


String

in C++

String ในภาษา C++

String ใน C++ คืออะไร ?

ใน C++ String คือ Object std::string ไม่ใช่อาร์เรย์ธรรมดา

แต่เป็น คลาส (Class) หรือ วัตถุ (Object) ที่อยู่ในไลบรารีมาตรฐาน <string>

ข้อดี

ยืดหยุ่น : สามารถเพิ่มหรือลดขนาดได้อัตโนมัติ (Dynamic)

ปลอดภัย : มีฟังก์ชันจัดการมากมายในตัว (เช่น การบวก, การค้นหา)

ใช้งานง่าย : มีตัวดำเนินการต่าง ๆ ที่ถูกโอเวอร์โหลดให้ใช้งานง่าย (เช่น + สำหรับการต่อ String)

String ในภาษา C++

การประกาศตัวแปร String

Syntax: การประกาศและการกำหนดค่า กรณีใช้คำสั่ง using namespace std;

รูปแบบการประกาศ	ตัวอย่าง	คำอธิบาย
ประกาศว่าง	string name;	ประกาศตัวแปร name ที่ยังไม่มีข้อความใด ๆ
กำหนดค่าเริ่มต้น	string text = "Hello";	กำหนดค่า "Hello" ให้ตัวแปร text ทันที
ประกาศซ้ำ	string s(5, 'A');	สร้าง String ที่มีอักขระ 'A' จำนวน 5 ตัว ("AAAAA")

String ในภาษา C++

โค้ดตัวอย่าง

Syntax: การประกาศและการกำหนดค่า กรณีใช้คำสั่ง using namespace std;

```
#include <bits/stdc++.h>
using namespace std;
string my_str;
string greeting = "Hi";
string stars(10, '*');
```

String ในภาษา C++

การรับค่า String

1. cin >> s; // รับค่า จนกระทั่งเจอช่องว่าง (Space), Enter หรือ Tab เท่านั้น
2. getline(cin, s); //รับค่า ทั้งหมด ในบรรทัด จนกว่าจะเจอขึ้นบรรทัดใหม่ (Enter)

String ในภาษา C++

การเข้าถึงตัวอักษรและความยาว

- การเข้าถึงตัวอักษร (Accessing Characters)

  ใช้เครื่องหมายวงเล็บ [ ] เช่นเดียวกับอาร์เรย์ การนับดัชนี (Index) เริ่มจาก 0

String s	Index 0	Index 1	Index 2
“C++”	C	+	+

- การหาความยาว (Length)

  ใช้ฟังก์ชัน: s.length() หรือ s.size()

String ในภาษา C++

ตัวอย่างการเข้าถึงตัวอักษรและความยาว

```
string s = "ComputerScience";
int len = s.length();
cout << "ความยาว (Length): " << len << endl; // Output: 15
char char_at_8 = s[8];
cout << char_at_8 << endl; // Output: S (ตัว S ของ Science)
```

String ในภาษา C++

ตัวอย่างการเข้าถึงตัวอักษรและความยาว

```
string s = "ComputerScience";
int len = s.length();
cout << "ความยาว (Length): " << len << endl; // Output: 15
char last_char = s[len - 1];
cout << last_char << endl; // Output: e
s[0] = 'K';
cout << s << endl; // Output: KomputerScience
```

String ในภาษา C++

การวนลูป (Looping) บน String

ใช้ **for** Loop ในการประมวลผล

เนื่องจาก String เป็นลำดับของอักขระ การวนลูปจึงเป็นหัวใจสำคัญในการจัดการ String

เมื่อต้องการทราบตำแหน่ง (Index) ของตัวอักษร

```
string data = "ABCDE";
for (int i = 0; i < data.length(); ++i) {     // พิมพ์ตัวอักษรพร้อมตำแหน่ง
       cout << "Index " << i << ": " << data[i] << endl;
}
```

String ในภาษา C++

```
#include <bits/stdc++.h>
using namespace std;
int main(){
string data = "ABCDE";
        for (int i = 0; i < data.length(); ++i) {
        cout << "Index " << i << ": " << data[i] << endl;
        }
        return 0;
}
```

