิบายการทำงานของโค้ด ได้

- `//comment` : ใช้สำหรับ `comment` บรรทัดเดียว
- `/*comment*/` : ใช้สำหรับ `comment` หลายบรรทัด

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     // I'm Comment
7
8     /*
9         I'm Comment
10    */
11
12     return 0;
13 }
```

# Input / Output

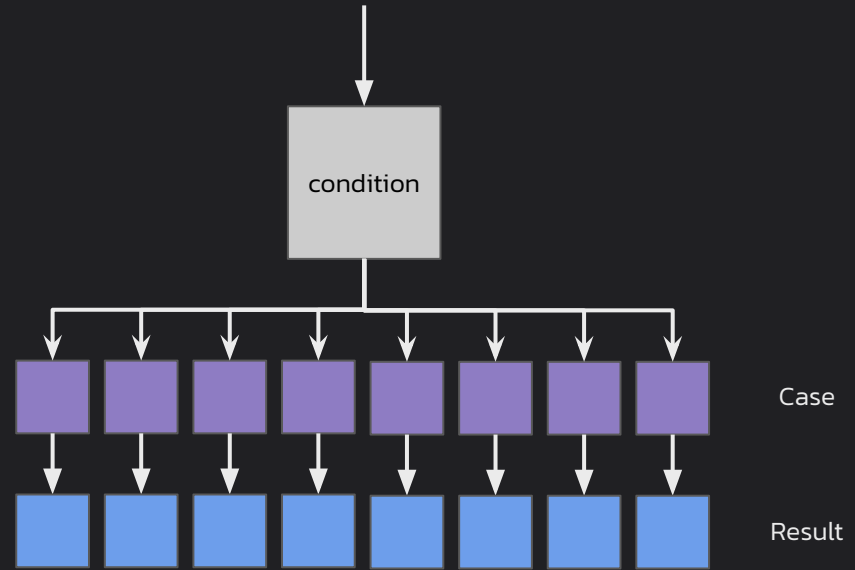
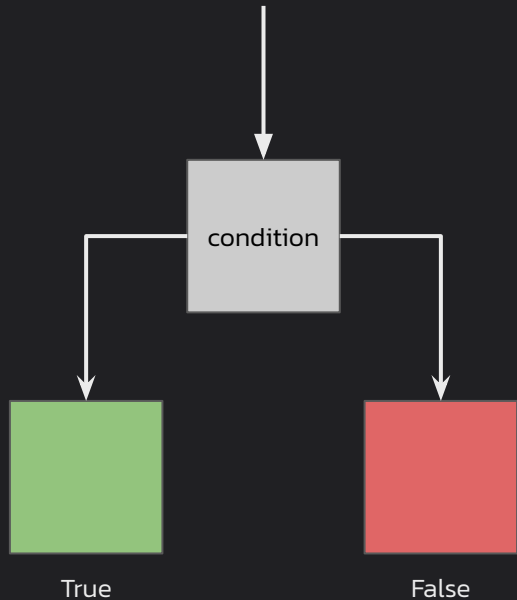
คำสั่ง `cin` นั้นเป็นคำสั่งที่ใช้ในการรับค่าใส่ตัวแปร เช่น ถ้าหากเรามีตัวแปรประเภทจำนวนเต็ม `x` อยู่ เราสามารถรับค่า `x` ได้ผ่านคำสั่ง `cin >> x;`



```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int x;
7     cin >> x;
8
9     cout << x * x;
10
11     return 0;
12 }
```

# Conditional Statement

ถ้าเรามีเงื่อนไขที่ต้องการพิจารณาว่าเป็นจริงหรือเป็นเท็จ และหลังจากที่รู้ค่าความจริงแล้ว จะทำคำสั่งใดต่อไป เราสามารถใช้คำสั่ง “เงื่อนไข” มาช่วยได้



# Conditional Statement

คำสั่งใน if จะทำงานเมื่อค่าความจริงในเงื่อนไขเป็นจริง



## Conditional Statement

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     int n;
9     cin >> n;
10
11     if(n>0) {
12         cout << "n is positive number";
13     }
14
15     return 0;
16 }
```

