



Department of Engineering
IT Services

C++

C++ is the object-oriented development of C with classes, member functions, operator overloading, constructors etc. It's the first language taught to undergraduates at CUED.



Starting C++

Local teaching

- [2015-16 1A C++ Computing](#)
- [2016-17 1B C++ Computing](#)
- [Tutorial Guide to C++ Programming](#) (for local 1A/1B students)
- CUED's C++ data book
- [Frequently asked C++ questions](#) (CUED)
- [A handout on Debugging](#)
- [Basic optimising and profiling](#) (CUED)
- [More C++](#) (updated in 2016) ([notes](#)).

General Reference

- The list of [Frequently Asked Questions](#) is invaluable, but isn't designed for complete beginners.

Tutorials

- [Learn C++ Programming](#) (programiz)
- [The cplusplus.com tutorial](#)
- [LearnCpp tutorial](#)
- [FunctionX C++ tutorials](#)
- [go4expert's C++ forum](#)

Special topics

- [cplusplus.com's iostream library reference](#)
- [An Introduction to the Standard Template Library \(STL\)](#) (Carlos Moreno)
- [yolinux STL tutorial](#)

Advanced WWW resources

General

- The [Lysator](#) list of C-related docs
- [The C++ Source](#) (Journal)
- [C/C++ Reference](#)
- [C++ resources network](#)
- [C++ Internet site list](#) (Robert Davies)
- Scott Meyers' [C++ articles](#)
- [C++ in the Real World: Advice from the Trenches](#)
- [Association of C & C++ Users](#) (see their [book reviews](#), etc.)
- [Bjarne Stroustrup's homepage](#) (papers, [glossary](#) and FAQs).
- [Guru of the Week](#)

Maths

- [Techniques for Scientific C++](#) (Todd Veldhuizen)
- [Available C++ Libraries FAQ](#)
- [Template Numerical Toolkit](#)
- [Graph Template Library](#)
- [Blitz++](#) (library for scientific computing)
- [What Every Computer Scientist Should Know About Floating-Point Arithmetic](#)
- [Object-Oriented Numerics Page](#) (libraries, articles, etc).
- [Netlib's C++ material](#)

Pointers

- [Pointers and references](#)
- [Pointers](#) (by Todd Gibson)
- [A Beginner's Guide to Pointers](#) (by Andrew Peace)
- [Smart Pointers - What, Why, Which?](#) (by Yonat Sharon)

Programming

- [Function Pointer Tutorials](#)
- [How to interpret complex C/C++ declarations](#) (by Vikram A Punathambekar)
- [C++ Portable Components](#)
- [C++ idioms](#)
- [SNIPPETS](#)
- the busy [comp.lang.c++.](#) and [comp.lang.c++.moderated](#) newsgroups.
- [Stackoverflow](#)

Other Topics

- [CERT C++ Secure Coding Standard](#)
- [MLC++](#) (Machine Learning Library)
- [Boost.org](#) - free, peer-reviewed, C++ libraries
- [Technical Report on C++ Performance](#) (ISO/IEC)

NEW

Local Information

General

- [C++ for C programmers](#)
- [A look at some of the newer C++ features](#)
- [C++ Notes for intermediate students](#) (especially CUED IIA students)
- [C++ and the Standard Library](#)
- [C++ course](#) (a crash course for CUED 3rd years and upward)
- [C++ objects, containers, complex numbers and maps](#)

Special topics

Types

- [C++ and simple type conversion](#)
- [Casting in C++](#) (intermediate level)
- [C++ type anomalies and special cases](#) (intermediate level)
- [typedef](#)

Variables

- [Passing values in C++](#) (the use of const)
- [C++ and static](#)
- [C++ and global variables](#)
- [C++ and enumeration](#)

Classes

- [C++ - a public/private issue](#)
- [C++ constructors](#)
- [C++ destructors](#)
- [How not to overload C++ operators](#)
- [C++: Visibility and Look-up](#) (intermediate level)
- [Virtuals and overloading](#) (intermediate level)
- [C++ Inheritance, Protection and Friends](#) (intermediate level)
- [C++ and Building Classes](#) (a vector arithmetic example)
- [C++ and Building Classes](#) (a limited-range integer example)

Generic programming

- [C++ vector memory and 2D vectors](#)
- [C++ vectors and copy/move constructors](#)
- [Bridge: A C++ programming Exercise](#) (using vectors)
- [Trees, Graphs and C++](#)
- [bind1st and bind2nd](#) (intermediate level)
- [mem_fun](#) (intermediate level)

Performance

- [Faster C++](#)
- [C++: Nasty Tricks](#)

Misc

- [C++11](#)

- [C++ "for" loop teaching aid](#)
- [C++ "function" teaching aid](#)
- [C++ I/O](#)
- [sizeof](#)
- [Complex numbers](#)
- [Templates](#)
- [C++ Templates and Friends \(intermediate level\)](#)
- [Namespaces](#)
- [C++ Smart Pointers \(intermediate level\)](#)
- [Mixing Languages on linux](#)
- [C++ and the main function](#)

Compilers and libraries

We have various [IDEs](#) (Integrated Development Environments) providing easy access to the compiler (which is g++ on the linux machines). [Geany](#) is used in the 1st year course. If you want to work away from CUED see

- [Remote 1AC++ Computing](#)
- [Installing C++ compilers](#)
- [Installing C++ graphics libraries](#)
- [Remote access to the Engineering Department machines](#)
- [g++ documentation](#)
- [g++ standard library documentation](#)

To use C++2011 on the central linux system, type

```
g++5 --std=c++11
```

With that version, the following should compile

```
#include <iostream>
#include <string>
#include <regex>

int main()
{
    std::string str("1231");
    std::regex r("^(\\d)\\d"); // entire match will be 2 numbers
    std::smatch m;
    std::regex_search(str, m, r);
    for(auto v: m) std::cout << v << std::endl;
}
```

When run, it should produce this output

```
12
1
```

Integrated Development Environments

An Integrated Development Environment (IDE) offers integrated editing, compiling, debugging and project-management facilities. On the teaching system we currently

have *Anjuta (version 1)*



, *Geany*



, *NetBeans*

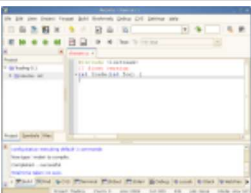
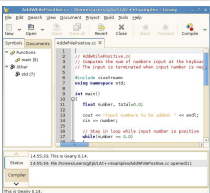
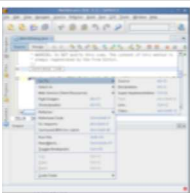
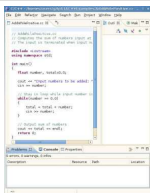
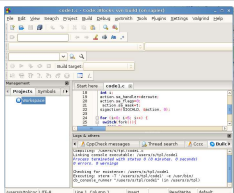


(with a C++ add-on), *Eclipse*



and *codeblocks*



				
Anjuta 1. Click to zoom in	Geany. Click to zoom in	Netbeans. Click to zoom in	Eclipse. Click to zoom in	Codeblocks. Click to zoom in

FAQ

- [CUED C++ Frequently Asked Questions](#)

© Cambridge University, Engineering Department, Trumpington Street, Cambridge CB2 1PZ, UK ([map](#))

Tel: +44 1223 332600, Fax: +44 1223 332662

[Privacy policy](#)

Contact: [tl136](#)