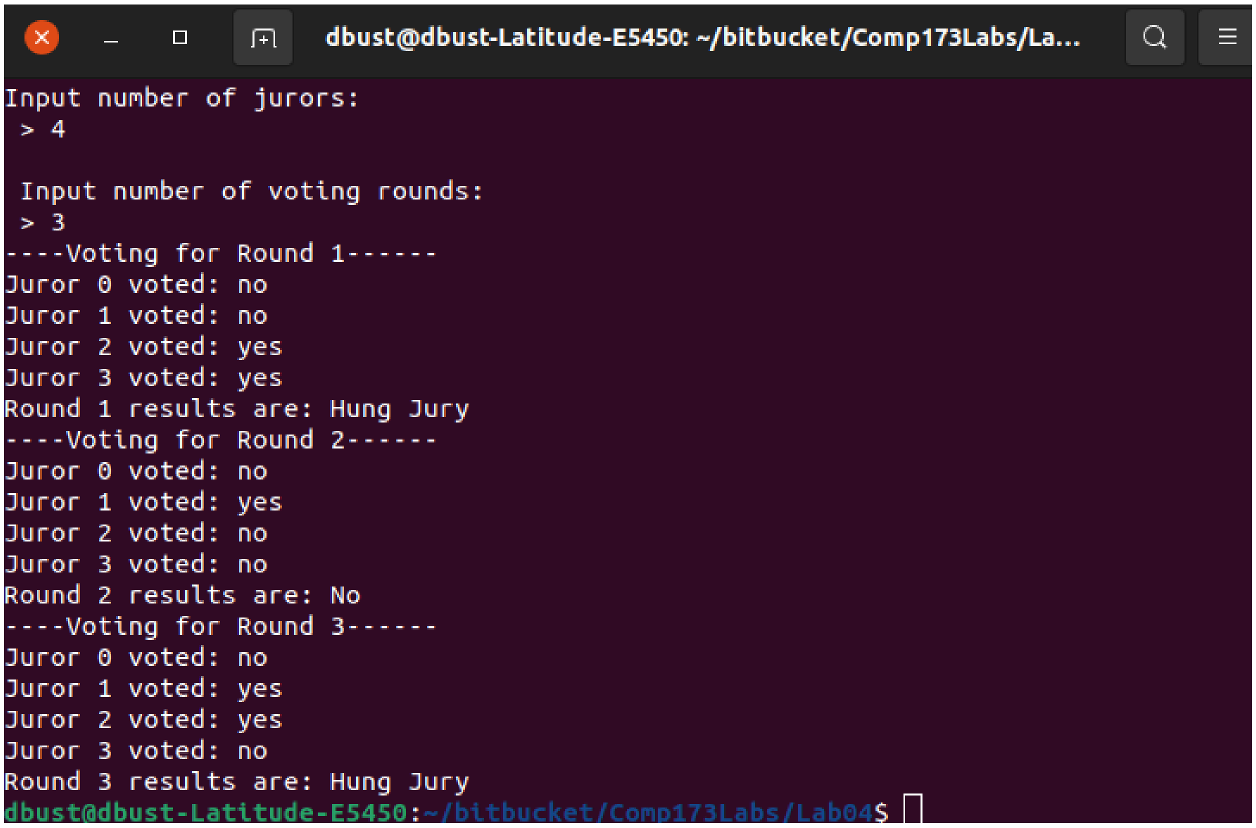
To run the program simply compile the judge.c file:
gcc judge.c
Then run the resulting program:
./a.out
Answer each prompt with an integer value then press enter



```
dbust@dbust-Latitude-E5450: ~/bitbucket/Comp173Labs/La...
Input number of jurors:
 > 4

 Input number of voting rounds:
 > 3
----Voting for Round 1------
Juror 0 voted: no
Juror 1 voted: no
Juror 2 voted: yes
Juror 3 voted: yes
Round 1 results are: Hung Jury
----Voting for Round 2------
Juror 0 voted: no
Juror 1 voted: yes
Juror 2 voted: no
Juror 3 voted: no
Round 2 results are: No
----Voting for Round 3------
Juror 0 voted: no
Juror 1 voted: yes
Juror 2 voted: yes
Juror 3 voted: no
Round 3 results are: Hung Jury
dbust@dbust-Latitude-E5450:~/bitbucket/Comp173Labs/Lab04$
```

My program follows the instructions very closely. The main thread creates a 2d array of ints that function as pipes. The main thread then creates n number of child threads where n is inputted by the user each with their own ID number from 0 to n-1 while the main thread retains the ID number of -1. After all threads are created, they enter the voting loop where the jurors cast votes and the judge tallys them and prints the results of each round. After voting has finished all the threads are ended while the main thread waits for them all the end and after ends itself. The two most interesting problems I faced were making the threads generate different numbers and learning to not overread the pipe buffer.