Software libraries

Victor Eijkhout, Susan Lindsey

COE 322 Fall 2021

1. Don't reinvent the wheel: use a library

Many things you want to program, have been thought of before:
see if there is a library for it.

Library: 'program without main':
you supply the main, functionality comes from library

2. External libraries

Don't reinvent the wheel: someone may have written stuff that you can use.

```
#include "fancylib.h"
```

Compilation

```
icpc -c yourprogram.cxx -I/usr/include/fancylib
icpc -o yourprogram yourprogram.o \
    -L/usr/lib/fancylib -lfancy
```

3. Where to find libraries

Search . . . There is a lot of stuff on github.

4. **Example:** `cxxopts`

`https://github.com/jarro2783/cxxopts`

Find the `2.2.1` release.
Use `wget` or `curl` to download straight to the class machine.
Unpack it.

5. Cmake based installation

The `cxxopts-2.2.1` directory has a file `CMakeLists.txt`

```
mkdir build
cd build
cmake -D CMAKE_INSTALL_PREFIX:PATH=${HOME}/mylibs \
    ..
make
make install
```

(This is an 'in-source' build. I don't like it: prefer to have the build directory elsewhere to keep the source untouched.)

6. Let's use this library

```
#include "cxxopts.hpp"
int main() {
  cxxopts::Options
    options("programname", "Program description");
}
```

compile

```
icpc -o program source.cpp \
   -I/path/to//cxxopts/installdir/include
```

Can you compile and run this?

7. Commandline options

```
options.add_options()
  ("h,help","usage information")
  ("n,nsize","size of the thing",
   cxxopts::value<int>()->default_value("4096"))
   // et cetera
  ;
auto result = options.parse(argc, argv);
if (result.count("help")>0) {
  std::cout << options.help() << std::endl;
  return 0;
}
int array_size = result["nsizes"].as<int>();
```