EECS 2031

Software Tools

Module 11 – Makefiles

YORK UNIVERSITÉ UNIVERSITY

Modular compilation

- We’ve seen already that a C program can be split across multiple **source files**
- Each source file can be compiled separately into an **object file**
- The linker combines all object files (and any libraries used) into the final executable

Modular compilation

- If only some of the source files get modified, there is no need to re-compile unchanged source files
- The object file created during the last compilation is reused
- This can make the compilation process much faster for large systems
- See participation activity 12.5.2 in the textbook

make

- **make** is a project management tool that helps automating the compilation and linking process
- It utilizes a **makefile** that specifies a set of rules and commands
- Rules specify dependencies between a target file and its prerequisites
- See **makefile**

make

- When executed with no arguments, make will look for a file called **`makefile`** in the current directory and execute the first target
- Can execute other targets with

```
make aTarget
```

- Can specify a different filename with

```
make –f myMakeFile
```