String ในภาษา C++

```
#include <bits/stdc++.h>
using namespace std;
int main(){
string data = "ABCDE";
        for (int i = 0; i < data.length(); ++i) {
        cout << "Index " << i << ": " << data[i] << endl;
        }
        return 0;
}
```

```
Index 0: A
Index 1: B
Index 2: C
Index 3: D
Index 4: E
```

Worksheet ใช้ลูปเพื่อเขียนโปรแกรมที่แสดง String และ index แบบย้อนกลับ (Reverse)

Ex.

string data = “COMPUTER”

Output : Index 7: R

Index 6: E

Index 5: T

Index 4: U

Index 3: P

Index 2: M

Index 1: O

Index 0: C

String ในภาษา C++

```
#include <bits/stdc++.h>
using namespace std;
int main(){
string data = "COMPUTER";
        for (int i = data.length()-1; i ≥ 0; --i) {
                cout << "Index " << i << ": " << data[i] << endl;
        }
        return 0;
}
```

ฟังก์ชันและตัวดำเนินการพื้นฐาน (Basic Operations)

- **การเชื่อมต่อ String (Concatenation)**

  ใช้ตัวดำเนินการ + หรือ += เพื่อรวม String เข้าด้วยกัน

ตัวดำเนินการ	ตัวอย่าง	ผลลัพธ์
+	"A" + "B"	"AB"
+=	S += "Word"	ต่อ "Word" เข้าที่ท้าย S

การเชื่อมต่อ String (Concatenation)

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string first = "Comp";
        string second = "Sci";
        string full = first + second + "ence";
        cout << "Full Name: " << full << endl; // Output: CompScience
        full += " (POSN)";
        cout << "Extended: " << full << endl; // Output: CompScience (POSN)
        return 0;
}
```

ฟังก์ชันและตัวดำเนินการพื้นฐาน (Basic Operations)

- **การเปรียบเทียบ (Comparison)**

  ใช้ตัวดำเนินการทางคณิตศาสตร์ (==, !=, <, >, <=, >=)

  เพื่อเปรียบเทียบ String ตามลำดับพจนานุกรม (Lexicographical Order)

  เปรียบเทียบตัวอักษรทีละตัวจากซ้ายไปขวา โดยอิงตามรหัส ASCII (เช่น 'a' > 'A', '2' > '1')


String ในภาษา C++

dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char
0	0	000	NULL	32	20	040	space	64	40	100	@	96	60	140	`
1	1	001	SOH	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT	36	24	044	$	68	44	104	D	100	64	144	d
5	5	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB	41	29	051	)	73	49	111	I	105	69	151	i
10	a	012	LF	42	2a	052	*	74	4a	112	J	106	6a	152	j
11	b	013	VT	43	2b	053	+	75	4b	113	K	107	6b	153	k
12	c	014	FF	44	2c	054	,	76	4c	114	L	108	6c	154	l
13	d	015	CR	45	2d	055	-	77	4d	115	M	109	6d	155	m
14	e	016	SO	46	2e	056	.	78	4e	116	N	110	6e	156	n
15	f	017	SI	47	2f	057	/	79	4f	117	O	111	6f	157	o
16	10	020	DLE	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM	57	39	071	9	89	59	131	Y	121	79	171	y
26	1a	032	SUB	58	3a	072	:	90	5a	132	Z	122	7a	172	z
27	1b	033	ESC	59	3b	073	;	91	5b	133	[	123	7b	173	{
28	1c	034	FS	60	3c	074	<	92	5c	134	\	124	7c	174	|
29	1d	035	GS	61	3d	075	=	93	5d	135	]	125	7d	175	}
30	1e	036	RS	62	3e	076	>	94	5e	136	^	126	7e	176	~
31	1f	037	US	63	3f	077	?	95	5f	137	_	127	7f	177	DEL


String ในภาษา C++

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string a = "apple";
        string b = "Apple";
        string c = "app";
        if (a > b) { // 'a' มีค่า ASCII มากกว่า 'A'
                cout << "apple > Apple: True" << endl;
        }
        if (c < a) { // 'app' เป็นส่วนนำหน้าของ 'apple'
                cout << "app < apple: True" << endl;
        }
        return 0;
}
```

```
apple > Apple: True
app < apple: True
```

