Lab 1
Makefiles and C Programming

CSCI 4061
Introduction to Operating Systems

Jan 22, 2024

Overview

- TA Introduction
- Recitation policies/structure
- GCC and Code Execution
- Makefiles
- Exercise

TA Information

1. Name:
2. Email:
3. Office Hours:

Recitation Structure

Mini-lecture

- Review of important class content
- Q & A
- Short lecture on topics helpful for the exercise
- Discussion of Programming Assignments

Exercise

- Programming exercise related to topic taught in class or for starting your project
- Template code provided
- Submit the completed code as zip to Canvas before the submission deadline (Tuesday 11:59 pm)

Recitation Policies

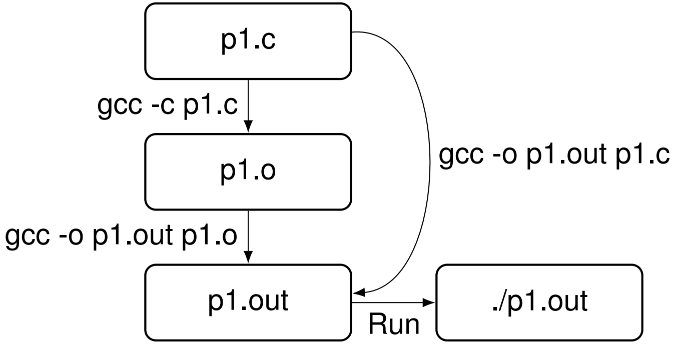
Policies

- Open resource / open collaboration
- Submit your own work that reflects your understanding and effort
- No switching of labs allowed
- No late submissions allowed

GCC

GNU Compiler Collection/ GNU C Compiler

- Compile C code to object file (may be executable)
- Your final code should compile on the gcc version in lab machines



GCC

Flags

`-c`: Unlinked non-executable object files, useful when multiple files are involved, generates filename.o by default

```
gcc -c p1.c
```

`-o`: Name the object file; when used without *−c*, it generates executable object file

```
gcc -o p1.o -c p1.c // generates unlinked object file
gcc -o p1.out p1.o  // generates linked executable
gcc -o p1.out p1.c  // generated linked executable
```

Makefile

GNU Make

- Controls generation of executables (other files)
- Set of rules to achieve complete an action or generate files
- Rule structure:

```
target: prerequisites ...
        recipe
        ...
```

- Prerequisites: Action or file that is required for the target
- Recipe: One or more actions carried out by make to generate target file or required action

```
p1.o: p1.c
      gcc -c p1.c
p1.out: p1.o
      gcc -o p1.out p1.o
```

Makefile

Usage

- Execute default/ first rule (p1.o)

```
$make
```

- Execute specific rule (p1.out)

```
$make p1.out # executes p1.o followed by p1.out
```

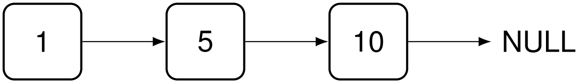
- Repeated calls to same make will compile the code only if there is a change in the dependencies

Programming Exercise

Activity

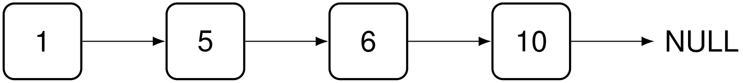
1. Create a sorted linked list using the given menu-driven template code. Complete the following function in the code.
   - `insertLL(value)`: inserts value while maintaining a sorted linked list
2. Complete the Makefile provided to generate the executable main
   - `make run` should execute the code

Sorted Linked List



Insert 6

1. Point next of 5 to 6
2. Point next of 6 to 10



Cases

- Empty linked list
- Insert a value before linked list head
- Insert a value at end of linked list
- Insert a value within the linked list

Deliverables

Individual submission: Zip and submit to Gradescope by Sept 12, 11:59pm

1. linked_list.h, linked_list.c
2. main.c
3. Makefile