ประพจน์ หรือ เงื่อนไข



# Conditional Statement

คำสั่งใน else จะทำงานเมื่อค่าความจริงในเงื่อนไขเป็นเท็จ



## Conditional Statement

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     int n;
9     cin >> n;
10
11     if(n%2==1){
12         cout << "n is odd number";
13     }else {
14         cout << "n is even number";
15     }
16
17     return 0;
18 }
```

% คือ เครื่องหมาย หาสเอาเศษ

# Conditional Statement

เราสามารถใช้เงื่อนไขหลายๆ เงื่อนไขในการตรวจสอบครั้งเดียวได้

```
Conditional Statement

1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     int n;
9     cin >> n;
10
11     if(n==0) {
12         cout << "n is zero";
13     }else if(n>0) {
14         cout << "n is possitive number";
15     }else {
16         cout << "n is negative number";
17     }
18
19     return 0;
20 }
```

# Conditional Statement

เราสามารถใช้ตัวดำเนินการทางตรรกศาสตร์ได้แก่ not, and และ or ในเงื่อนไข if ได้



## Conditional Statement

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     int x=4, y=5;
9
10    if(x==4 || !(y==2) && x!=y) {
11        cout << "This statement is true";
12    }else {
13        cout << "This statement is false";
14    }
15
16    return 0;
17 }
```

ตัวดำเนินการทางตรรกศาสตร์

สัญลักษณ์ที่ใช้ในภาษา C++

และ

&&

หรือ

||

นิเสธ

!

วงเล็บ

นิเสธ

และ

หรือ

# Conditional Statement

Switch statement เป็นการระบุเงื่อนไขโดยขึ้นอยู่กับค่าของตัวแปร

```
Conditional Statement

1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     int x;
9     cin >> x;
10
11     switch(x) {
12         case 1:
13             cout << "x is 1";
14             break;
15         case 2:
16             cout << "x is 2";
17             break;
18         default:
19             cout << "x isn't 1 and 2";
20             break;
21     }
22
23     return 0;
24 }
```

กรณี x = 1

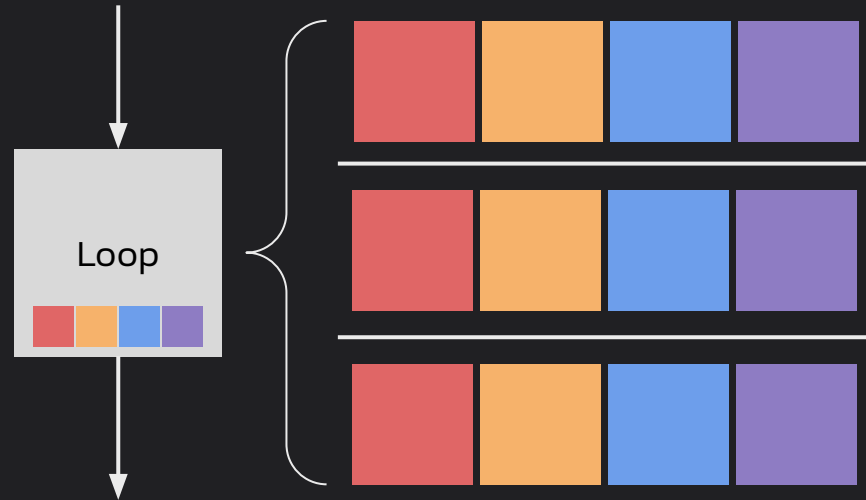
กรณีที่ x ไม่เท่ากับค่าด้านบน

## Conditional Statement : Exercise

- ให้เขียนโปรแกรมรับค่าตัวเลข และจำแนกว่าเลขนั้นเป็นจำนวนเต็มประเภทใด : (+, -, 0)
- ให้เขียนโปรแกรมรับค่าตัวเลข และจำแนกว่าเลขนั้นเป็นเลขคู่หรือเลขคี่
- ให้เขียนโปรแกรมรับค่าตัวเลข และจำแนกว่าเป็นจำนวนเต็ม หรือว่าเป็นทศนิยม

# Loops

หากต้องการลดภาระในการเขียนโปรแกรมซ้ำๆ หรือถ้าหากเราต้องการให้โปรแกรมทำงานตามจำนวนรอบที่กำหนดไว้ เราสามารถใช้คำสั่ง "loop" มาช่วยได้



