Lab 4. Majority Voting

In this lab we will use pipes to implement a majority voting system that involves one judge and *N* voters. *N* is specified by the user and should be equal to or greater than 3. The judge is the parent process, and the voters are child processes. At each round, every voter will randomly choose to vote yes or no, show the vote on the screen, and send the vote to the judge. The judge will then declare the final decision following the majority rule and display it on the screen. The voting continues until it reaches the maximal number of rounds, and all processes should be terminated properly before exit. The same group votes for all the rounds.

Step 1: Ask user to input number of voters and max number of rounds

Step 2 (1st for loop): The judge process creates N pipes using **pipe()** system call.

Step 3 (2nd for loop):

The judge process creates N child processes using **fork()** system call. (Only the judge process can call fork and child processes don't call fork.)

NOTE:

- At this step, each child process needs to keep the voter id (= the index of the loop = the pipe id) that will be used in Step 4. You may set the judge process's voter id to be -1 to differentiate it from the voter processes.
- The processes should not call exit() in this step. All processes should continue to Step 4.

Step 4 (3rd for loop): At each round,

For the judge process (if the process's voter id is -1):

- (**4th for loop**) Collect votes from voters using **read()** system call
- Output the majority voting result

For each voter process (if the process's voter id is between 0 and N):

- Send the vote to the judge process using **write()** system call (Voter Process *i* uses Pipe *i* to communicate with the judge, DO NOT use a loop here)

Sample Output:

```
(base) Jinzhus-MacBook-Pro-2021:Lab4Pipes jinzhugao$ ./lab4
Enter the number of voters: 3
Enter the number of rounds: 2


------ Round 1 -------
Round 1: Voter 0 votes: YES
Round 1: Voter 1 votes: YES
Round 1: Voter 2 votes: NO
--- Round 1: The voting result is a YES.



------ Round 2 -------
Round 2: Voter 0 votes: YES
Round 2: Voter 1 votes: NO
Round 2: Voter 2 votes: NO
--- Round 2: The voting result is a NO.
```

Note that your output may be different each time you run your code and may also be out of order, which is perfectly fine as long as the voting result for each round is correct.

Submissions:

- (80%) Well-commented source code
- (20%) Write a report that
    - has a diagram describing how processes interact/communicate with each other
    - describes how to compile and run your program
    - includes screenshots of your running program

Late Assignments and Make-up Work: The only acceptable excuses for missing an assignment due date are serious illness, death in the immediate family or important professional activities. Illness or death in the family may require documentation. Excuses for professional activities must be approved by the instructor in advance.

Work may be submitted up to 48 hours late with a 20% reduction each day in score. No work may be submitted more than 48 hours after the due date without prior permission from the instructor.

Individual Work and Collaboration: Computer professionals usually work in a cooperative environment, yet proper assessment requires that work be done by individuals. To alleviate confusion, the following policy will be followed:

Collaborative work is encouraged. This includes students working together on problem sets, planning solution strategies and helping each other to debug programs. Collaboration must stop short of the writing of program code or English that represents your work. You may not directly copy the work of another student. It is your responsibility to ensure that the work you submit is an honest representation of your own understanding of the material.

Marginal cases will be resolved by oral examination of the students involved. If they understand the material in the assignment, it will be considered honest collaboration. If they do not, then it will be considered academic dishonesty.

Collection of Work for Assessment: Student work may be retained to assess how course learning objectives are being met and for accreditation purposes.

Honor Code

The Honor Code at the University of the Pacific calls upon each student to exhibit a high degree of maturity, responsibility, and personal integrity. Students are expected to:

- act honestly in all matters
- actively encourage academic integrity
- discourage any form of cheating or dishonesty by others
- inform the instructor and appropriate university administrator if she or he has a reasonable and good faith belief and substantial evidence that a violation of the Academic Honesty Policy has occurred.
- Violations will be referred to and investigated by the Office of Student Conduct and Community Standards. If a student is found responsible, it will be documented as part of her or his permanent academic record. A student may receive a range of penalties, including failure of an assignment, failure of the course, suspension, or dismissal from the University. The Academic Honesty Policy is located in Tiger Lore and online at http://www.pacific.edu/Campus-Life/Safety-and-Conduct/Student-Conduct/Tiger-Lore-Student-Handbook-.html.

SOECS Honor Code Implementation Process

The School of Engineering and Computer Science holds all of its students to a strict standard of academic integrity. In the case of a suspected violation of the University academic honor code, faculty members will evaluate the alleged infraction and may take a range of actions, up to and as serious as submitting an "F" or "No Credit" for the course. They will also report it immediately to the chair of the department, the School Assistant Dean's Office, and the Office of Student Conduct and Community Standards. The Assistant Dean's Office and the Office of Judicial Affairs may pursue further sanctions, up to and as serious as disqualification from the University, based in part on the seriousness of the incident and any prior violations. Students may also be prevented from dropping or withdrawing from the course, even if the deadline to do so has not expired.