CS15 Pair Programming Missive 2018

Why Pair Program?

Pair programming has many benefits. The main motivation behind it is simple: many students find that CS15's strict, non-collaboration policy on all projects can be difficult to navigate and, at times, isolating. Additionally, many upper-level CS courses, as well as industry-level programming, is not siloed in this way.

While we fully recognize and celebrate the benefits of collaborative coding, CS15's culture of larger projects and no exams requires that bigger assignments remain individual (see our Collaboration Policy here). To better reflect what you will see later in the field of CS and to foster a more friendly and open environment in the course, all labs completed in CS15 will be completed in pairs. By working in pairs, you will be able to share your ideas with your peers and in turn be exposed to their ideas about a common problem. With open communication, you will hopefully enjoy a more welcoming environment and be better prepared for future work in the CS department here at Brown and beyond.

How to Pair Program

It may sound simple, but a successful partner programming experience depends on both partners agreeing on a common set of guidelines. Remember that as part of a group, you are also responsible for working together productively with your partner and resolving intra-partner stress if it should arise. The following are some general guidelines for CS15 labs.

Before the Lab

Your section leaders will create and assign pairs before the start of the first lab of the week. You will be notified of your pairing by email. If you have any immediate concerns with your pairing, contact your section leader as soon as possible to resolve those concerns before the start of the lab.

Drivers and Navigators

Originally introduced to Brown CS by CS17, in CS15 you will follow a driver-navigator style division of labor. The **driver** is responsible for typing code and vetting the suggestions of the navigator to arrive at an agreed-upon codebase. The **navigator** is responsible for not only telling the driver (in semi-general terms) what to type, but is also responsible for reviewing the driver's work. In addition to catching incidental mistakes, the navigator considers the code at a more strategic level: how will this fit with the rest of the code? Will this implementation require changes elsewhere? Could we design this program, method, or class better?

Every fifteen minutes or so, each pair will switch roles by sliding the keyboard over.

More specifically:

- While we hope you will switch roles every 15 minutes, you can wait for a natural breaking point. No one should drive more than 20 minutes or less than 10.
- It's important that both partners spend an approximately equal amount of time in each role. Some programmers may enjoy driving better, others navigating; however, neither partner does the other a favor by letting them spend more time in a fixed role.
- Switching roles is done by sliding the keyboard — noticeably, not by rearranging the chairs and adjusting the monitor. That is to say, regardless of roles, you are both working together on the problem at hand. As such, you are seated next to each other with the monitor adjusted so you can both see the screen easily.

What pair programming is *not* is a divide-and-conquer strategy, where two people will split up a task and each person works on "their part" separately. We encourage working on problems and writing code collaboratively during labs, each step of the way.

Completing a Lab

Labs are designed to be easily completed in the 90-minute period. However, it is perfectly natural that some students will want or need more than 90 minutes to complete the lab.

If you do not complete the lab during the 90-minute period, you must complete the rest of the lab individually. A TA will copy the portions of the lab your pair worked on collaboratively to your individual course folder. You must use only the files copied to your individual folder to

complete the lab. You may *not* work on the lab with a partner outside of your scheduled lab period. To do so is to violate the collaboration policy. You may still be checked off and receive full credit for a lab for which you completed some pieces individually. Alert a TA at the beginning of the next lab period that you need a checkoff, and they will be happy to assist.

Being a Team Player

Partner programming is great — provided that all group members act in the best interest of their group. Some suggestions to complete labs smoothly:

Respect each other's time

Arrive on time to each lab section. If you need to switch labs, remember that you must email your section leaders and the leaders of the section you hope to attend *before* the first lab of each week. If a legitimate, unexpected emergency arises on the day of your lab, notify your section leaders as soon as possible. Failure to notify your section leaders means that your partner will be left alone at the start of lab, and you will be penalized. Respect your partner's time during the lab as well. Don't text or check email during your pair programming sessions; stay focused on the joint task at hand.

Work as a Unit

Take collective ownership of the code you and your partner are writing, abandoning the notion of "my part" and "your part." And, in light of that view, make sure you speak up when you think an error has been introduced, and don't be too proud to admit a mistake. Again, communication is crucial. Perhaps you're having a hard time adjusting to pair programming, or your partner continues to say "my" instead of "our" work. Speak your mind and work through any problems. Offer to cease driving when it's time ("Hey, would you like a go at the keyboard?"), and remind your partner it's time to switch when you're navigator ("Mind if I drive for a while?").

Be open-minded and light-hearted

One of the most important predictors of success in pair programming is buy-in: if you are determined to make the practice fail, it will. Choose a healthy perspective: laugh at your mistakes, apologize if you hurt your partner's feelings, and, more generally, look at pair programming as an opportunity to learn.

Seek advice when you need it

Partner programming is new to CS15. It's likely new to you as well as your TAs. As such, the entire course staff is more than willing to provide any advice, guidance, or clarification that you

may need. We want your feedback — never hesitate to tell us what you think is and is not working.

If you feel that you are encountering a problem with your partner for a given week, and don't feel comfortable approaching your partner directly, always feel free to contact your section leader or the HTAs of the course. They can handle your comments with any degree of anonymity you desire, whether that be not pairing you with a specific person in the future, or addressing problems during lab as they arise.

It is not acceptable in pair programming for a single person to do all (or even most) of the work and then add their partner's name. Academic honesty is always more important than fulfilling a pair programming requirement. If your partner is unwilling to help or fails to show up at your lab time, or if your partner is unwilling or resistant to letting you contribute your share of the work, contact your section leaders or the HTAs, who will be happy to assist in resolving the conflict.

Partner Programming and the Collaboration Policy

While we hope you will enjoy the group aspects of partner programming in labs, it should go without saying that partner programming must stop as soon as the lab period ends (or you complete the lab). You are still bound by the collaboration policy, which continues to enforce absolutely no partner work on any other assignment for CS15. This means that during labs, you may only discuss the content of that particular lab, as it pertains to only that particular lab. You may *not:*

- Discuss how concepts covered in the lab relate to a project
- Discuss the handout for any project or homework
- Discuss implementations for any project or homework
- Discuss any Java concepts that pertain to the lab in any context besides that of the lab.

This is not a complete list! You are still bound by the collaboration policy for the course, which provides a much more complete discussion of allowed and forbidden collaboration. As always, if you have any questions, no matter how small, please contact the TAs.

We look forward to your feedback on this mode of collaborative coding, and hope you have fun with this mechanism for learning!