Validate:

- Check argc for 2-8 arguments.
- Check that each argument is 1 character long.
- Check that each argument is an integer(atoi)
  - check for ASCII 0 input separately.
- Use an array with indices 1-7 to check for uniqueness.

Shared memory:

- Allocate shared memory using function from
  - http://www.csl.mtu.edu/cs4411.ck/www/NOTES/process/process.html
- Break down into smaller functions, especially detach for each fork.
- Print function: for loop to iterate through the array in shared mem.
  - Unbuffered print

Fork children

- for loop up to argc
- use if statement so that only parent is spawning children.

Child tasks:

- print shared memory(using shared mem print function above)
- Multiply index
  - sharedMem[index] *= 2
  - sharedMem[index] *= uniqueID
- print shared mem again.
- Detach from shared mem
- exit(0)

Store child information

- 2d array for storing exit codes and PID
  - array indexes will correspond to child unique ID.
  - Wait() in a for loop up to argc, storing each exit code AND PID as they are returned.
- Print child information.
  - Iteratate through the 2d array with a nested for loop?
    - Inefficient…

Cleaning up

- detach shared memory.
- Deallocate shared memory
  - free()

Children: lots of parameters required for each child function, use a CHILD struct.

- Shared memory pointer
- unique ID
- number of children spawned
- child PID
- parent PID