$Id: pair-programming.mm,v 1.50 2021-01-05 17:55:07-08 - - $
PWD: /afs/cats.ucsc.edu/courses/cse110a-wm/Syllabus/pair-programming
URL: https://www2.ucsc.edu/courses/cse110a-wm/:/Syllabus/pair-programming/

```
I am working alone, not doing pair programming.
All of the work submitted here is code written by me,
except for code copied from the instructor's directories.

I have read and understood the UC Santa Cruz
Academic Misconduct Policy for Undergraduates.
https://ue.ucsc.edu/academic-misconduct.html

-- Firstname Lastname (cruzid@ucsc.edu)
```

Figure 1. Solo programming `README`

```
We are working together doing pair programming.
All of the work submitted here is code written by us,
except for code copied from the instructor's directories.

We both have read and understood the UC Santa Cruz
Academic Misconduct Policy for Undergraduates.
https://ue.ucsc.edu/academic-misconduct.html

We acknowledge that each partner in a programming pair is
responsible for the work performed by the other partner,
and all work has been reviewed by both partners working
together.  Both of us know the format of the PARTNER file.

-- Firstname1 Lastname1 (cruzid1@ucsc.edu)
-- Firstname2 Lastname2 (cruzid2@ucsc.edu)
```

Figure 2. Pair programming `README`

1. Solo programming

Pair programming is optional. If you choose not to do pair programming, submit a **`README`** with the statement shown in Figure 1. (Substitute name and cruzid shown in italics.) You may then ignore the rest of this document, but must comply with this paragraph. Also, be sure to look in the assignment directory's **`.score/`** subdirectory for instructions to graders.

The file **`README`** must be spelled exactly that way, in upper case. Not **`README.txt`**, or **`readme`**, or other spellings which differ from the name required. Similar rules apply to the file **`PARTNER`**, if applicable.

2. Pair programming

Pair programming is a software development technique in which two programmers work together at one keyboard. One types in code while the other reviews each line of code as it's typed in. The person typing is called the ***driver***. The person reviewing the code is called the ***observer***. The two programmers switch roles frequently.

While reviewing, the observer also considers the strategic direction of the work, coming up with ideas for improvements and likely future problems to address. This frees the driver to focus all of his or her attention on the "tactical" aspects of completing the current task, using the observer as a safety net and guide.

Some benefits of pair programming are : design quality, learning and training, overcoming difficult problems. You have another head to bounce problems off of without cheating. Some drawbacks are : work preferences, difficulties in scheduling meetings, working with an uncooperative partner.

`http://en.wikipedia.org/wiki/Pair_programming`

3. Guidelines

If you wish to do so, you may use pair programming to develop your programs. Following are some guidelines :

1. Pair programming is optional. You may choose a pair partner, or you may work independently. Both partners are responsible for the other partner's actions.
2. Before choosing a partner, interview each other to verify that he or she has the necessary knowledge and that your levels of accomplishment are roughlly the same. You are entirely responsible for you choice of partner, or for you choice of working alone.
3. Spend most of your development time together at the same workstation.
4. After working alone each partner should review each line added by the other partner. Use comparison tools to check changes between various versions.
5. Alternate between driving and observing approximately after every 30 minutes of programming time.
6. One partner submits the code and the other watches to verify that the correct code has been submitted.

4. Requirements

In addition to the requirements for students working alone, pair programming teams have a few additional requirements.

1. The **`README`** submitted by each partner must be the same and must agree. See Figure 2. Both partners must submit a **`README`** and a **`PARTNER`** file, but only one partner should submit the code.
2. Both partners name and username must appear in a comment at or near to top of ***every*** file submitted, except for the **`PARTNER`** file.

3. If both partners have submitted the project, then both partners must state in their **README** which of the submissions is to be graded. Only one partner's work will be graded.
4. The **PARTNER** file must contain one single word, namely the partner's username. This is most easily done from the command line rather than using an editor. Example: `echo` cruzid `>PARTNER`
5. Use the script **partnercheck** to verify correctness. Obviously these files can not be kept in the common working directory. The **PARTNER** file is used to match partners and automatically enter the score into both partners' grade file line items.
6. ***Error:*** The **PARTNER** file must contain your partner's CruzID and a new line character, and nothing else: not a personal name, not an entire email address. It is not read by the grader. it is read by a script that matches partners.
7. If you and your partner come up with irreconcilable differences and can no longer work together, explain this clearly in the **README**, each separately, and do not work together again. Each should then submit work separately.
8. In case of breakup, do not submit a **PARTNER** file. Since you can not delete a file once submitted, if you have already submitted on, resubmit a **PARTNER** file with zero bytes in it.
9. Pair programming code will be graded in the usual way by the grader, but additional points will be deducted for any of the following:
   - Both partners' names and usernames were not in a comment at or near the top of each file submitted (except for the **PARTNER** file).
   - The **README** file did not contain the required information.
   - The **PARTNER** file was not exactly correctly formatted.
   - In case of a team breakup, the reasons were not documented in the **README**.
10. ***"But I thought that my partner submitted the code."*** That is absolutely not an acceptable excuse. When you are doing pair programming both partners are sitting together at the same workstation and it is the observer's duty to verify that the driver actually did submit the correct version of the code.
11. The grader will fill in **.score/SCORE.pair** along with the **.score/SCORE** file. Make sure to verify that you have done what is required.

5. Sharing files

When you are working together at the same computer, you will be logged into one or the other's account. If you are working separately on occasion, one partner might want to access files in the other partner's account. Be sure that each partner has a backup copy of all common files. Each partner should regularly make a backup copy of the common code, in case one partner deletes everything, and so that each has copies of the code after the end of the quarter.

https://its.ucsc.edu/unix-timeshare/tutorials/afs.html
https://its.ucsc.edu/unix-timeshare/tutorials/afs-quickref.html
https://its.ucsc.edu/unix-timeshare/tutorials/afs-primer.html