Programming in Modern C++

Module M01: Course Overview

Partha Pratim Das

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ac.in

All url's in this module have been accessed in September, 2021 and found to be functional

Module Objectives

- To understand the importance and ease of C++ in programming
- To Know Your Course including objective, prerequisites, outline, evaluation, books, and tools

Module Outline

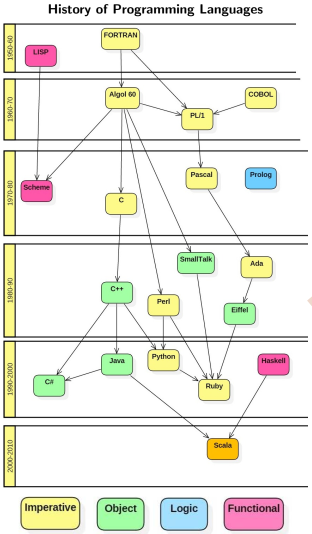
Know Your C/C++

Know Your C/C++

Source:

- Do any companies still use C++?, quora, 2019

History of Programming Languages

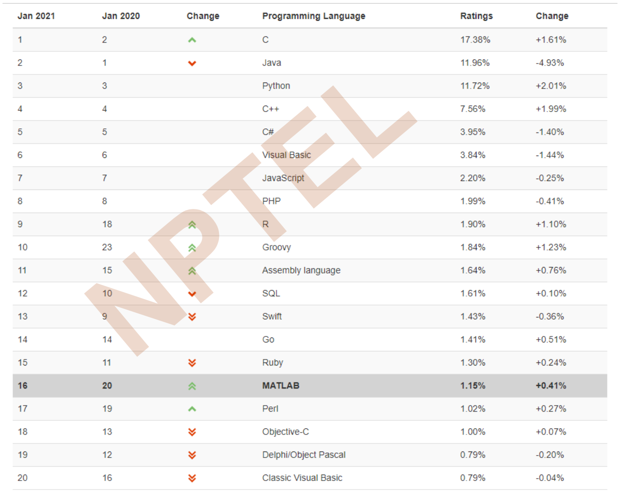


Paradigms: *Imperative*: Algorithms + Data, *Object*: Data, *Logic*: Facts + Rules + Queries, and *Functional*: Functions

- **FORTRAN**: IBM
- **LISP**: John McCarthy
- **Algol 60**: John Backus & Peter Naur
- **COBOL**: Grace Murray Hopper
- **PASCAL**: Niklaus Emil Wirth
- **Prolog**: Alain Colmerauer & Philippe Roussel
- **Scheme**: Guy L. Steele & Gerald Jay Sussman
- **C**: Brian W. Kernighan & Dennis M. Ritchie
- **SmallTalk**: Alan Kay, Dan Ingalls, & Adele Goldberg
- **Ada**: Jean Ichbiah & Tucker Taft
- **C++**: Bjarne Stroustrup
- **Objective-C**: Brad Cox
- **Perl**: Larry Wall
- **Java**: James Gosling
- **Python**: Guido van Rossum
- **Haskell**: Paul Hudak
- **C#**: Microsoft Corporation
- **Ruby**: Yukihiro Matsumoto
- **Scala**: Martin Odersky

Source: Programming Language Evolution

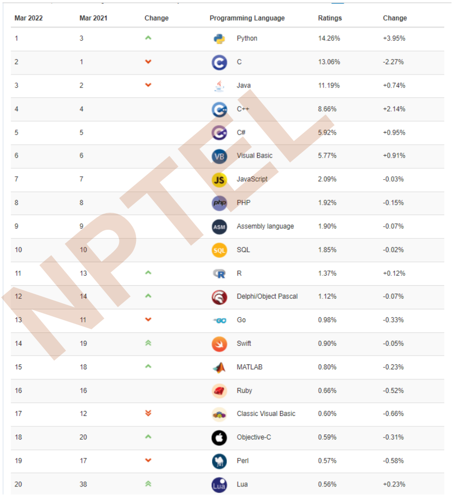
TIOBE Index of Programming Languages: January 2021



