Developing Soft and Parallel Programming Skills Using Project-Based Learning

FALL 2019 - Group 5Pis
Anthony Asilo, Chinonye Ekwenchi, Daniel Ingo, Nick Tatgenhorst, Jeremias Caceres

Planning and Scheduling

As a coordinator, Anthony Asilo is in charge of maintaining the groups efficiency and productivity throughout the project by organizing and hosting meetings. Additionally, Anthony will create the Slack, Github, and communicate with the members of the group through the means of a GroupMe. Anthony will also hold onto the PI and delegate tasks based on the availability of members and priority of tasks and refer to the schedule.

Chinonye Ekwenchi will be in charge of participating in the Teamwork Basics, assisting in the PI Installation and ARM Assembly Programming, partaking in the Report for her section of choice, and presenting her part in the video. Additionally, Chinonye will stay in connection with her group through GroupMe.

Daniel Ingo will be in charge of participating in the Teamwork Basics, assisting in the PI Installation and ARM Assembly Programming, partaking in the Report for his section of choice, and presenting his part in the video. Additionally, Daniel will stay in connection with her group through GroupMe.

Nick Tatgenhorst will be in charge of participating in the Teamwork Basics, assisting in the PI Installation and ARM Assembly Programming, partaking in the Report for his section of choice, and presenting his part in the video. Additionally, Nick will stay in connection with her group through GroupMe.

Jeremias Caceres will be in charge of participating in the Teamwork Basics, assisting in the PI Installation and ARM Assembly Programming, partaking in the Report for his section of choice, and presenting his part in the video. Additionally, Jeremias will stay in connection with her group through GroupMe.

Teamwork Basics

1. What to do to get the task accomplished and the team members' satisfaction high?
   - Communicate everything and work when you have time to.
2. As a team, select two cases out of the four mentioned in Handling Difficult Behavior.
   - Argues - Coordinator will try to resolve complications, if not will get a Mussa
   - Talks too little - be nice and try to get them to talk and if not then as long as they are doing their work it's all good
3. When making decisions, If the team is having trouble reaching consensus, what should you do? (use your own words and your own context)
   - Vote for the decision
4. What should you do if a person may reach a decision more quickly than others and pressure people to move on before it is a good idea to do so?
   - Slow down!
5. What happens if most people on the team want to get an "A" on the assignment, but another person decides that a "B" will be acceptable?
   - Don't purposely do half the work and we will get an "A" (hopefully)

Work Norms: How will work be distributed?
- Coordinator will distribute in Task 1 (Planning and Scheduling)

Who will set deadlines?
- Coordinator will set deadlines (refer to the schedule)

What happens if someone doesn't follow through on his/her commitment (for example, misses a deadline)?
- Dependant on the reasoning, deduction anywhere from 0-50%

How will the work be reviewed?
- Coordinator or volunteer will revise

What happens if people have different opinions about the quality of the work?
- Sort it out and compromise

What happens if people have different work habits (e.g., some people like to get assignments done right away; others work better with the pressure of a deadline).
- Work done well is work done well

2. Facilitator Norms: Will you use a facilitator?

How will the facilitator be chosen?
- The coordinator will be the facilitator unless otherwise specified

Will you rotate the position?
- When necessary, anyone and everyone can facilitate

What are the responsibilities of the facilitator? (see below)

- Focus on tasks created
- Makes sure all members of the group are participating
- Keep members on track in regards to time frame
- Come up with alternatives when the group is stalled
- Address team issues
- Clearly and comprehensively condense group decisions

Focus the team on the task (both short term and long term) • Get participation from all team members • Keep the team to its agreed-upon time frame (both short term and long term) • Suggest alternative procedures when the team is stalled • Help team members confront problems • Summarize and clarify the team’s decisions

3. Communication Norms: When should communication takes place and through what medium (e.g., do some people prefer to communicate through email while others would rather talk on the phone)?
   - Groupme for offline, Slack for online, google docs for reports and

4. Meeting Norms: What is everyone's schedule?
   - Please Refer to the schedule posted in groupme

Should one person be responsible for coordinating meetings?
- Yes, coordinator, but extremely flexible and will meet when members are needed or available

Do people have a preference for when meetings are held?
- Not necessarily, as long as 2-3 people can meet!

