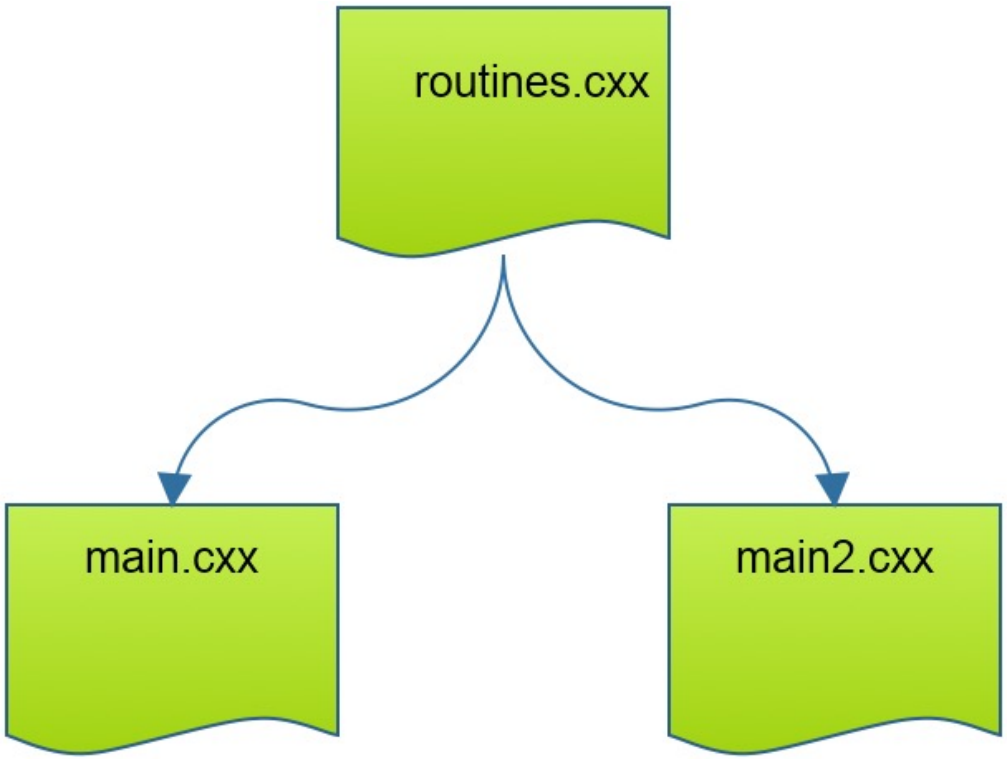
Prototypes

Victor Eijkhout, Susan Lindsey

COE 322 Fall 2021

1. Include files

Reuse code by include it in multiple mains.



We will develop a better scenario.

2. Prototypes and forward declarations, 1

A first use of prototypes is forward declaration.

Some people like defining functions after the main:

```
int f(int);
int main() {
  f(5);
};
int f(int i) {
  return i;
}
```

versus before:

```
int f(int i) {
  return i;
}
int main() {
  f(5);
};
```

3. Prototypes and forward declarations, 2

You also need forward declaration for mutually recursive functions:

```
int f(int);
int g(int i) { return f(i); }
int f(int i) { return g(i); }
```

4. Prototypes for separate compilation

```
// file: def.cxx
int tester(float x) {
  .....
}
```

```
// file : main.cxx
int tester(float);

int main() {
  int t = tester(...);
  return 0;
}
```

5. Compiling and linking

Your regular compile line

```
icpc -o yourprogram yourfile.cc
```

actually does two things: compilation, and linking. You can do those separately:

1. First you compile

   ```
   icpc -c yourfile.cc
   ```

   which gives you a file `yourfile.o`, a so-called object file; and
2. Then you use the compiler as linker to give you the executable file:

   ```
   icpc -o yourprogram yourfile.o
   ```

6. Dealing with multiple files

Compile each file separately, then link:

```
icpc -c mainfile.cc
icpc -c functionfile.cc
icpc -o yourprogram mainfile.o functionfile.o
```

7. Prototypes and header files

Header file contains only prototype:

```
// file: def.h
int tester(float);
```

The header file gets included both in the definitions file and the main program:

```
// file: def.cxx
#include "def.h"
int tester(float x) {
  .....
}
```

```
// file : main.cxx
#include "def.h"

int main() {
  int t = tester(...);
  return 0;
}
```

What happens if you leave out the `#include "def.h"` in both cases?

8. Class prototypes

Header file:

```
class something {
private:
  int i;
public:
  double dosomething( int i, char c );
};
```

Implementation file:

```
double something::dosomething( int i, char c ) {
  // do something with i,c
};
```

Review quiz 1

For each of the following answer: is this a valid function definition or function prototype.

- `int foo();`
- `int foo() {};`
- `int foo(int) {};`
- `int foo(int bar) {};`
- `int foo(int) { return 0; };`
- `int foo(int bar) { return 0; };`

9. Make

Good idea to learn the `Make` utility for project management.

(Also `Cmake`.)