Jan 2021	Jan 2020	Change	Programming Language	Ratings	Change
1	2	▲	C	17.38%	+1.61%
2	1	▼	Java	11.96%	-4.93%
3	3		Python	11.72%	+2.01%
4	4		C++	7.56%	+1.99%
5	5		C#	3.95%	-1.40%
6	6		Visual Basic	3.84%	-1.44%
7	7		JavaScript	2.20%	-0.25%
8	8		PHP	1.99%	-0.41%
9	18	▲▲	R	1.90%	+1.10%
10	23	▲▲	Groovy	1.84%	+1.23%
11	15	▲▲	Assembly language	1.64%	+0.76%
12	10	▼	SQL	1.61%	+0.10%
13	9	▼▼	Swift	1.43%	-0.36%
14	14		Go	1.41%	+0.51%
15	11	▼▼	Ruby	1.30%	+0.24%
16	**20**	▲▲	**MATLAB**	**1.15%**	**+0.41%**
17	19	▲	Perl	1.02%	+0.27%
18	13	▼▼	Objective-C	1.00%	+0.07%
19	12	▼▼	Delphi/Object Pascal	0.79%	-0.20%
20	16	▼▼	Classic Visual Basic	0.79%	-0.04%

Top Programming Languages - IEEE Spectrum, index — TIOBE - The Software Quality Company

TIOBE Index of Programming Languages: Mar-22 Post Recording



Mar 2022	Mar 2021	Change	Programming Language	Ratings	Change
1	3	↑	Python	14.26%	+3.95%
2	1	↓	C	13.06%	-2.27%
3	2	↓	Java	11.19%	+0.74%
4	4		C++	8.66%	+2.14%
5	5		C#	5.92%	+0.95%
6	6		Visual Basic	5.77%	+0.91%
7	7		JavaScript	2.09%	-0.03%
8	8		PHP	1.92%	-0.15%
9	9		Assembly language	1.90%	-0.07%
10	10		SQL	1.85%	-0.02%
11	13	↑	R	1.37%	+0.12%
12	14	↑	Delphi/Object Pascal	1.12%	-0.07%
13	11	↓	Go	0.98%	-0.33%
14	19	⇈	Swift	0.90%	-0.05%
15	18	↑	MATLAB	0.80%	-0.23%
16	16		Ruby	0.66%	-0.52%
17	12	⇊	Classic Visual Basic	0.60%	-0.66%
18	20	↑	Objective-C	0.59%	-0.31%
19	17	↓	Perl	0.57%	-0.58%
20	38	⇈	Lua	0.56%	+0.23%

Top Programming Languages - IEEE Spectrum, index — TIOBE - The Software Quality Company. **Rating of C++ is up from 7.56% to 8.66%**

Choosing the Right Language

- Most systems need several languages for different parts of the system
  - HTML for front-end rendering and Javascript for active front-end logic
  - Java for servlet (business layer) and JSP for server-end embedding
  - SQL for data manipulation
- Nature of Application decides the choice of the language
  - Systems Programming ⇒ C++ (very high performance with complex behavior)
  - Embedded Programming ⇒ C (very high performance with frugal dev tools)
  - Application Programming ⇒ Java (medium performance with quick & robust app)
  - Web Programming ⇒ Python (low performance with portability)

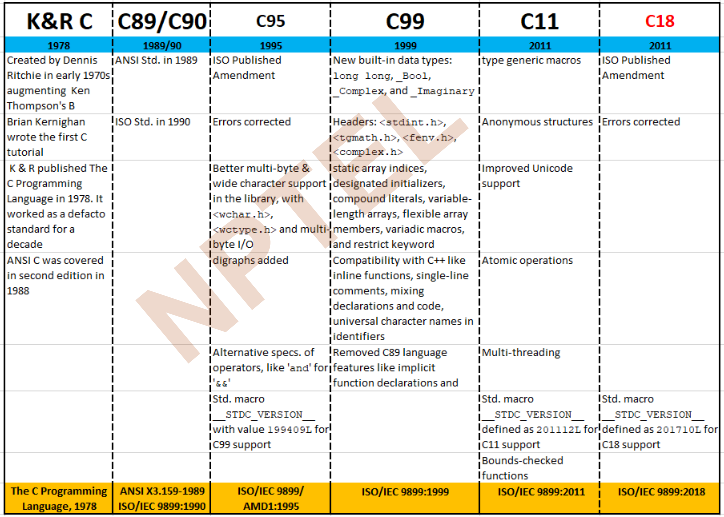
Source: Why Undergraduates Should Learn the Principles of Programming Languages?, ACM SIGPLAN, 2011

Why learn C/C++?

