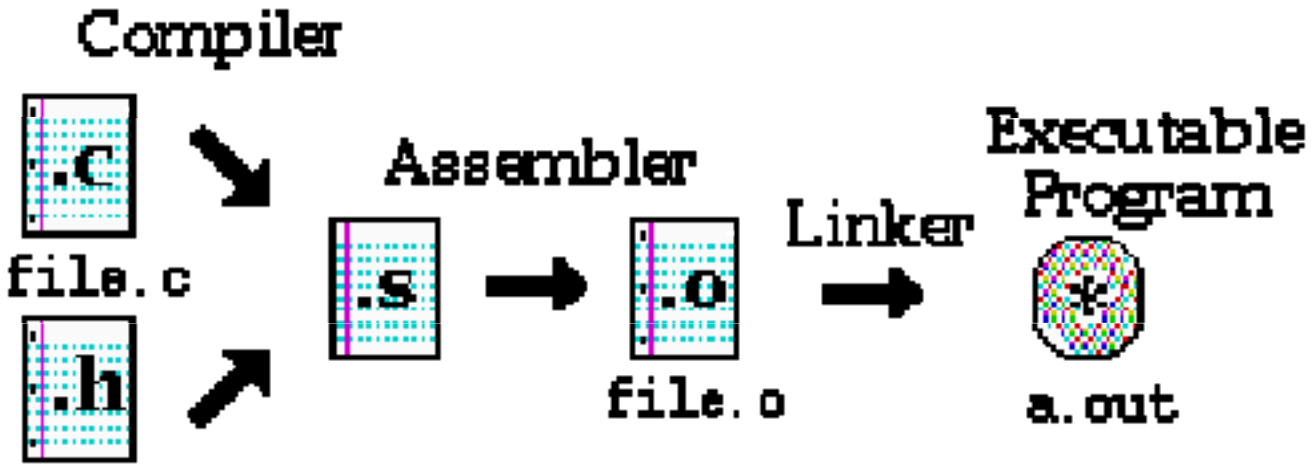
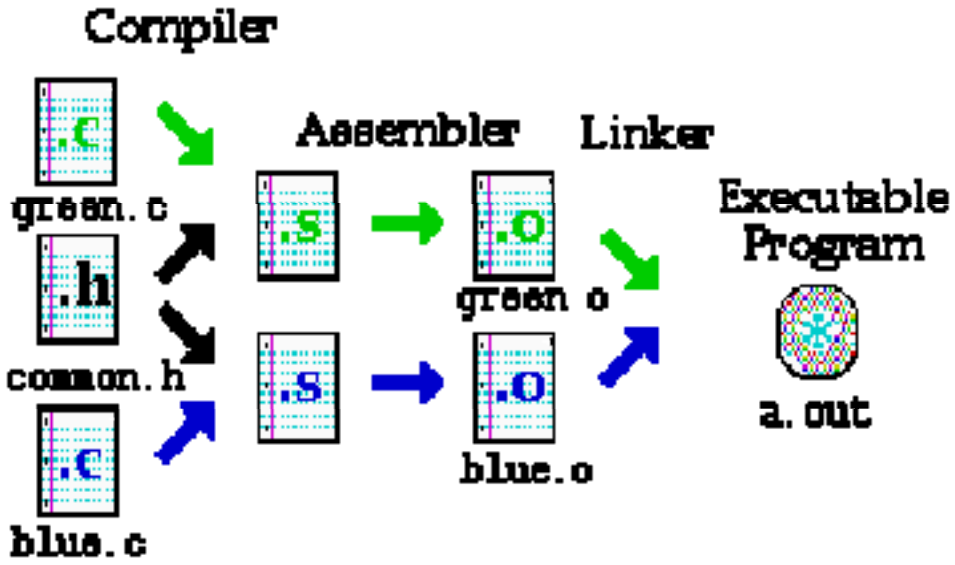
Makefiles

Tutorial adapted from
http://www.eng.hawaii.edu/Tutor/Make/
and
http://www.cs.bgu.ac.il/~sadetsky/openu/myMakeTutorial.txt

