

C++ Overview

David L. Levine
Christopher D. Gill
Department of Computer Science
Washington University, St. Louis
levine,cdgill@cs.wustl.edu

<http://classes.cec.wustl.edu/~cs342/>

C++ Overview

- What is C++?
- Origination and Evolution of C++
- Why Use C++?
- How Does C++ Differ from Java?
- C++ and Java Minimal Examples
- Compiling C++

What is C++?

C++ is a general purpose programming language designed to make programming more enjoyable for the serious programmer.

–Bjarne Stroustrup, *The C++ Programming Language, First Edition*

What is C++ (cont'd)?

- Based on C
 - Supports procedural programming paradigm
 - Can link with compiled C code (and libraries)
 - Portable (using preprocessor)
- Adds polymorphism
 - Run-time (dynamic) binding of function calls
- Adds inheritance
 - Reuse interfaces
 - Reuse implementations

What is C++ (cont'd)?

- Adds generic code (template class) support
- Adds exception handling
- Supports large-scale programming
 - Separate compilation
 - Namespaces
 - Libraries (archives)

Origination of C++

- Designed in early 1980's by Bjarne Stroustrup of Bell Labs
- Backward compatible with C, as much as possible
 - First “compiler”, *cfront*, actually translated C++ to C
- Improvements over C
 - Stronger typechecking
 - Supports data abstraction
 - Supports object-oriented programming
 - Supports generic programming

Evolution of C++

- Added namespaces, exception handling, run-time type identification (RTTI), improved templates, *etc.*
- Improved compilers
- Added Standard Template Library (STL) containers and algorithms
- Standardized by ANSI, DIN, BSI, ISO (ISO/IEC 14882)

Why Use C++?

- To maximize execution speed
- To support reuse, with separation of interface and implementation
- To support data abstraction and dynamic binding
- For portability
- For backward source compatibility with C
- For link compatibility with C, Basic, Fortran, Ada, *etc.*
- To maximize execution speed

How Does C++ Differ from Java?

- C++ programs run standalone; the Java interpreter loads and runs any class with a `main ()` method
- Can separate C++ class interface (header) from implementation (definitions)
- C++ allows multiple inheritance of implementations
- C++ supports generic programming with template classes
- C+ memory must be managed by programmer; it does not provide built-in garbage collection like Java
 - C++ pointer variables access memory
- C++ passes arguments by value, by default

How Does C++ Differ from Java? (cont'd)

- C++ arrays are not first class citizens
- C++ allows operator overloading
- C++ allows global variables, but they should be avoided
- C++ has a preprocessor; Java relies on the constrained language definition for portability
- Built-in C++ types are implementation dependent

C++ and Java Minimal Examples

```
#include <iostream.h>
int
main (int, char *[])
{
  cout << "Hello, world!" << endl;
  return 0;
}

public class Hello
{
  /**
   * Entry point.
   *
   * @param argv currently unused
   */
  public static void main (String[] argv)
  {
    System.out.println ("Hello, world!");
  }
}
```

Compiling C++

- Sun CC 4.2
 - `pkgadd sc` (on CS machines, `pkgadd sc_4.0`)
 - `CC +w -g -o main main.cpp`
- GNU g++
 - `pkgaddperm egcs`
 - `g++ -Wall -W -g -o main main.cpp`
- `-g` option inserts debugging symbols
 - use `strip` to remove
- Without `-o` option, executable will be named `a.out`
- For more information: `man CC`, `info g++`, `info gcc`