- C++ is used in development of **Core Software**
  - *Databases*: Oracle, MySQL, MongoDB, MemSQL, etc. used for YouTube, Twitter, Facebook, etc.
  - *OS*: Windows, Linux, Android, Ubuntu, iOS, etc. are written in a combination of C and C++
  - *Compilers / VMs / Tools*: GNU Compiler Collection (GCC); JVM, PVM; MATLAB, IDE
  - *Web Browsers*: Chrome, Firefox, Safari, etc.
  - *Graphic Engine*: Applications in image processing, computer vision, screen recorders, games etc.
  - *Embedded Systems*: Smart watches, MP3 players, GPS systems, etc.
- C++ has **Core Strengths** like
  - *Fast*, *Portable*, and *Scalable*
  - Offers multiple levels of *Abstraction*: hardware to objects to meta-programs
  - *Multi-Paradigms*: Imperative / Procedural (C / Python), Object-Oriented (Algol / Java), Functional (LISP), Generic / Meta-Programming (template, lambda), Concurrent (Java)
- C++ has a **Large Community**
- C++ has **Abundant Library Support (STL)**
- C++ skills attract **High Salary**
- **Caveat**
  - It takes more time to be skilled in C++ compared to, say, Python due to its complexity and diversity
  - It is better to use Java / Python for simple front-end applications that are not performance critical
  - C++ is not best suited for front-end graphics applications for the lack of graphics library

Source: Top 10 Reasons to Learn C++, GeeksforGeeks, 2019

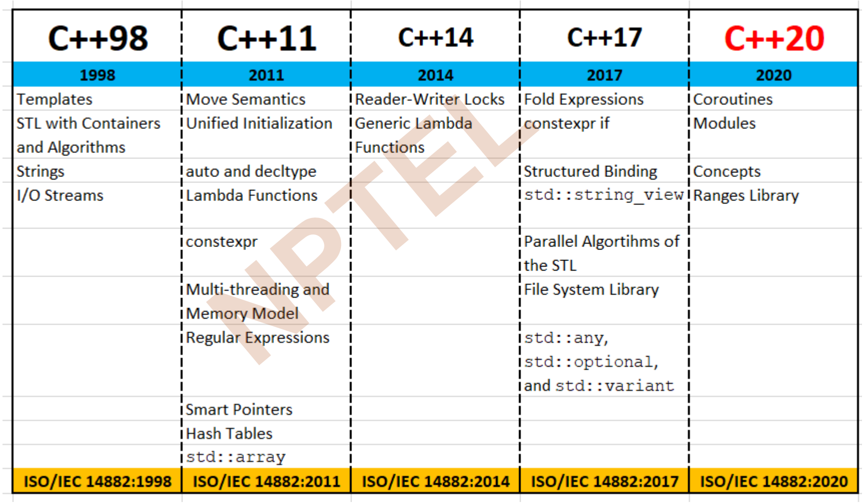
C Standards



K&R C	C89/C90	C95	C99	C11	C18
1978	1989/90	1995	1999	2011	2011
Created by Dennis Ritchie in early 1970s augmenting Ken Thompson's B	ANSI Std. in 1989	ISO Published Amendment	New built-in data types: `long long`, `_Bool`, `_Complex`, and `_Imaginary`	type generic macros	ISO Published Amendment
Brian Kernighan wrote the first C tutorial	ISO Std. in 1990	Errors corrected	Headers: `<stdint.h>`, `<tgmath.h>`, `<fenv.h>`, `<complex.h>`	Anonymous structures	Errors corrected
K & R published The C Programming Language in 1978. It worked as a defacto standard for a decade		Better multi-byte & wide character support in the library, with `<wchar.h>`, `<wctype.h>` and multi-byte I/O	static array indices, designated initializers, compound literals, variable-length arrays, flexible array members, variadic macros, and restrict keyword	Improved Unicode support	
ANSI C was covered in second edition in 1988		digraphs added	Compatibility with C++ like inline functions, single-line comments, mixing declarations and code, universal character names in identifiers	Atomic operations	
		Alternative specs. of operators, like 'and' for '&&'	Removed C89 language features like implicit function declarations and	Multi-threading	
		Std. macro `__STDC_VERSION__` with value `199409L` for C99 support		Std. macro `__STDC_VERSION__` defined as `201112L` for C11 support	Std. macro `__STDC_VERSION__` defined as `201710L` for C18 support
				Bounds-checked functions	
The C Programming Language, 1978	ANSI X3.159-1989 ISO/IEC 9899:1990	ISO/IEC 9899/ AMD1:1995	ISO/IEC 9899:1999	ISO/IEC 9899:2011	ISO/IEC 9899:2018