# Loops

while loop ใช้เมื่อถ้าหากเราต้องการให้โปรแกรมทำงานตามเงื่อนไขบางอย่าง



## While Loop

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int n=10;
7
8     while(n!=0) {
9         cout << "Hello World\n";
10        n--;
11    }
12 }
```

เงื่อนไขการทำงาน

# Loops

ถ้าหากเราต้องการทำคำสั่งใน loop ก่อนเช็คเงื่อนไขเราสามารถใช้ do-while loop ได้



## Do-while Loop

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int n=10;
7
8     do {
9         cout << "Hello World\n";
10        n--;
11    }while(n≠0);
12 }
```

เงื่อนไขการทำงาน



# Loops

การใช้ for loop ส่วนมากแล้วจะใช้สำหรับการทำงานตามจำนวนรอบ



## For Loop

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int n=10;
7
8     for(int i=0; i<n; i++) {
9         cout << "Hello World\n";
10    }
11 }
```

เงื่อนไขการทำงาน

# Loops : Continue & Break

`continue` ใช้เมื่อต้องการให้เริ่มต้น Loop ครั้งใหม่  
`break` ใช้เมื่อต้องการให้ Loop จบการทำงาน



## Break Command

```
1 for(int i=0; i<10; i++) {  
2     if(i==5) {  
3         break;  
4     }  
5     cout << i << " ";  
6 }
```



## Continue Command

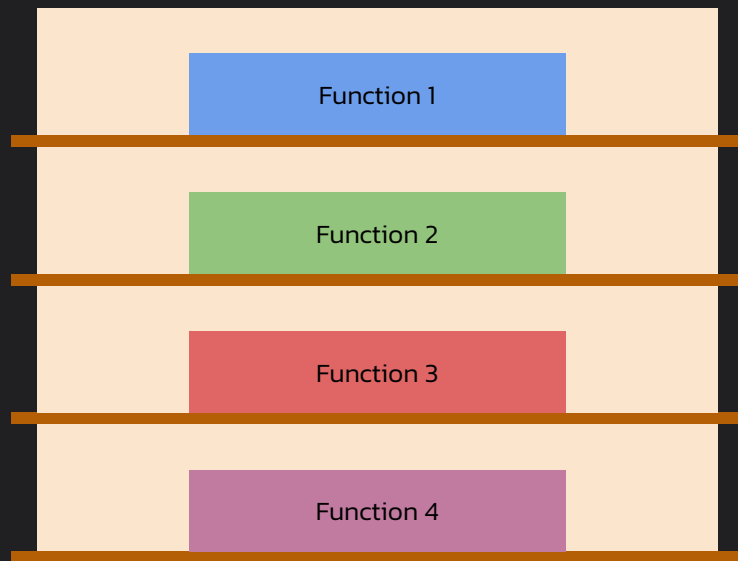
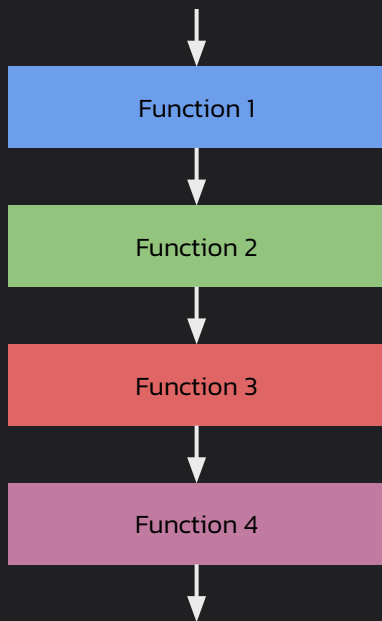
```
1 for(int i=0; i<10; i++) {  
2     if(i==5) {  
3         continue;  
4     }  
5     cout << i << " ";  
6 }
```

# Loops : Exercise

- ให้เขียนโปรแกรมแสดงผลสี่เหลี่ยมขนาด  $n \times n$
- ให้เขียนโปรแกรมแสดงผลสามเหลี่ยมขนาด  $n \times n$
- ให้เขียนโปรแกรมแสดงผลกรอบสี่เหลี่ยมขนาด  $n \times m$
- ให้เขียนโปรแกรมแสดงผลกรอบสี่เหลี่ยมโดยมีเส้นทแยงมุมขนาด  $n \times n$
- ให้เขียนโปรแกรมรับค่าตัวเลข  $n$  ตัว และหาค่า min/ max ของตัวเลขชุดนั้น
- ให้เขียนโปรแกรมรับค่าตัวเลข  $n$  ตัว และหาค่าเฉลี่ยของตัวเลขชุดนั้น
- ให้เขียนโปรแกรมเกมทายใจ โดยให้ทายตัวเลขเรื่อย ๆ จนกว่าจะเจอตัวที่ตั้งเอาไว้
- ให้เขียนโปรแกรมหาค่ารากที่สองของค่า  $x$  : Newton's Method

# Function

เป็นส่วนหนึ่งของโปรแกรมที่จะทำงานเมื่อถูกเรียกใช้ โดยเราสามารถใส่ค่าต่าง ๆ เข้าไปคำนวณ หรือกำหนดการทำงานต่าง ๆ ได้



# Function

Function ถูกแบ่งโดยใช้เกณฑ์การคืนค่าเป็น 2 ประเภท ได้แก่

## ● ● ● Non-returning function

