DEBUGGING

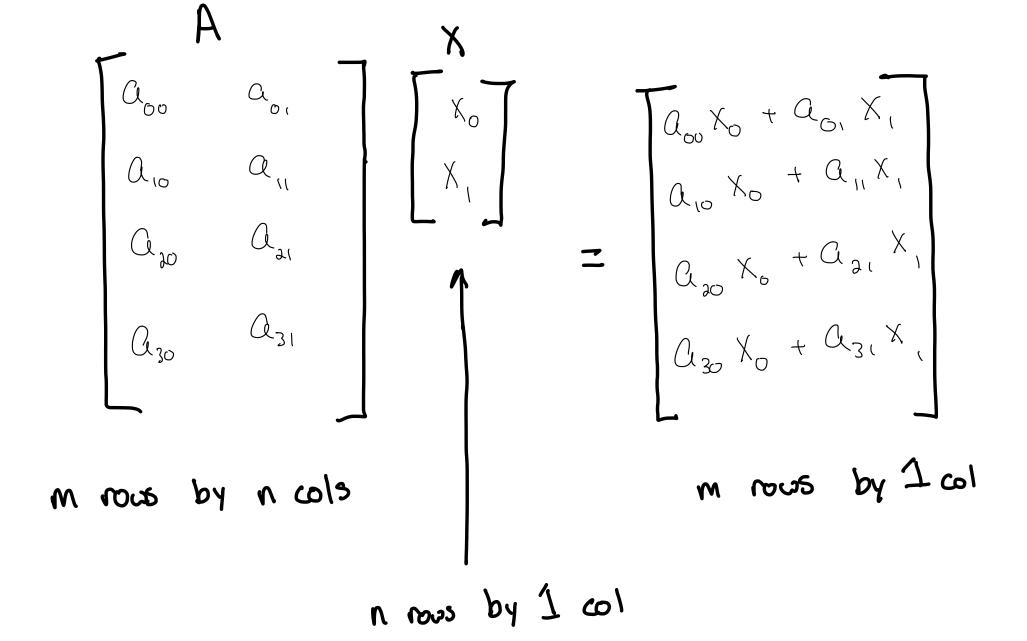
GENERAL DEBUGGING

- gdb = GNU Project debugger
- lets you look at what's going on inside a program as it runs
- helpful for identifying:
 - segfaults
 - logical errors

RUNNING GDB

```
gcc -g program.c
gdb a.out
```

- inside gdb:
 - run start execution form beginning
 - run < input_file > output_file if
 you want to run with input/output redirection
 - kill-stop program execution
 - quit exit gdb



$$\begin{bmatrix} 0 & 2 \\ 4 & 6 \\ 8 & 10 \\ 12 & 14 \end{bmatrix} = \begin{bmatrix} 0*0 + 2*2 \\ 4*0 + 6*2 \\ 8*0 + 10*2 \\ 12*0 + 14*2 \end{bmatrix} = \begin{bmatrix} 2/7 \\ 12 \\ 28 \\ 28 \end{bmatrix}$$

BASIC COMMANDS

- p variable_name print a specific variable
- p *array@len print array with certain length (you specify array and len)
- backtrace how did your program get to this point
 - often helpful after a segfault
 - can also use bt (abbreviated command)
- step execute one line of code
 - step nsteps execute a specified number
 of lines of code

BREAKPOINTS

- Setting breakpoints
 - break function name
 - break line number
 - break line_number if condition
- continue step until next break boint
- info break list all breakpoints

BREAKPOINTS (CONT.)

- Enabling/disabling
 - disable breakpoint number
 - enable breakpoint number
- Deleting breakpoints
 - delete delete all breakpoints
 - delete breakpoint_number delete a
 specific breakpoint

VALGRIND

```
gcc -g program.c
valgrind --leak-check=yes ./a.out
```

- tool used to debug memory leaks
- may be other errors at the top -- don't just ignore them