Latest Version as of Sep-21: C18: ISO/IEC 9899:2018, 2018

C++ Standards



C++98	C++11	C++14	C++17	C++20
1998	2011	2014	2017	2020
Templates	Move Semantics	Reader-Writer Locks	Fold Expressions	Coroutines
STL with Containers and Algorithms	Unified Initialization	Generic Lambda Functions	constexpr if	Modules
Strings	auto and decltype		Structured Binding	Concepts
I/O Streams	Lambda Functions		`std::string_view`	Ranges Library
	constexpr		Parallel Algortihms of the STL	
	Multi-threading and Memory Model		File System Library	
	Regular Expressions		`std::any`, `std::optional`, and `std::variant`	
	Smart Pointers			
	Hash Tables			
	`std::array`			
ISO/IEC 14882:1998	**ISO/IEC 14882:2011**	**ISO/IEC 14882:2014**	**ISO/IEC 14882:2017**	**ISO/IEC 14882:2020**

Fixes on C++98: C++03: ISO/IEC 14882:2003, 2003
Latest Version as of Sep-21: C++20: ISO/IEC 14882:2020, 2020

Know Your Course

Know Your Course

Course Objectives

- Learn to develop software using C++ (C++98/03)
  - Features of C++ over and above C
  - Object-Oriented Paradigm in C++
  - STL for extensive code reuse
- Learn to improve software development using modern C++ (C++11)
  - Features of C++11 over and above C++98/03
  - Concurrent Programming in C++
  - Better quality and efficiency by C++11
- Cultivate skills to design, code, debug, and test software written in C++
- Attain strong employability with hands-on skills of software development

Course Prerequisites

Data Structures

- Array
- List
- Binary Search Tree
  - Balanced Tree
- B-Tree
- Hash Table / Map

Algorithms & Programming in C

- Sorting
  - Merge Sort
  - Quick Sort
- Search
  - Linear Search
  - Binary Search
  - Interpolation Search

Object-Oriented Analysis and Design

NPTEL Courses

- Design and Analysis of Algorithms
- Introduction to Programming in C
- Object-Oriented Analysis and Design

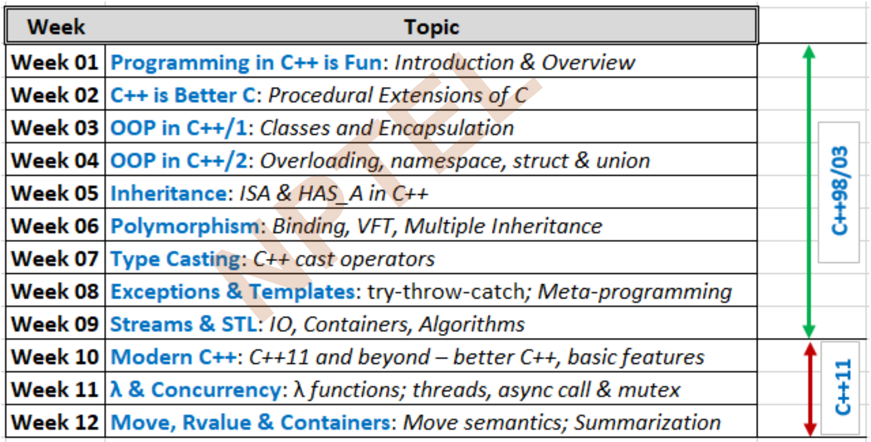
Quick Recap Modules

- Two self-study modules (QR1 & QR2) are provided for quick recap in Week 0
- Recap would be necessary before moving on to Module 02

Course Outline

- The course comprises:
  - 60 **Modules** (5 modules / week for 12 weeks). These are numbered serially as `Mnn`
    - ▷ These cover the course syllabus
    - ▷ These are used in assignments and examinations
  - Supplementary **Quick Recap** modules to revise C language and related topics in Week 0. These are numbered serially as `QRn`
    - ▷ These may be used to recapitulate C programming, as needed
    - ▷ These are not directly part of the syllabus, but cover the prerequisites. So their understanding are critical for the main modules. Those who know, may skip
  - **Tutorials** to build skills in C / C++ programming. These are numbered serially as `Tnn`
    - ▷ Some tutorials are of *Complementary* nature. These talk about various aspects of program development, program building, programming practices, etc. that may help to develop software using C / C++
    - ▷ Remaining tutorials are of *Supplementary* nature. These talk about additional information about C / C++ like how to mix these language, what is their compatibility etc.
    - ▷ Tutorials are not part of the syllabus. These are included for developing allround skills for those who desire so