```
1 void hello() {  
2     cout << "Hello World";  
3 }
```

Function นี้จะไม่มีการคืนค่ากลับมา

## ● ● ● Returning function

```
1 string hello() {  
2     return "Hello World";  
3 }
```

ประเภทของตัวแปรที่ต้องการคืนค่า

Function นี้จะมีการคืนค่ากลับมา

# Function : Parameter & Argument

เราสามารถนำตัวแปรเข้าไปคำนวณใน Function ได้ ผ่าน parameter

● ● ● Parameter & Argument

```
1 #include <iostream>
2
3 using namespace std;
4
5 int add(int a, int b) {
6     return a+b;
7 }
8
9 int main() {
10
11     cout << add(10, 5);
12     return 0;
13 }
```

Parameters

Argument

# Function : Prototype

Prototype คือ การประกาศฟังก์ชันโดยปราศจากส่วนของการทำงาน



Function Prototype

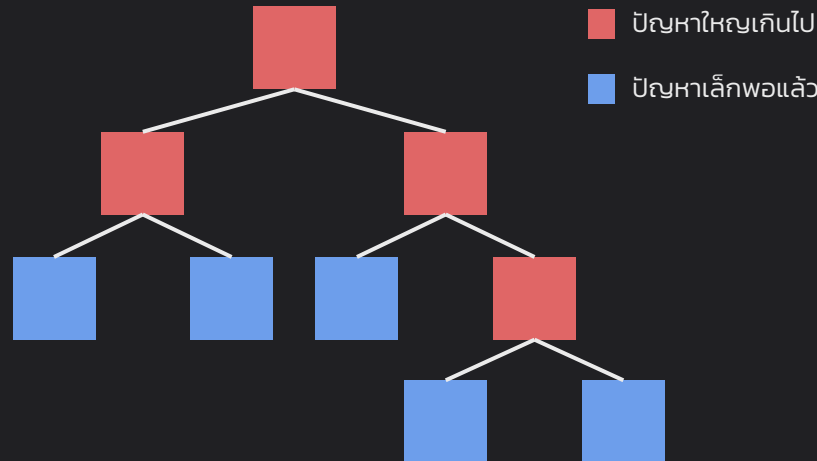
```
1 #include <iostream>
2
3 using namespace std;
4
5 int add(int a, int b);
6
7 int main() {
8     return 0;
9 }
10
11 int add(int a, int b) {
12     return a+b;
13 }
```

Function Prototype

# Function : Recursive Function

**Recursive Function** คือ ฟังก์ชันที่เรียกตัวเองไปเรื่อย ๆ จนกว่าจะเจอเงื่อนไขให้หยุด

- เราจะมอง Recursive Function เป็นคนที่ขี้เกียจสุด ๆ ไปเลย
- โดยแนวคิดของฟังก์ชันนั้นจะเป็น
  - ถ้าปัญหามีขนาดเล็กพอให้ทำให้เสร็จไปเลย
