

# C++ vs. C and Java

- Version 1: Dr. Ofir Pele
- Version 2: Dr. Miri Ben-Nissan
- Version 3: Dr. Erel Segal-Halevi

<b><i>C++ = C + Java + More...</i></b>	<b>C</b>	<b>C++</b>	<b>Java</b>
<b>Low-level machine programming</b>	Yes	Yes	No
<b>High-level OO programming</b>	No	Yes	Yes
<b>Generic template programming</b>	No	Yes	Limit-ed
<b>Complexity</b>	Low	High	Med.

# Why use C++?

- Combine high-level abstractions with low-level time and memory control.
- Understand the machine.
- Practice learning a complex language.

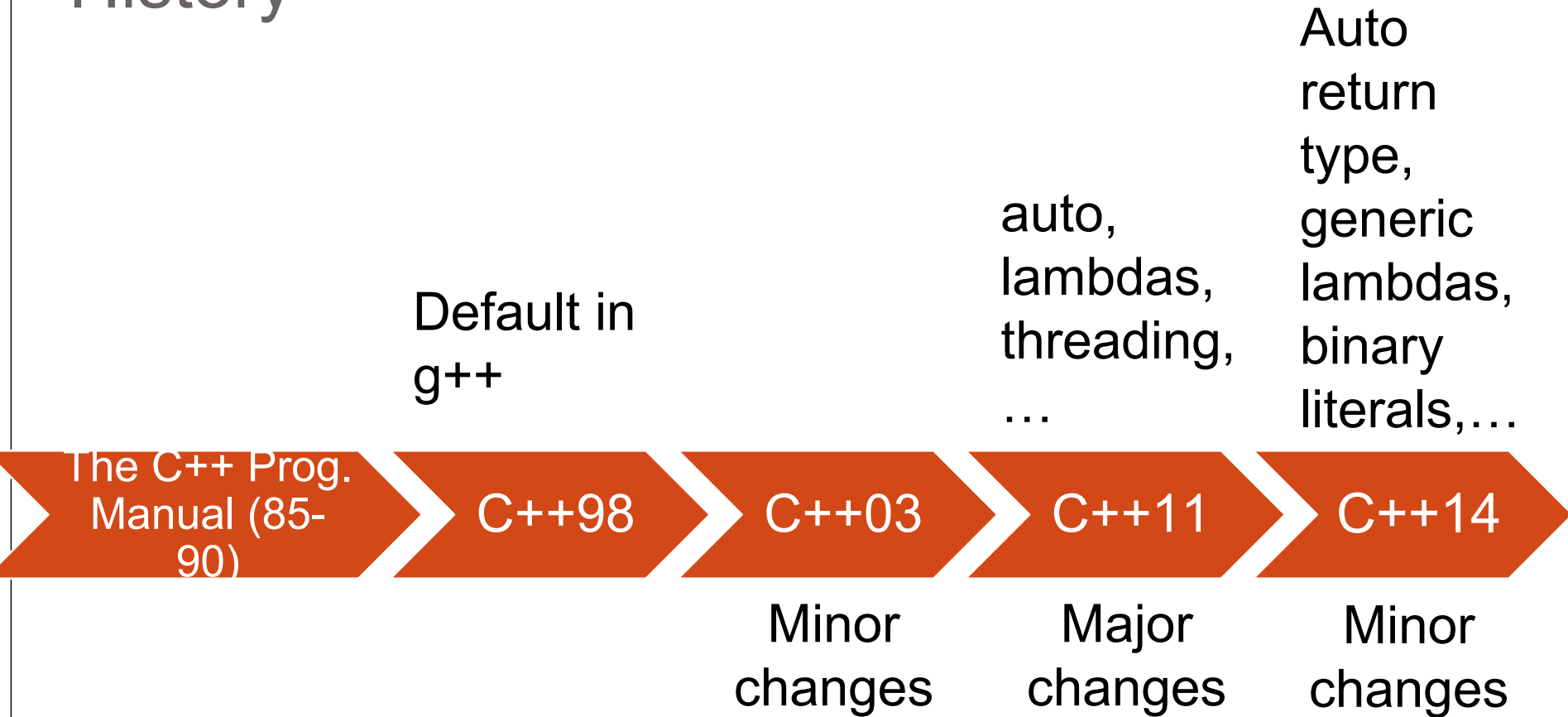
## Some software written in C++:

- **Facebook:** <https://github.com/facebook/folly>
- **Bitcoin:** <https://github.com/bitcoin/bitcoin>
- **LibreOffice:** <https://github.com/LibreOffice/core>
- **Unreal:** <https://github.com/EpicGames/UnrealEngine>
- **TensorFlow:** [https://github.com/tensorflow/tensorflow\(+python\)](https://github.com/tensorflow/tensorflow(+python))

# C++ vs. Java – memory

- In C++, the memory consumption of a data structure is tight – you get only what you ask for.
- In Java, your data structures might consume much more memory.
- See example in folder 2.
  - *Why is this?*

# History



**We'll learn parts of C++-11, 14, 17,  
Mostly parts that makes C++ more “pythonic” while keeping it  
efficient**

# Future



C++17

C++20

...

# The missing types

# strings in C++

```
#include <iostream>
#include <string>
int main()
{
    std::string str;
    int a;
    double b;
    std::cin >> str >> a >> b;
    if(std::cin.fail())
    {
        std::cerr << "input problem\n";
        return 1;
    }
    std::cout << "I got: " << str << ' '
    << a << ' ' << b << std::endl;
}
```

:More about string functions

<http://www.cppreference.com/cppstring>



# Boolean variables

```
#include <iostream>
int main()
{
    int a = 5;
    bool isZero = (a == 0);
    // same conditions
    if(!isZero && isZero==false &&
    isZero!=true && !!! isZero && a )
    {
        std::cout << "a is not zero\n";
    }
}
```

Good  
style



# C++-11 namespace (folder 3)

- Groups different variables and functions together;
- Reduces danger of name-collision when including different libraries;
- Can span multiple files.
- Standard library namespace: `std`;
- Another example: `folly`  
[https://github.com/facebook/folly/blob/master/folly/stop\\_watch.h](https://github.com/facebook/folly/blob/master/folly/stop_watch.h)
- Standard library namespace: `std`;

# C++-11 enum class (folder 4)

```
enum class Season : char {  
    WINTER, // = 0 by default  
    SPRING, // = WINTER + 1  
    SUMMER, // = WINTER + 2  
    AUTUMN  // = WINTER + 3  
};  
  
Season curr_season;  
curr_season= Season::AUTUMN;  
curr_season= SUMMER; // won't compile! (good)  
curr_season= 19; // won't compile! (good)  
int prev_season= (int)Season::SUMMER; // won't  
compile! (good)
```

## Error Handling in C++ *(folders 5-6)*

	<b>Exception</b>	<b>Assert</b>
<b>Used during:</b>	Normal run	Development
<b>Used for:</b>	Handling exceptional conditions.	Spotting internal errors and bugs.
<b>Disabling:</b>	No	With compiler flag