Part VI

Important libraries

Important libraries

Don't reinvent the wheel

- It may be interesting to code your own linear algebra solver (say), but is it worth your time?
- There are some good scientific libraries out there.
- The nice thing is, they needn't be c++ libraries, as you can use c libraries in c++.
- Even for basic functionality, there are libraries.

SciNet compute • calcul CANADA

STL: Standard Template Library

Offers a lot of basic functionality

- Supplies a lot of data types and containers (templated).
- Often presented as part and parcel of the C++ language itself.
- Also contains a number of algorithms for e.g., sorting, finding
- Efficiency implementation dependent, and generally not great.

Some of the STL data types

`vector`	Relocating, expandable array
`list`	Doubly linked list
`deque`	Like vector, but easy to put something at beginning
`map`	Associates keys with elements
`set`	Only keys
`stack`	LIFO
`queue`	FIFO
`...`	

STL: Standard Template Library

Example

```
#include "iostream"
#include "vector"
class Grape {
   public:
      int nseeds;
};
int main() {
   using namespace std;
   Grape grapes[10];
   vector<Grape> bunch(grapes,grapes+9);
   bunch.push_back(grapes[9]);
   for (int i=0; i<bunch.size(); i++)
      cout << bunch[i].nseeds << endl;
   vector<Grape>::iterator i;
   for (i=bunch.begin(); i!=bunch.end(); i++)
      cout << (*i).nseeds << endl;
}
```

STL: Standard Template Library

Gotcha: Performance

- The purpose of the STL is not to provide a high performance library, i.e., runtime speed is not the objective.
- Rather it aims to have flexible containers with a uniform usage pattern.
- As a result, using e.g. an **`std::vector`** in an inner loop of you computation, instead of a simple array, can substantially slow down your code (even with the improvements in the implementation since the early days).
- The STL still does not have higher dimensional arrays, and the last thing you want is to have vectors of vectors.

SciNet compute • calcul CANADA

Other useful (scientific) libraries

library	functionality	C++	parallel
MPI	distributed parallel program	✓	✓
OpenMP	shared memory parallelism	✓	✓
Blas/Lapack	linear algebra (in MKL, ESSL)	×	✓ ×
Petsc	matrices, vectors, linear solvers	×	✓
GSL	numerical library	×	×
Boost	continues where STL left off	✓	×
	(+math, statistics, random, blas)		Thread&MPI
IT++	templated matrix implementations	✓	×
Blitz++	(not exhaustive)	✓	×
Armadillo		✓	✓ ×
POOMA		✓	✓
Eigen		✓	×
...			

Again: Don't reinvent the wheel!

Part VII

Further reading

Not covered so we could get to the heart of the matter:

Basic stuff (you'll want to learn these)

- Const correctness
- Booleans
- Inline functions
- Preprocessor
- New names for c header files
- Default parameters

Advanced material

- Initializer lists
- Static class members and enums
- Advanced template parameters
- Abstract base classes
- Multiple inheritance
- Exceptions

SciNet compute • calcul CANADA

Books and links

Books

- *C++ Interactive Course,* Lafore, Waite Group '96
- *C++ FAQs,* Cline, Lomow & Girou, Addison-Wesley '99
- *The C++ Programming Language,* Stroustup, Addison-Wesley '00
- *C+ Templates* Vandervoorde & Josuttis, Addison-Wesley '03
- *Effective C++,* Meyers, Addison-Wesley '03 Addison-Wesley,

Online

- *C++ FAQ,* www.parashift.com/c++-faq-lite
- *C++ Annotations,* www.icce.rug.nl/documents/cplusplus
- *C++ Reference,* www.cplusplus.com/reference

Google is your friend!