  - ถ้าปัญหามีขนาดใหญ่ไปจะแก้แค่บางส่วน ส่วนที่เหลือส่งให้ร่างโคลนของตัวเองทำ



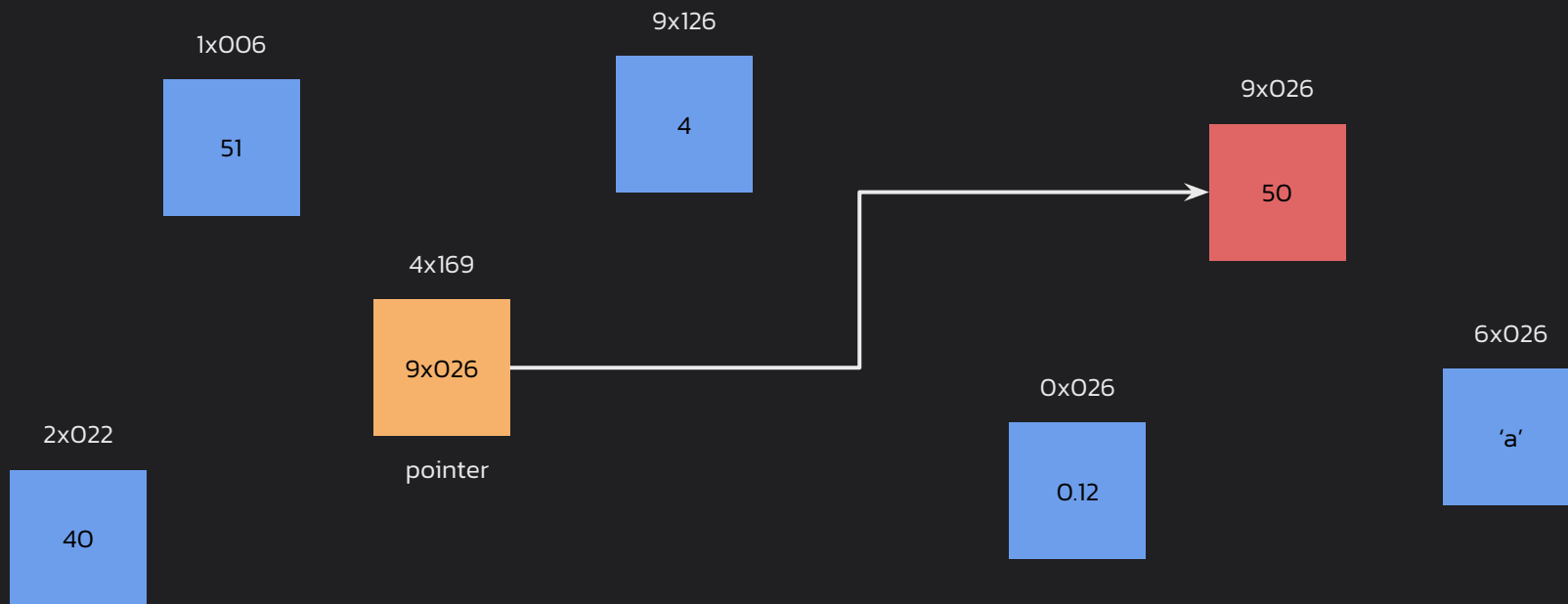


# Function : Exercise

- จงเขียน Function แสดงค่า "Hello World" : `hi()`;
- จงเขียน Function บวกเลข 3 ตัวได้แก่ a, b และ c : `add(a, b, c)`;
- จงเขียน Function หาค่า min/ max ของตัวเลข a และ b : `min(a, b)`, `max(a, b)`;
- จงเขียน Function เปลี่ยนค่าเงิน โดยสามารถกำหนดค่าเงินต้นทางและปลายทางได้ : `exchange(a, "THB", "USD")`;
- จงเขียน Function `Countdown N to 1` โดยใช้ Recursive Function
- จงเขียน Function `Factorial` โดยใช้ Recursive Function
- จงเขียน Function `Fibonacci` โดยใช้ Recursive Function
- จงเขียน Function `Reverse String` โดยใช้ Recursive Function
- จงเขียน Function `Tower of Hanoi` โดยใช้ Recursive Function

# Pointer

Pointer ใช้สำหรับเก็บค่าของตำแหน่งของข้อมูลในหน่วยความจำ



# Pointer

$*x$  : ใช้สำหรับถามหาค่าของตัวแปรที่ pointer  $x$  นั้นชี้อยู่  
 $\&x$  : ใช้สำหรับถามหาตำแหน่งของตัวแปร  $x$



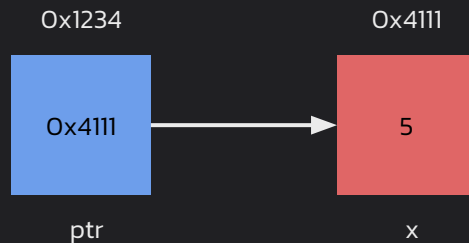
Pointer

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     ios_base::sync_with_stdio(0), cin.tie(0);
7
8     int* ptr;
9     int x=5;
10    ptr = &x;
11
12    cout << *ptr;
13
14    return 0;
15 }
```