Course Outline: Modules



Week	Topic	
Week 01	**Programming in C++ is Fun**: *Introduction & Overview*	C++98/03
Week 02	**C++ is Better C**: *Procedural Extensions of C*	
Week 03	**OOP in C++/1**: *Classes and Encapsulation*	
Week 04	**OOP in C++/2**: *Overloading, namespace, struct & union*	
Week 05	**Inheritance**: *ISA & HAS_A in C++*	
Week 06	**Polymorphism**: *Binding, VFT, Multiple Inheritance*	
Week 07	**Type Casting**: *C++ cast operators*	
Week 08	**Exceptions & Templates**: try-throw-catch; *Meta-programming*	
Week 09	**Streams & STL**: *IO, Containers, Algorithms*	
Week 10	**Modern C++**: *C++11 and beyond – better C++, basic features*	C++11
Week 11	**λ & Concurrency**: *λ functions; threads, async call & mutex*	
Week 12	**Move, Rvalue & Containers**: *Move semantics; Summarization*	

Course Outline: Tutorials

- **Tutorials** are complementary or supplementary:
  - *Complementary* Tutorials introduce new ideas and skill areas to complement the understanding of the C/C++ languages. These include:
    - ▷ How to build a C/C++ program and / or static and dynamic libraries?
    - ▷ How to automate build using `make` utility?
    - ▷ What tools may be used to design, develop, test, and manage C / C++ software?
    - ▷ How to reuse?
      - — binary (static or dynamic library)
      - — code (template and meta-programming)
      - — design (desing pattern)
    - ▷ and more
  - *Supplementary* Tutorials provide additional information and insight to supplement the understanding of the C/C++ languages. These include:
    - ▷ How to mix C/C++ in a single program?
    - ▷ What is the compatibility of C/C++?
    - ▷ What are the coding styles to write good C/C++ programs?
    - ▷ and more

Course Evaluation

- **Assignments:** Once every week
  - Quiz Assignments
  - Programming Assignments
  - Weekly Assignment Score = Quiz Assignment Score + Programming Assignment score
  - Best 9 assignment scores (out of 12) to be considered for certification criteria
- **Unproctored Test:** 20 Marks
  - Type of questions: Programming. Very similar to the Programming assignments
  - You can appear the test from your home/college/work place itself using your PC (It may not support the mobile)
- **Proctored Test:** 80 Marks
  - Type of the questions: MCQ, MSQ, and short answer (SA) or one word type.
  - You need to visit the allocated exam center for this test
  - Online test (Computer based)
- **Certification Criteria**
  - All the scores are scaled to 100
  - Assignment score >= 40/100 AND Unproctored test score >= 40/100 AND Proctored test score >= 40/100

    (OR)

    Assignment score >= 10/25 AND Unproctored test score >= 10/25 AND Proctored test score >= 20/50
  - All the above three conditions have to be satisfied.
- *Note: NPTEL may change the certification criteria. However, You will get notified regarding the changes through an announcement prior to the tests. The evaluation process, marks distribution and certification criteria will be decided by the Instructor who runs the course in a specific semester.*

Textbooks,Tutorials, Standards, and Blogs

- **Textbooks**
  - The C Programming Language, Brian Kernighan and Dennis Ritchie, 1988 [Used here]
  - C programming: A Modern Approach, 2nd Ed., Kim N. King, 2008
  - C++ Primer, 5th Ed., S. Lippman, J. Lajoie, and B. Moo, 2012 [Most popular textbook]
  - Programming: Principles and Practice using C++, 2nd Ed., Bjarne Stroustrup, 2014 [Used here]
  - The C++ Programming Language, 4th Ed., Bjarne Stroustrup, 2013 [Authentic C++ Book]
- **Tutorials** [Free]
  - C Tutorial
  - Learn C and C++ Programming: C Tutorial [C], C++ Tutorial [C++]
  - LEARN C++: Skill up with our free tutorials [C++11, Used here]
- **Standards**
  - ISO C Standard: ISO/IEC 9899:2018 [Latest Standard]
  - ISO C++ Standards: ISO/IEC 14882:2020 [Latest Standard]
  - C++98 and C++03, C++11, C++14, C++17, C++20 [Free: Used here]
- **Blogs** [Free & Used here]
  - Bjarne Stroustrup: Creator of C++
  - Andrei Alexandrescu: Creator of D
  - Scott Meyers: Prolific educator of C++
  - Herb Sutter: Sutter's Mill: Chair of ISO C++ standards committee for over a decade