Where is a good place to hold meetings?
- Library, Discord

What happens if people are late to a meeting?
- Acceptable, give notice, 15% for every 30 minutes not notified

What happens if a group member misses a meeting?
- 25% if not notified,

What if he/ she misses several meetings?
- 0%

5. Consideration Norms: Can people eat at meetings? Smoke?
   - Yes, Yes

What happens if someone is dominating the discussion?
- Intervene and explain that their time to shine is over.

How can norms be changed if someone is not comfortable with what is going on in the team?
- Talk to coordinator, if not, other members. Last resort is Mussa

Raspberry PI Installation

Upon receiving the PI Kit and reviewing Task 4, it was concluded that installing RASPBIAN on the MicroSD card was the first step. Upon downloading RASPBIAN 10 (Debian Buster) (because there was no .zip for RASPBIAN 9 Debian Stretch), we utilized the Etcher program to burn RASPBIAN 10 onto the MicroSD then proceeded to load it onto the PI itself, we opened the Connect_PI_with_Monitor_and_Keyboard.pdf to realize that we need a monitor, keyboard, and mouse. However, the green light blinked meaning the OS installed successfully. We then uninstalled RASPBIAN 10 (Debian Buster) to try to install RASPBIAN 9 (Debian Stretch) when we realized we could use Etcher on an .iso file as well. Upon installation, the green light did not flash for Stretch. Because of this, and a bit of research we conducted, we came to the conclusion that RASPBIAN 10 (Debian Buster) had no major key differences in architecture; we proceeded to reinstall RASPBIAN 10 (Debian Buster), however it was unsuccessful. Upon the completion of Etcher, it said “The Disk you inserted could not be read”. We had to format the MicroSD to have a format of MS-DOS (FAT) and a scheme of Master Boot Record. It connected successfully and the green light was blinking. We continued on with the assembly with the next step being connecting the Pi to a monitor, mouse, and keyboard. Unfortunately, we could not rent a keyboard and no one in the group currently had one, so we went and purchased one. After that we hooked up the Pi to the monitor, which we were treated with a “No Signal.” After doing a bit more research on connecting the Pi to monitors, we stumbled upon the suggestion to go into the Config.txt file and uncomment the line hdmi_force_hotplug=1. This changed the “No Signal” message to a “No Sensor” one. In response to this, we switched from the older monitor we were using to a newer television, which enabled it to finally function.

Assembly ARM Programming

After the installation of the Operating System, we began working on the ARM Assembly Program. The program was fairly easy to implement because of the code being provided to us in the ARM_Assembly_Programming page of the project instruction set. After we wrote, saved and built the first program, which had us move the number five into the first register, subtract one from register one and add four again to register one with the number one being moved to register seven. After we placed a breakpoint in line twelve, enabling us to look into the registers to see what the code was doing while it was running. What we saw was similar to what we have observed in our labs when we look into the registers, with register one having the number eight in it and register seven having the number one in it. Similarly, for the second program we were able to use the code provided to us in the previously mentioned Arm Assembly page, of which had us move the number ten into register one, move the number eleven into register two, move the number seven into register three and the number two into register four before adding the contents of the second register into the contents of the first, multiplying the contents of the third register by the contents of the fourth, and subtracting the contents of the third register out of the contents of the first. Because this program was larger than the first, we placed the breakpoint further down in line fourteen so that we could view the results of the code in the registers. What we were shown was that in register one had the number seven, register two had the decimal number eleven and hex representation B, register three had the decimal number fourteen and hex representation E, and register four having the number two within it. While assembling the first program we received a few issues in the code because we had forgotten to place the ‘#’ prefix in front of the numbers being moved, added, and subtracted from their assigned registers and because of this, we did not make that mistake in the second program.

Appendix

Slack: http://csc3210-5pis.slack.com
GitHub: https://github.com/pillared/CSC3210-5Pis