Workshop การนับและตรวจสอบ

โจทย์ฝึกหัด : นับองค์ประกอบใน String ให้เขียนโปรแกรมรับข้อความ 1 บรรทัด จากนั้นนับจำนวนตัวอักษรตัวพิมพ์ใหญ่ และจำนวน ตัวเลข ที่ปรากฏในข้อความนั้น

แนวทางการแก้ปัญหา:ใช้ getline() รับ String ทั้งหมดใช้ลูป for วนอ่านตัวอักษรทุกตัว

ใช้ฟังก์ชัน isupper(c) และ isdigit(c) จากไลบรารี <cctype>

(ซึ่งถูก Include มาแล้วใน <bits/stdc++.h>) เพื่อตรวจสอบเงื่อนไข


String ในภาษา C++

ฟังก์ชันการค้นหาและการตัด String

การค้นหาตำแหน่ง (Find Position) ใช้เมธอด **log.find()** เพื่อหาตำแหน่ง (Index) ของ String ย่อย หรืออักขระที่ต้องการ

ตัวอย่าง string log = "USER-101; Login SUCCESS; Timestamp: 1400";

ฟังก์ชัน	การใช้งาน	ผลลัพธ์
log.find(sub)	หาตำแหน่งที่ พบครั้งแรก	8
log.rfind(sub)	หาตำแหน่งที่ พบครั้งแรก (เริ่มจากท้าย)	23

String ในภาษา C++

```cpp
#include <bits/stdc++.h>
using namespace std;
int main() {
        string log = "USER-101; Login SUCCESS; Timestamp: 1400";
        cout << "Text : " << log << endl;
        size_t pos_semi = log.find(';');
        cout << "find ; index " << pos_semi << endl; // ผลลัพธ์: 8
        pos_semi = log.rfind(';');
        cout << "rfind ; index " << pos_semi << endl; // ผลลัพธ์: 23
        // กรณีหาข้อความที่ไม่มีในสตริง
        size_t pos_fail = log.find("ADMIN");
        if (pos_fail == string::npos) {
                cout << "Message not found 'FAIL' (npos)" << endl;
        }
        return 0;
}
```

size_t คือ ชนิดข้อมูลเลขจำนวนเต็มแบบไม่ติดลบ (Unsigned Integer) ที่ถูกออกแบบมาโดยเฉพาะเพื่อใช้เก็บ "ขนาด" หรือ "จำนวน" ของสิ่งต่าง ๆ ในหน่วยความจำ เช่น ขนาดของอาร์เรย์, ความยาวของสตริง, หรือจำนวนสมาชิกใน vector

ฟังก์ชันการค้นหาและการตัด String

การตัด String ย่อย (Substring) ใช้เมธอด data.substr(pos, len) เพื่อดึงส่วนหนึ่งของ String ออกมา

Argument	คำอธิบาย
pos	ตำแหน่งดัชนีเริ่มต้น ที่ต้องการตัด (บังคับ)
len	ความยาว ของ String ที่ต้องการตัด (ถ้าละเว้น จะตัดไปจนสุด String)

String ในภาษา C++

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    string date = "2025-09-29";

    // ตัด Year (จากดัชนี 0, ยาว 4 ตัว)
    string year = date.substr(0, 4);
    cout << "Year: " << year << endl; // Output: 2025

    // ตัด Day (จากดัชนี 8, ยาว 2 ตัว)
    string day = date.substr(8, 2);
    cout << "Day: " << day << endl; // Output: 29