ประกาศ pointer ชื่อ ptr

ptr เก็บที่อยู่ของ x

แสดงค่าในตัวแปร x



# Pass by Value & Pass by Reference



## Pass by Value

```
1 #include <iostream>
2
3 using namespace std;
4
5 void changeX(int x) {
6     x = 10;
7 }
8
9 int main() {
10     int x=5;
11
12     changeX(x);
13
14     cout << x; // 5
15
16     return 0;
17 }
```



## Pass by Reference

```
1 #include <iostream>
2
3 using namespace std;
4
5 void changeX(int* x) {
6     *x = 10;
7 }
8
9 int main() {
10     int x=5;
11
12     changeX(&x);
13
14     cout << x; // 10
15
16     return 0;
17 }
```

# Pointer : Exercise

- จงเขียนโปรแกรมให้ pointer ชี้ที่ตำแหน่งของตัวแปรประเภท int แล้วอ่านค่าของตัวแปรผ่าน pointer
- จงเปลี่ยนค่าของตัวแปรผ่าน Function **\*\* โดยใช้ pointer \*\***
- จงเขียน Function ให้สลับค่าตัวแปร a และ b : swap(a, b);
- จงเขียนโปรแกรมให้ pointer ชี้ pointer ที่เป็น pointer of int

# Array

เราสามารถใช้ array ในการเก็บข้อมูลชนิดเดียวกันที่มีจำนวนมากได้ ซึ่งการเก็บข้อมูลใน array นั้นรวดเร็ว และง่ายต่อการใช้งานเป็นอย่างมาก

0x000	0x001	0x002	0x003	0x004	0x005	0x006	0x007
1	5	3	0	-11	21	-1	-1
0	1	2	3	4	5	6	7

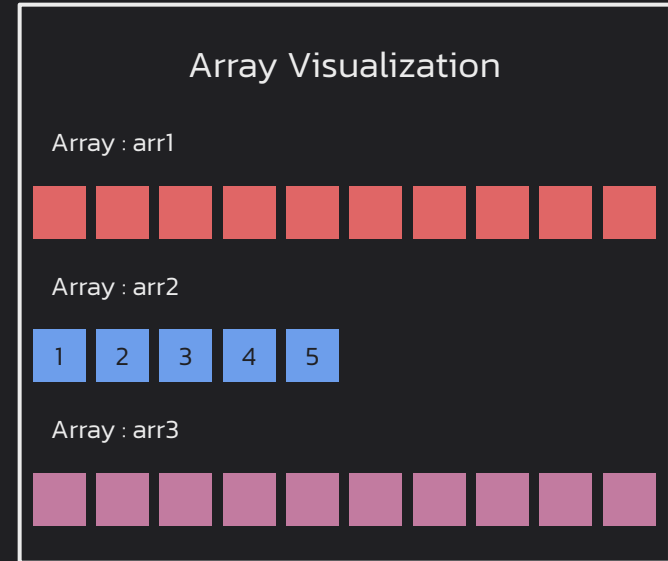
# Array : Usage

Array

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int arr1[10];
7     int arr2[] = {1, 2, 3, 4, 5};
8
9     int n=10;
10    int arr3[n];
11
12    int arr2_length = sizeof(arr2) / sizeof(arr2[0]);
13
14    cout << arr1[0] << " " << arr2[0] << " " << arr3[0];
15
16    return 0;
17 }
```

การประกาศ Array

การเข้าถึงตำแหน่งใน Array



การคำนวณหาความยาวของ Array

# Array : Local Variable & Global Variable

โดยปกติแล้วถ้าเราประกาศตัวแปรแบบ Global แล้วไม่ได้ประกาศค่าไว้ตัวแปรตัวนั้นจะมีค่าเป็นค่า 0 โดยอัตโนมัติ

- `Int` => 0
- `Bool` => false
- `Char` => `'\0'`
- `Float` => 0

