

Oct. 4, 2007. Mack Grady, grady@mail.utexas.edu. Dept. of ECE, U.T. Austin

I use this program to create weekly random lab partner pairs in EE362L and guarantee "no repeated partners." I've written it in Visual Basic and have gradually improved it over about two years. Both source and executable are included. You do not need to have VB on our PC - simply use the executable version *.exe.

The students actually prefer to be assigned partners - that is one less worry for them. I generally do not permit semester-long partners - that arrangement often leads to one being the leader, and one the follower. It is better experience for them to have a new partner every week, or at the start of every multi-week lab.

Use the to pair students from the entire class (Roster_all.txt). Or, if you want to break the starting team numbers for individual sections, use individual roster files Roster_1.txt, Roster_2.txt, Roster_3.txt, Roster_4.txt. I use individual section roster files to restart the team number sequence according to our physical lab locker numbers:

```
(for section 1, start with team number 247,...)
(for section 2, start with team number 259,...),
(for section 3, start with team number 271,...),
(for section 4, start with team number 283,...).
```

The version in this zipped file is the same program is the same one I use, but the lock combination files have been changed, and the student names and emails have been made generic 1,2,3, etc.

Comments:

1. Input file Roster_all.txt contains all class members. Input file combinations_all.txt contains all lock combinations. See the files for format.

2. Input file Roster_1.txt contains all class members in section 1. Input file combinations_1.txt contains all lock combinations used by section 1. There are black locks, and blue locks, for weekly swapping.

3. The program can be run for "all sections," or separately for all four sections.

4. File team_names_history.txt contains student name pairs for past labs. When the program runs, it reads that file and re-seeds the pairing until repeats are avoided. The "Save History" button appends the present solution to team_names_history.txt. That is how the history file is initialized.

5. Backup your team_names_history.txt file before running the program in case you make a mistake when appending with the "Save History" button.

5. Pairs (i.e., teams) are chosen among the names with the same leading integer in the Roster_* files. Students with the same leading integer are a pool from which random pairs are chosen.

6. If there is an odd number of students, then one is randomly chosen as "SOLO.". To force a solo, let that student have a unique leading integer. To force a pair, give both students the same unique leading integer.

7. To avoid repeated student pairs, step thru or let the program iterate until it finally finds a solution with no repeats or has only the required repeats (for example, some students want permanent solo status, or some students are permanently paired).

8. The program produces an email file, which I cut and paste into Eudora, one team at a time, and email to that team. It includes their lock and combination. I print out that entire file, and we use it in the lab to call teams forward and take role. I do this the day before, so when they arrive to their lab session, they already know who their partner will be and may email each other in advance.

9. After a successful run for a lab, e.g. Lab 2, I save the roster files and email files with extension _2. Otherwise, they will be overwritten the next time I run the program.

Regards, MG