    // ตัด Month โดยละเว้นความยาว (จะตัดไปจนสุด)
    string rest = date.substr(5);
    cout << "Rest: " << rest << endl; // Output: 09-29

    return 0;
}
```

Workshop การแยกชื่อไฟล์และนามสกุล

โจทย์ฝึกหัด : ให้เขียนโปรแกรมรับชื่อไฟล์เต็ม (Full Filename) และแยกส่วนชื่อไฟล์ และส่วนนามสกุล (Extension) ออกจากกัน

Input : report.data.v1.csv

Output ที่ต้องการ :

ชื่อไฟล์ : report.data.v1

นามสกุล : csv

แนวทางการแก้ปัญหา :

1. ใช้ rfind('.') เพื่อหาตำแหน่งของเครื่องหมายจุด (.) ตัวสุดท้าย
2. ใช้ substr() เพื่อตัด String ส่วนนามสกุลออกมา (จากตำแหน่งหลังจุดสุดท้าย)
3. ใช้ substr() อีกครั้งเพื่อตัด String ส่วนชื่อไฟล์ (จากดัชนี 0 ไปจนถึงก่อนจุดสุดท้าย)


String ในภาษา C++

การแปลง String กับตัวเลข

การทำโจทย์คอมพิวเตอร์ เราอาจต้องรับ String (เช่น "123") และแปลงเป็นตัวเลข (123) เพื่อนำไปคำนวณ หรือเมื่อคำนวณเสร็จแล้ว ก็ต้องแปลงกลับเป็น String เพื่อแสดงผลลัพธ์

การแปลง	ฟังก์ชัน	ตัวอย่าง
Number ⟶ String	to_string(num)	to_string(123) ⟶ "123"
String ⟶ Number	stoi(s)	stoi("456") ⟶ 456 (Integer)
String ⟶ Double	stod(s)	stod("3.14") ⟶ 3.14 (Double)

String ในภาษา C++

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string s1 = "100";
        string s2 = "25";

        int num1 = stoi(s1); // 1. String → Number (เพื่อนำไปคำนวณ)
        int num2 = stoi(s2);

        int sum = num1 + num2;
        cout << "SUM " << sum << endl; // Output: 125

        // 2. Number → String (เพื่อนำไปต่อกับข้อความ)
        string result_str = "RESULT " + to_string(sum);
        cout << result_str << endl; // Output: ผลลัพธ์คือ: 125
        return 0;
}
```

การแก้ไข String append(), insert(), replace()

String ใน C++ สามารถแก้ไขได้ง่ายโดยใช้เมธอดเหล่านี้ append(), insert(), replace() ซึ่งมีประโยชน์มากเมื่อต้องจัดการข้อมูลใน String

- **การต่อท้าย (Append)** ฟังก์ชัน **s.append(other_string);**
- **การแทรก (Insert)** ฟังก์ชัน **s.insert(pos, string_to_insert);**
- **การแทนที่ (Replace)** ฟังก์ชัน **s.replace(pos, len, replacement_string);**

- **การต่อท้าย (Append)** ฟังก์ชัน **s.append(other_string);**

  ใช้สำหรับ ต่อ String หรือ Substring เข้าไปที่ท้าย String เดิม

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string s = "Hello";
        string suffix = " World";
        // ต่อ String ทั้งหมด
        s.append(suffix);
        cout << "Append Full: " << s << endl; // Output: Hello World
        // ต่อเพียงบางส่วน " C++" (จากดัชนี 0, ยาว 4 ตัว)
        s.append(" C++ Language", 4);
        cout << "Append Part: " << s << endl; // Output: Hello World C++
        return 0;
}
```

- **การแทรก (Insert)** ฟังก์ชัน **s.insert(pos, string_to_insert);**

  ใช้สำหรับ แทรก String หรือ Substring เข้าไปในตำแหน่ง (Index) ที่ต้องการ

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string date = "20250929";
        // แทรก '-' ที่ตำแหน่ง 4 (ระหว่าง Year กับ Month)
        date.insert(4, "-");
        cout << "Insert 1: " << date << endl; // Output: 2025-0929
        // แทรก '-' ที่ตำแหน่ง 7 (ระหว่าง Month กับ Day)
        date.insert(7, "-");
        cout << "Insert 2: " << date << endl; // Output: 2025-09-29
        return 0;
}
```

