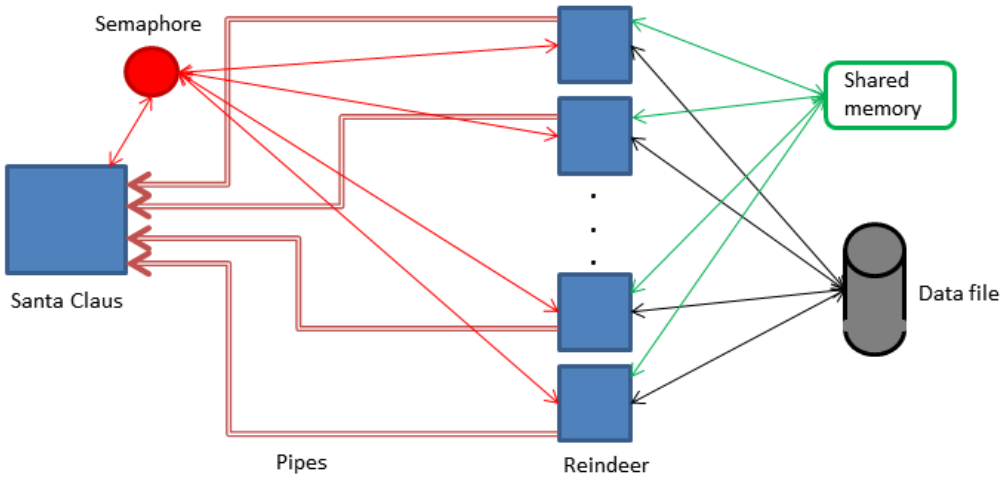
HW-9 CS451 100 points Due: Monday, December 9

This is the story of the stash of truffles and the 9 flying reindeer (and Santa Claus) and the motto of this *last* assignment is: Ave Caesar! Morituri te salutamus - Hail Caesar! We who are about to die salute you.

Now Santa Claus and the 9 flying reindeer went for a stroll in the woods one day, looking for some truffles to eat. Lo and behold they found someone's stash of truffles. The little flying reindeer eyes grew huge with desire. But, Santa Claus knew that their little flying reindeer eyes were much bigger than their little flying reindeer tummies, so he set some rules…

Not quite as sophisticated as the five dining philosophers, but it will suffice. We will simulate the feast as follows:



Santa Claus (parent) will get the data filename as a command-line argument. Santa Claus will then create the 9 pipes, 1 semaphore and 1 shared memory (initialized to 0). Santa Claus will then make sure the semaphore is free (unlocked). Santa Claus will then fork & exe the nine little flying reindeer and pass on to them the data filename, the pipe identifiers, semaphore identifier, and the shared memory identifier as command-line arguments.

Note that since elves are not very creative, Santa Claus simply named his little flying reindeer: _p1, _p2, … _p9 (note the underscores in the names).

Note that the data file will just be a file of characters, so you might want to test on your code on something like:

```
Once upon a midnight dreary, fingers cramped and vision bleary,
System manuals piled high and wasted paper on the floor,
Longing for the warmth of bedsheets,
Still I sat there, doing spreadsheets:
Having reached the bottom line,
I took a floppy from the drawer.
Typing with a steady hand, I then invoked the SAVE command
But got instead a reprimand: it read "Abort, Retry, Ignore"
```

Whenever a little flying reindeer attempts to run, it will first check the semaphore and if it is free (unlocked) it will lock the semaphore and continue; otherwise, it must wait until the semaphore is unlocked. A little flying reindeer that gets to run will then generate a random number N between 1 and 10 (inclusive) and read N characters from the file, pass its name (program1, etc) followed by those characters to Santa Claus, and add the value of N to the value in the shared memory.

Santa Claus will extract that data from the pipe, write it to a file (hw9.out), and free the semaphore. Thus, the hw9.out file will have, for example, characters such as:

_ p1hel_p4lo

Once all of the truffles (data) have been consumed, the little flying reindeer who got the last morsel, needs to pass the characters back to Santa Claus, acquire the value from the shared memory and pass it back to Santa Claus (who will then append it to the file), and then somehow signal the rest of the flying reindeer that the truffles (data) are all gone so they can ALL shut down (take a siesta?). Note that this flying reindeer may generate a random number larger than the number of characters left, you will need to watch for EOF to prevent a seg fault. Also, you could encounter the EOF before reading all of the characters, but the number returned to Santa Claus must **always** be equal to the actual number of characters read. Finally, Santa Claus will append onto the output the file the total number of characters read by all of the flying reindeer (which should also equal the number of characters in the file).

So on completion, the hw9.out file might look like:

_p1hel_p4lo5

This way Santa Claus will have a record of who is the biggest pig of the bunch!

WARNING:
These programs must be tested on multiple data sets with multiple runtimes each. If it fails on our test set, well let's just say you will find yourself on Santa's naughty list.

REQUIREMENTS:
1. Your program MUST run on the cscluster (cscluster.cs.und.edu).
2. Your full name MUST appear as a comment at the beginning of your program.
3. Your parent code MUST be named hw9-yourname.cpp
4. The programs MUST compile with g++. fpermissive flag is NOT allowed.
5. Your program MUST use system V semaphores (semget, semop, and semctl).
6. Your code MUST adhere to coding style 1TBS (https://en.wikipedia.org/wiki/Indentation_style)
7. You really only need one child, named p.c, but you can create 9 children (named pN.c or pN.cpp, where N is 1 through 9) if you want. Or, you can simply spawn the 1 child 9 times.
8. You MUST include your source files and you MUST include a Makefile to compile them.
9. Your tarball MUST be named hw9-yourname.tar

!!! If you fail to abide by ANY of the "MUST" requirements your submission may not be graded at all (0 points) !!!