References

- **C++98/03**
  - Effective C++, 3rd Ed., 2005 and More Effective C++, 1st Ed., 1996, Scott Meyers [Used here]
  - Modern C++ Design, Andrei Alexandrescu, 2001 [Used here]
  - Exceptional C++, 1999 and More Exceptional C++, 2001 by Herb Sutter
  - Effective STL, 1st Ed., Scott Meyers, 2001
  - C++ Coding Standards, 1st Ed., Herb Sutter and Andrei Alexandrescu, 2004 [Used here]
  - The D Programming Language, Andrei Alexandrescu, 2010 [Future of C Family?]
  - Google C++ Style Guide
- **C++11, ...**
  - Effective Modern C++, Scott Meyers, 2015 [Used here]
  - Overview of the New C++ (C++11/14), Scott Meyers, 2015 [Used here]
  - C++ Move Semantics - The Complete Guide, Nicolai M. Josuttis, 2020
  - C++ Concurrency in Action, 2nd Ed., Anthony Williams, 2019
  - C++17 - The Complete Guide, Nicolai M. Josuttis, 2020
  - C++17 In Detail, Bartlomiej Filipek, 2019
  - Professional C++, 4th Ed., Marc Gregoire, 2018
  - Functional Programming in C++, Ivan Čukić, 2018
  - C++ Templates, 2nd Ed., D. Vandevoorde, N. M. Josuttis, and D. Gregor, 2017
  - The C++ Standard Library: A Tutorial and Reference, 2nd Ed., Nicolai M. Josuttis, 2012

Tools

- MinGW - Minimalist GNU for Windows [Free & Downloadable. Used here]
  - A native Windows port of the GNU Compiler Collection (GCC), with freely distributable import libraries and header files for building native Windows applications
  - Use GDB: The GNU Project Debugger for code debugging
  - Check How to install gdb in windows 10 to install minGW and gdb for Windows together
- GNU Online Compiler [Free & Online]
  - From Language Drop-down, choose C (C99), C++ (C++11), C++14, C++17
  - To mark the language for gcc compilation, set `-std=<compiler_tag>`
    - Tags for C are: `c89`, `c90`, `c11`, `c17`, `c18`, etc. Further `-ansi` means `-std=c90`
    - Tags for C++ are: `c++98`, `c++03`, `c++11`, `c++14`, `c++17`, `c++20`, etc. Further `-ansi` means `-std=c++98`
    - Check 3.4 Options Controlling C Dialect and 2 Language Standards Supported by GCC for details and options
- Code::Blocks [Free & Online]
  - A free, open source cross-platform IDE that supports GCC, Clang, Visual C++, and others
  - Choose language flag based on the choice of compiler (check on the manual)
- Programiz Online Compiler [Free & Online]
  - Supports C18 and C++14
- OneCompiler [Free & Online]
  - Supports C11 and C++14
- *While using a compiler, make sure that you know the language version you are compiling for*

Tools: Checking Compiler Version

- Check __cplusplus macro in C++:

```
#include <iostream>
#include <typeinfo>
int main() {
    if (__cplusplus == 201703L) std::cout << "C++17\n";
    else if (__cplusplus == 201402L) std::cout << "C++14\n";
    else if (__cplusplus == 201103L) std::cout << "C++11\n";
    else if (__cplusplus == 199711L) std::cout << "C++98\n";
    else std::cout << "pre-standard C++\n";
}
```

- Check __STDC_VERSION__ macro in C:

```
#include <stdio.h>
int main() {
    if (__STDC_VERSION__ == 201710L) printf("C18\n");       // C11 with bug fixes
    else if (__STDC_VERSION__ == 201112L) printf("C11\n");
    else if (__STDC_VERSION__ == 199901L) printf("C99\n");
    else if (__STDC_VERSION__ == 199409L) printf("C89\n");
    else printf("pre-standard C\n");
}
```

Source: 3.7.1 Standard Predefined Macros

Module Summary

- Understood the importance and ease of C++ in programming
- Learnt about the course - objective, prerequisites, outline, evaluation, books, and tools