Lecture 3

TOC

- 1. C++ □□
- 2. Hello, World!
- 3. _____
- 4. 00000
- 5. 00000
- 6.
- 7. 00000
- 8. Control Flow
- 9. Statement

C++	
ПП	

- 0 C 00000C++ 00000000 C 000000
- 00000000 C++ 00000000000000

Hello, World!

```
00000 C++ 00
```

```
#include <iostream>
int main() {
   std::cout << "Hellor, World!\n";
   return 0;
}</pre>
```



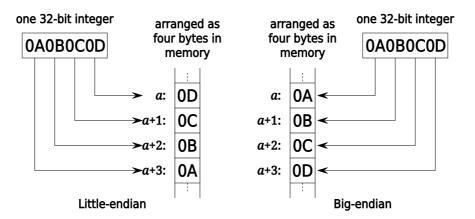
```
g++ hello_world.cpp -o hello_world
./hello_world
```



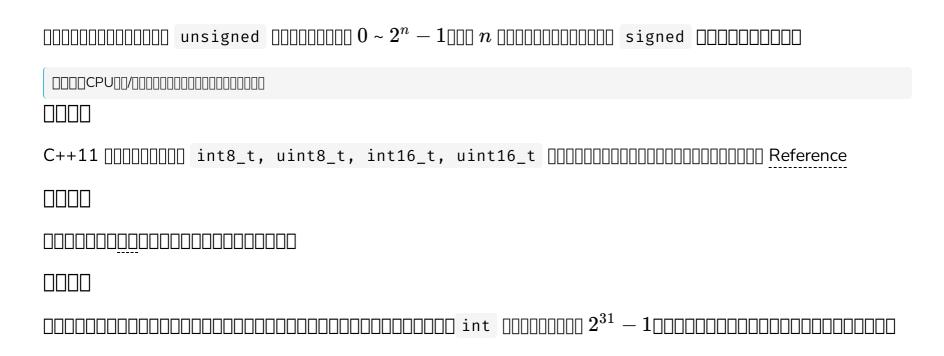








bool	bool	1	true/false
char	char	1	$-2^7 \sim 2^7 - 1$
short	short	2	$-2^{15} \sim 2^{15} - 1$
int	int	4	$-2^{31} \sim 2^{31} - 1$
long	long	4	$-2^{31} \sim 2^{31} - 1$
long long	long long	8	$-2^{63} \sim 2^{63} - 1$



float	float	4	3.4e-38 ~ 3.4e38
double	double	8	1.7e-308 ~ 1.7e308
long double	long double	12	3.4e-4932 ~ 1.1e4932

000001EEE 75400000000000000000

[[[[][(Since C++23)

- lacksquare
- NaN(Not a Number): 0/0, ∞/∞ \square

```
std::numeric_limits<double>::infinity();
std::numeric_limits<double>::quiet_NaN();
std::numeric_limits<double>::signaling_NaN();
```

void □□

- - C-style cast: (type)expression
 - C++ style cast: static_cast, dynamic_cast, const_cast, reinterpret_cast

```
int a = 10;
float b = (float)a; // C-style cast
double b = static_cast<double>(a); // C++ style cast
```

 $\square\square\square\square\square$ char \rightarrow int \rightarrow long \rightarrow long long $\square\square\square$

- [[[[] int [[[] int

C++ style cast: static_cast, dynamic_cast, $\square\square\square\square\square\square$ float \rightarrow double \rightarrow long double $\square\square\square$

```
#include <iostream>
int main() {
    char a= 127;
    char b = 3;
    int c = a + b;
    std::cout << c << std::endl;
    return 0;
}</pre>
```

C++ style cast v.s. C-style cast

- C-style cast □□□□□□□

```
constexpr int x = 10;
constexpr int y = 10;
constexpr int sum = x + y; // 00000

const int a = 10;
a = 20; // Error
```

```
int a = 10; // 00000
float b = 3.14; // 000000
char c = 'a'; // 000000
bool d = true; // 000000
char e[] = "Hello, World!"; // 000000
```

```
int a = 10;
int* p = &a; // 00
int& r = a; // 00

int* p = nullptr; // 000
*a = 11; // 00000000
r = 12; // 00000000
```

- 0000000000



- 000000000 ASCII 000UTF-8 000UTF-16 0000
- □□□□□□□□□ R"()" □□□□□□□□□□□□□□□□□□ R"(Hello, \n World!)" □

```
int arr[5] = {1, 2, 3, 4, 5}; // 0000
int arr[5]; // 0000
int arr[] = {1, 2, 3, 4, 5}; // 0000000
int arr[5] = {1, 2}; // 222222, 222 {1, 2, 0, 0, 0}

arr[0] = 10; // 00000000
*(arr + 1) = 20; // 222222222
```

Reference:

- https://en.cppreference.com/w/c/language/array
- https://en.cppreference.com/w/c/language/array_initial

```
# include <iostream>
// void foo(int *args)
// void foo(int args[5])
void foo(int args[]) {
  // args [[[[]]
  static assert(sizeof(args) = sizeof(int*), "Error");
  std::cout << sizeof(args) << std::endl; // 8</pre>
int main() {
  int arr[5] = \{1, 2, 3, 4, 5\};
  std::cout << sizeof(arr) << std::endl; // 20
 foo(arr);
  return 0;
```

[[[](C struct)

```
struct Student {
    int id;
    char name[20];
    float score;
};
struct Class {
    int id;
    Student students[30];
};
struct Student s = {1, "Alice", 90.5};
struct Student t = {.id = 1, .name = "Alice", .score = 90.5};
struct Class c = {1, {s, t}};
// access struct member
std::cout << s.id << s.student[0].name << std::endl;</pre>
```

```
Union U {
    int a;
    double b;
};

size_t size = sizeof(U); // 8

U u;
u.a = 10;
std::cout << u.a << std::endl; // 10
u.b = 3.14;
std::cout << u.b << std::endl; // 3.14</pre>
```

enum([]])

```
#include <iostream>
int main() {
  enum Color {
      RED,
      GREEN,
      BI UF
  };
  enum Weather {
      SUNNY = 10,
      RAINY = 20,
     CLOUDY = 30
  };
  Color c = RED;
  Weather w = SUNNY;
  std::cout << c << '\n' << w << std::endl;
```

Control Flow



if 🔲

```
if (condition) {
    // code block
} else if (condition) {
    // code block
} else {
    // code block
}
```

switch 🔲

```
switch (expression) {
    case constant1:
        // code block
        break;
    case constant2:
        // code block
        break;
    default:
        // code block
}
```



```
while (condition)
  // statement

do {
    // code block
} while (condition);
```

for 🔲

```
for (int i = 0; i < 10; i++)
   // statement

int b = 10;
for (; b > 0; b--)
   // statement
```

Statement

```
0000000 C++ 0000,0000000

1. 0000

2. 00000

3. 0000

4. 0000

5. 0000

6. 0000

7. 0000

8. try 0
```

```
// {} ????????
{
   int a = 1;
   std::cout << a << std::endl;
}</pre>
```