```
Local Variable & Global Variable

1 #include <iostream>
2
3 using namespace std;
4
5 int arr[10];
6 int x;
7
8 int main() {
9     int y;
10
11     cout << y << " " << x << " " << arr[0];
12
13     return 0;
14 }
```



# Array : Exercise

- ให้เขียนโปรแกรมรับตัวเลข  $n$  ตัว แล้วถามหาผลรวมในช่วง  $m$  ครั้ง :  $[l, r]$
- ให้เขียนโปรแกรมรับตัวเลข  $n$  ตัว แล้วถามหาค่า  $\min/\max$  ในช่วง  $m$  ครั้ง :  $[l, r]$
- ให้เขียนโปรแกรมบวก Matrix ขนาด  $n \times m$
- ให้เขียนโปรแกรมคูณ Matrix ขนาด  $n \times r$  และ  $r \times m$
- ให้เขียนโปรแกรมแสดงผลจำนวนเฉพาะในช่วง  $l$  ถึง  $r$  :  $[l, r]$
- ให้เขียนโปรแกรมเรียงข้อมูลใน array
- ให้เขียนโปรแกรมรับตัวเลข  $n$  ตัว แล้วถามว่าในข้อมูลชุดนั้นมีข้อมูลที่ต่างกันกี่ตัว

# String

String หรือข้อความ คือ array ของตัวอักษร เช่น "Hi" จะถูกมองเป็น ['H', 'i', '\0']

0x000	0x001	0x002	0x003	0x004	0x005
H	e	l	l	o	\0
0	1	2	3	4	5

# String

เราสามารถสร้าง String ได้ด้วยการใช้ Array of Character ได้



Comment

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     char s[10] = "HI";
7
8     cout << s;
9
10    return 0;
11 }
```

Array of Character

# String

ในภาษา C++ มีตัวแปรประเภท `string` ให้ใช้ได้เลย จึงไม่จำเป็นต้องสร้างเป็น array of character

```
String

1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     string s = "Hello World";
7
8     cout << s << '\n';
9     cout << s.length() << '\n';
10
11     for(int i=0; i<s.length(); i++) {
12         cout << s[i] << ' ';
13     }
14
15     return 0;
16 }
```

ใช้ถามหาความยาวของ String

อ้างอิงตำแหน่งเหมือน array ปกติ

# String : Exercise

- ให้เขียนโปรแกรมรับชื่อ และปีเกิด พร้อมแสดงผลข้อมูลชื่อ และอายุตามลำดับ
- ให้เขียนโปรแกรมรับข้อความ a และ b แล้วให้แสดงค่า ab ออกมา
- ให้เขียนโปรแกรมรับประโยค และแสดงผลออกมา
- ให้เขียนโปรแกรมกลับข้อความ จากหน้าไปหลัง และเปลี่ยนจากหลังไปหน้า
- ให้เขียนโปรแกรมตรวจสอบว่าข้อความที่รับเข้ามาเป็น palindrome หรือไม่
- ให้เขียนโปรแกรมตรวจสอบข้อความว่า ในข้อความนั้นประกอบไปด้วย ตัวอักษรภาษาอังกฤษพิมพ์เล็ก หรือพิมพ์ใหญ่ หรือทั้งคู่ : lower, upper or both
- ให้เขียนโปรแกรมตรวจสอบว่าข้อความที่รับเข้ามามีตัวอักษรภาษาอังกฤษติดกันเกิน 10 ตัว หรือไม่
- ให้เขียนโปรแกรมเปลี่ยนข้อความที่รับเข้ามาเป็นตัวอักษรภาษาอังกฤษพิมพ์เล็กทั้งหมด
- จงเขียนโปรแกรมเข้ารหัสข้อมูล : `encode(s, p, m);`

# Structure & Union

Structure และ Union คือประเภทของตัวแปรที่เรากำหนดขึ้นมาเอง



## Structure

```
1 struct Student{  
2   int id;  
3   string name;  
4   float weight, height;  
5 };
```

id

name

weight

height

42358

RuffLogix

50.0

170.0

Student [ Struct ]



## Union

```
1 union Student{  
2   int id;  
3   string name;  
4   float weight, height;  
5 };
```

???

???

Student [ Union ]

## Structure & Union : Exercise

- ให้ออกแบบ Structure เก็บข้อมูลจุด โดยประกอบไปด้วยพิกัด x และ y
- ให้ออกแบบ Structure เก็บข้อมูลนักเรียน โดยประกอบไปด้วยพิกัด student id, name, age และ สถานะการสอบ (ผ่าน/ ไม่ผ่าน)
- ให้ออกแบบ Structure เก็บข้อมูลของสัตว์ โดยประกอบไปด้วย weight, height, n\_legs, have\_tail, hp, atk\_damage และพฤติกรรมต่าง ๆ เช่น walk(), run(), sleep(), communicate\_with(a, b) เป็นต้น