Operating Systems
Labs Agenda
2018/2019

http://d3s.mff.cuni.cz/osy

CHARLES UNIVERSITY
Faculty of Mathematics and Physics

Department of Distributed and Dependable Systems

Vojtěch Horký
horky@d3s.mff.cuni.cz
General Information

- **Course web site**
  - http://d3s.mff.cuni.cz/osy

- **Course mailing list**
  - osy@d3s.mff.cuni.cz
  - https://d3s.mff.cuni.cz/mailman/listinfo/osy

- **Labs**
  - According to the schedule on the course web site
    - Odd/even weeks: Wednesday *15:40 in SW2*
    - Vojtěch Horký, horky@d3s.mff.cuni.cz
    - Petr Tůma, tuma@d3s.mff.cuni.cz
Grading

**Labs credits**

- **Semestral tasks**
  - Five topics in total
  - Extended home assignments + credits test or smaller assignments during labs
  - Expected course of action
    - Think about extended assignment
    - Sketch implementation to verify you understand it and know how to code it
    - Come to the labs and implement the smaller assignment (same topic)
    - If you fail during the labs, finish the home assignment
  - 4 of 5 must be passed, when some of them are of “home” type, pass credit test

- **Selective semestral assignment**
  - For single students or a team of multiple students
  - Talk to us

**Written exam**

- Final score combined with number of passed tasks
**Overall grading**

- Let $P \in [0, 1]$ be the exam score
- Let $U \in [0, 1]$ be the lab score
  - Derived from number of passed tasks (min 80%)
  - Or the quality of selective assignment

**Final grade computation:**

- $U \times P \geq 0.7 \rightarrow 1$
- $U \times P \geq 0.55 \rightarrow 2$
- $U \times P \geq 0.4 \rightarrow 3$
- Otherwise $\rightarrow 4$
Semestral Tasks

- Five distinct topics
  - Memory management
  - Virtual memory
  - Synchronization
  - System calls
  - Device drivers
Semestral Tasks (2)

**During the labs**
- The topic will be announced using the mailing list in advance
  - Including a recommended reading list
- The detailed task specification will be handed out during the labs
- The task has to be implemented during the labs where it was handed out (90 minutes should be sufficient)
  - If successful, the task is marked as *fulfilled*
- The teacher is there to answer questions and give you basic guidance

**As a home assignment**
- The topic and a detailed task specification will be announced using the mailing list
- The task has to be submitted within week after the lab – see schedule on the web (the task is more complex than the task for the labs)
  - If successful, the task is marked as *conditionally fulfilled*
- For questions and basic guidance use the mailing list
Semestral Tasks (3)

- **Grading of each task**
  - **Fulfilling the necessary requirements**
    - Fulfilling the required properties of the implementation
    - Fulfilling the required interface
    - Passing the unit tests provided
    - ...
  - **Other (soft) criteria**
    - Quality of the implementation
    - ...

Semestral Tasks (4)

- **Getting the credits**
  - At least 4 *fulfilled* topics → credits
  - At least 4 *fulfilled* and *conditionally fulfilled* topics → possibility to take the credits test

- **Credits test**
  - Similar form as the semestral tasks
    - Slightly more complex, usually for 2 hours
  - Similar requirements and criteria
    - If passed → credits
  - During the exam period
    - Up to 2 tries
Individual (bespoke) assignments

- A non-trivial topic related to real operating systems
  - For somewhat experienced developers
- The topic, schedule, requirements and credits criteria need to be agreed on with the teacher
- For single students or a team of students (up to 4)
  - This affects the complexity, deadlines, etc.
- Possibility to consult with external companies
  - Oracle, Red Hat, SUSE, Microsoft, Avast, ...

Possible topics

- HelenOS, GNU/Linux, Windows, macOS, *BSD, MINIX 3, seL4, Genode, Haiku, RTEMS, ...
- See respective Google Summer of Code ideas lists for inspiration
Selective Semestral Assignments (2)

- **Deadlines**
  - Depending on the agreement with the teacher, but usually:
    - **Beta/Milestone:** End of the winter semester
    - **Final version:** End of the summer semester

- **Grading**
  - Quality of the implementation (programming techniques, data structures)
  - Fulfilling the required interface
  - Efficiency
  - Coding style quality (code readability, structure, consistency)
  - Quality of comments and of the developer's documentation
  - $U \geq 0.5 \rightarrow$ credits
Side-note: Cheating, etc.

- Only your original code will be graded

  - This course requires individual/independent work
    - You want to learn how to code an operating system, right? Well, there is no other way to learn it than actually writing the code yourself.
    - Learning the tricks to pass off foreign code as original code is **not** the purpose of this course
      - It is surprisingly easy to detect such frauds (with and even without tools for that purpose)
      - Frankly, it is embarrassing for both parties

- Practical rule: Whenever you use someone else's code, declare it clearly as such (never mislead anyone about the source of any code, even unintentionally)
  - If declared properly, we won't punish you for using foreign code (but it might obviously affect the grading in certain cases)