Team Project Overview and Proposal Guideline

Prof. Myeonggyun Han

Computer Science and Engineering
Kyungpook National University

Project Overview

- **Goal**: Strengthen your system programming skills through practical, hands-on development.
  - Feel free to **choose a topic (자유 주제)** that interests you!
  - It must be **system programming-related** and **bigger** than the labs

- **Project Duration**: Approximately 6 weeks

- **Team Formation**: 2-3 members per team
  - Check your team number and teammates here:
    - https://shorturl.at/pdckq
  - Communicate with your teammates and **start early**!

- **Evaluation Weight**: 25% of final grade
  - Proposal: 5%
  - Final Presentation & Code: 20%

Project Timeline

1) Proposal Presentation

- **Date**: November 12 (during class)
  - Refer to this document (slide #4 and #8) for proposal guidelines
- **Slide Submission Deadline**: November 12, 9:00 AM (2 weeks from now)

2) Final Presentation

- **Date**: December 10 (during class)
  - Detailed guidelines will be provided later
  - Note: Final exam is scheduled for December 17 (during class)
- **Slide Submission Deadline**: December 10, 9:00 AM (6 weeks from now)

3) Code Submission

- Submit your code, along with a README and Makefile, via GitHub
  - Detailed guidelines will be provided later
- **Deadline**: December 21, 11:59 PM

▪ Important Notice

- Late tokens cannot be used for the team project deadlines

Project Proposal Guidelines

- **Submission**: presentation slides (5 to 10 slides)
  - How to submit: send via email to mhan@knu.ac.kr
    - Email Subject Format: [ELEC462] Team <팀번호> – Proposal Submission

- **Required Contents** (presentation duration: **5 minutes** per team):
  1) Team Number and Members (optional: team name)
  2) Project Topic and Description
     - **Motivation**: Why did you choose this topic?
     - **Overview**: What is the goal of your project? (Brief functional description)
  3) List of Major Functionalities
     - At least **3 core functionalities**
  4) Technical Implementation Plan
     - Planned use of **system calls and libraries**
     - Highlight key **system-level techniques** to be used

- **Important Notice: No Plagiarism**
  - **Do NOT copy** existing open-source code or your friends' code.
  - Any violation will result in an **F grade**, no exceptions.

Suggested Project Topics

Topics: *Choose something **bigger** than the lab assignments*

1) Network / Distributed System Programs

- Remote ls, pwd, grep, find
- Chatting or file transfer programs
- CLI-based file downloader (like wget)

2) Utility Programs

- Code similarity checker, spell checker, duplicate file checker, etc.
- Simple text editor (like vi)
- System resource monitor

3) Games

- Tetris, Ping Pong, Card games, etc.

4) **Any other interesting topic** that you would like to develop!

- For inspiration: https://mark.random-article.com/weber/unix/project/ideas.html

Suggested Project Topics (Cont.)

- **You may refer to projects from the previous semester:**
  - https://github.com/KNU-ELEC462/team-projects-2025s

- **Note on Game Topics**:
  - You may choose a game as your project topic
  - However, creativity will be evaluated more strictly for game projects
    - Refer slide #10 for evaluation criteria
  - We encourage exploring other ideas that better highlight your creativity
  - Still, if your **game includes unique system programming aspects** (e.g., low-level devices such as a custom controller or other low-level system features), you **can earn a high creativity score**

Project Requirements

- **Minimum Lines of Code**:
  - **500+ lines** (*excluding comments*)
- **Use of system calls**
  - You must use **at least 5** different system calls
    - Examples: open, close, read, write, lseek, opendir, closedir, readdir, stat, chmod, chown, ioctl, signal, kill, fork, exec, wait, etc.
  - Redundant or trivial usage will not be counted
- **Core Functionalities**
  - Implement **at least 3** distinct core functionalities
    - Each functionality must be <u>meaningful</u> & <u>independent</u> from a user's perspective
  - **Example**
    - "File upload" and "File download" ➔ counted as **<u>two</u>** separate functionalities
    - "User login" + "Chat functionality" + "File transfer" ➔ **<u>three</u>** core functionalities
  - Implementing additional functionalities may increase your complexity score

Project Requirements (Cont.)

- **Documentation**
  - Provide a clear and complete README and a working Makefile.
    - **README**: Instructions for building, running, and using your program.
    - **Makefile**: Automate the build process
  - A detailed guide will be provided later
- **Development Environment**
  - **OS**: Ubuntu 24.04
  - **Language**: **C**
    - Other languages and shell scripts are *NOT allowed*
- **Bonus Points; Extra credit will be given for**:
  - Implementing more than 5 core functionalities
  - Using advanced system programming features, such as:
    - Threads (pthread)
    - Socket programming (network communication)

Evaluation Criteria: Proposal

- **Proposal (50 pts, 5% of total grade)**
  - If you include all required contents in your presentation slides, you will receive a full score. If any required content is missing, points will be deducted accordingly.
- **Required contents**:

  1) Team Number and Members (5 pts)

  2) Project Topic and Description (10 pts)
  - **Motivation**: Why did you choose this topic?
  - **Overview**: What is the goal of your project? (Brief functional description)

  3) List of Major Functionalities (20 pts)
  - At least **3 core functionalities**

  4) Technical Implementation Plan (15 pts)
  - Planned use of **system calls and libraries**
  - Highlight key **system-level techniques** to be used

Evaluation Criteria: Final & Code

- **Final Presentation and Code (200 pts, 20% of total grade)**
  - **Code requirements (100 pts):**
    - Minimum 500 lines (excluding comments)
    - At least 5 different system calls, at least 3 core functionalities
  - **Documentation (20 pts):**
    - README and Makefile
  - **Complexity (30 pts):**
    - Implementing more than 5 core functionalities
    - Using advanced system programming features (e.g., threads, socket)
  - **Creativity (30 pts):**
    - Originality and uniqueness of the project topic
    - Creativity will be evaluated more strictly for game projects
  - **Presentation (20 pts):**
    - Slide contents and demo (detailed guide will be provided later)
  - Peer evaluation will be conducted