- **การแทนที่ (Replace)** ฟังก์ชัน **s.replace(pos, len, replacement_string);**

  ใช้สำหรับ แทนที่ ส่วนหนึ่งของ String ด้วย String ใหม่

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string config = "LogLevel: DEBUG";
        string s = "System";
        // แทนที่คำว่า "DEBUG" ด้วย "ERROR"
        // (เริ่มที่ดัชนี 10, ยาว 5 ตัว)
        config.replace(10, 5, "ERROR");
        cout << "Replace 1: " << config << endl; // Output: LogLevel: ERROR
        // แทนที่คำว่า "LogLevel" (ดัชนี 0, ยาว 8 ตัว) ด้วย string s
        config.replace(0, 8, s);
        cout << "Replace 2: " << config << endl; // Output: System: ERROR
        return 0;
}
```

การแปลงตัวพิมพ์เล็ก/ใหญ่

ที่ใช้เทคนิค Range-based for loop with Reference เพื่อเปลี่ยน String ต้นฉบับโดยตรง

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string text = "CompUter ScieNCE 101";
        cout << "Original: " << text << endl;
        for (char &c : text) {   // 1. แปลงเป็นตัวพิมพ์ใหญ่ทั้งหมด (UPPERCASE)
                c = toupper(c); // เปลี่ยนตัวอักษรใน text โดยตรง
        }
        cout << "To Upper: " << text << endl; // Output: COMPUTER SCIENCE 101

        for (char &c : text) { // 2. แปลงกลับเป็นตัวพิมพ์เล็กทั้งหมด (LOWERCASE)
                c = tolower(c); // เปลี่ยนตัวอักษรใน text โดยตรง
        }
        cout << "To Lower: " << text << endl; // Output: computer science 101
        return 0;
}
```

การแทนที่ตัวอักษร

ที่ใช้เทคนิค Range-based for loop with Reference เพื่อเปลี่ยน String ต้นฉบับโดยตรง

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    string s = "i-am-a-string";

    // ใช้อ้างอิง (&) เพื่อแก้ไข String ต้นฉบับ
    for (char &c : s) {
        if (c == '-') {
            c = '_'; // เปลี่ยนเครื่องหมาย '-' เป็น '_'
        }
    }
    cout << "Edited String: " << s << endl; // Output: i_am_a_string
    return 0;
}
```

String Stream: การแยกข้อมูลที่ซับซ้อน (Tokenization)

```
#include <bits/stdc++.h>
using namespace std;
int main() {
        string data_line = " ";
        getline(cin, data_line); //ตัวอย่างใส่ข้อมูล ItemX 5 19.99
        stringstream ss(data_line); // สร้าง String Stream จาก String ที่รับมา
        // ประกาศตัวแปรสำหรับเก็บโทเคนแต่ละส่วน
        string item_name;
        int quantity;
        double price;
        ss >> item_name >> quantity >> price; // ดึงข้อมูลออกจาก Stream ทีละส่วนตามประเภทตัวแปร
        cout << "Item: " << item_name << endl;     // Output: ItemX
        cout << "Quantity: " << quantity << endl;  // Output: 5
        cout << "Price: " << fixed << setprecision(2) << price << endl; // Output: 19.99
        return 0;
}
```

- **ให้นักเรียนลองออกแบบโค้ดสำหรับโจทย์ : Run-Length Encoding (RLE)**

  คือการแทนที่ ลำดับของข้อมูลที่ซ้ำกันติดกัน (A Run) ด้วยการบันทึกเพียง สองค่า คือ

  - จำนวนครั้งที่ซ้ำ (Count หรือ Length)
  - ตัวข้อมูลที่ซ้ำกัน (The Data/Value)

Ex.

input : AAAAABBBCCDAA

output : 5A3B2CD2A