Compiling a single program



Compiling multiple related files



gcc green.c blue.c

OR

gcc –o program1 green.c blue.c

More efficient compilation

gcc –c green.c => create green.o

gcc –c blue.c => create blue.o

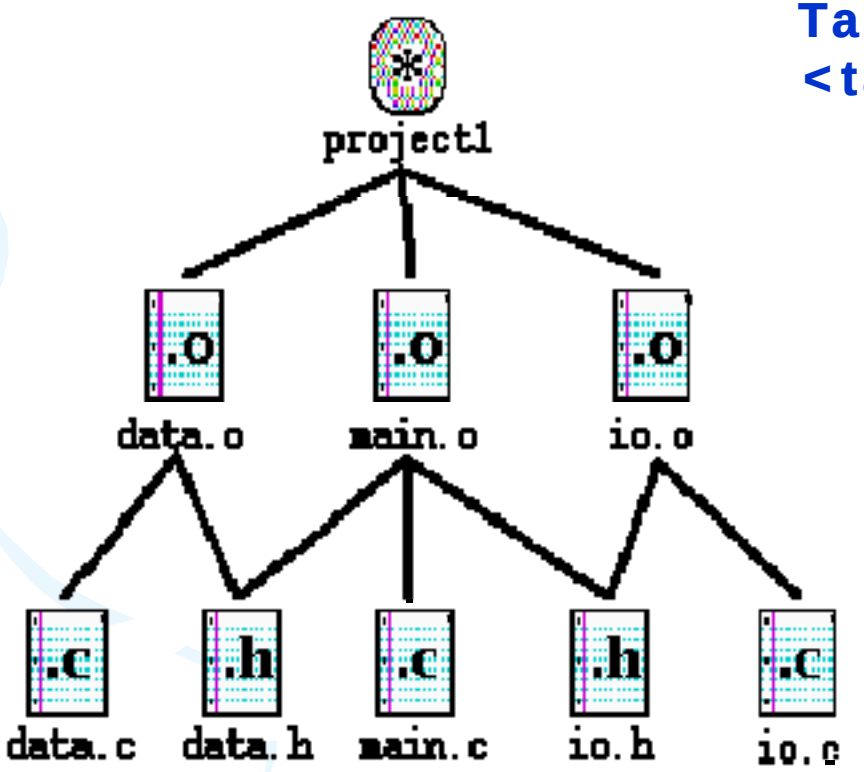
gcc green.o blue.o => link and create a.out

- The “c” flag compiles and creates object files while ignoring functions and variables defined externally.

Splitting your C program

- When you separate your C program into many files, keep these points in mind:
  - When you use a global variable,
    - Define it in one source file
      - int global_variable;
    - declare it in the header file
      - extern int global_variable;
    - include the header file in the appropriate C source files
  - Write all function prototypes in header files and include the header files in the appropriate C source files.
  - Exactly one of the files must have a main() function.

Dependency graphs and Makefiles



Target: Dependencies
<tab> How to compile

Makefile

```
all: project1

project1: data.o main.o io.o
<tab>   gcc data.o main.o io.o –o project1
```

Sample Makefile

```
projectl: data.o main.o io.o
   gcc data.o main.o io.o -o projectl
data.o: data.c data.h
   gcc -c data.c
main.o: data.h io.h main.c
   gcc -c main.c
io.o: io.h io.c
   gcc -c io.c
```

Makefile

- Create a file called Makefile.
- At the prompt, type “make”.

Make clean

clean:

<tab>rm –f *.o

At the prompt:

make clean

To create more than one executable

1. Include the following line in the Makefile

all: <exe1> <exe2> ….

2. Write the rest of the Makefile.

3. At the prompt, type “make”.

To make more than one….

- If you don’t want to re-create all the executables every time

a. Specify only the ones that should be with “all”.

OR

b. Don’t use “all”. Instead, type:

make <exe1> <exe2> ….

Macros in Makefiles

```
CC = gcc
DEBUG = -g
LFLAGS = -Wall $DEBUG -o
CFLAGS = -Wall –c $DEBUG
main: green.o  blue.o
        $(CC) $LFLAGS green.o blue.o
green.o: green.c common.h
        $(CC) $CFLAGS green.c
blue.o: blue.c common.h
        $(CC) $CFLAGS blue.c
```

References

- http://www.mtsu.edu/~csdept/FacilitiesAndResources/make.htm#What%20make
- http://www.eng.hawaii.edu/Tutor/Make/
- http://www.cs.bgu.ac.il/~sadetsky/openu/myMakeTutorial.txt

Doubly linked-list