Welcome!

Today:
➔ Midterm Review
➔ More on Preprocessor

Logistics:
➔ Scores for Midterm 1, and Assignment 1 Style are up
➔ Let us know if mistakes are in the readings. Ex: the online copy of Computer Systems... has different section numbering
➔ Assignment 3 coming soon

Fun fact...

Remember to vote!

Early voting:
Sat & Sun, Feb 17-18

General Voting:
Thurs, Feb 21

go.unc.edu/theo
Midterm Review

Great job!
Midterm Review

Statistics

Raw Scores:
- Best: 25.7/26 (99%)
- Worst: 10.3/26 (40%)
- Avg (mean): 22/26 (86%)

Top incorrect questions:
1. Q. 1.3.2 (79%)
2. Q. 1.1.5 (57%)
3. Q. 1.1.12 (54%)
4. Q. 1.1.1 (47%)
5. Q. 1.1.7 (33%)
6. Q. 1.1.9 (29%)
7. Q. 1.3.5 (28%)

Great job!
Q. 1.1.1: "warning: implicit declaration of function"

Code available at https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l10/q1_demo.c
Q. 1.1.5: string termination

Code available at https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l10/q5_demo.c
Q. 1.1.12: define FALSE as the value 0

Code available at https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l10/q12_demo.c
To find documentation for a C function, I might try the command: (Select all that apply)

A) whatis
B) man
C) info
D) cmatrix

Lecture 3 (and others)
Lecture 7 (and others)
More on the Preprocessor

Beyond \#define and \#include. Picking up from last time...

This section of the slides was not covered in-class, but the midterm question demos sufficiently covered the preprocessor. These examples have been cut from future lectures, but are left here for your study.
```c
#define NV_PCCSR_CHANNEL_INST(i) (0x00800000+(i)*8)

// There are a total of 512 possible channels
#define MAX_CHID 512
typedef union {
    struct {
        // 0:31
        /*...*/ uint32_t inst_ptr:28;
        /*...*/ enum INST_TARGET inst_target:2;
        /*...*/ uint32_t parm;
        /*...*/ bool inst_b1;
        // 32:64
        /*...*/ bool enable;
    } 0
}...```
More on the Preprocessor

nvdebug.h: More Complex

#define NV_RL_ENTRY_SIZE(g) \
   (((g)->chip_id >= NV_CHIP_ID_VOLTA) ? sizeof(struct gtv00_runlist_tsg) : \
   sizeof(struct gk110_runlist_tsg))

#define for_chan_in_tsg(g, chan, tsg) \
   for (chan = (typeof(chan))((u8*)tsg + NV_RL_ENTRY_SIZE(g)); \n   (u8*)chan < ((u8*)tsg) + (1 + tsg_length(g, tsg)) * NV_RL_ENTRY_SIZE(g); \n   chan = (typeof(chan))((u8*)chan) + NV_RL_ENTRY_SIZE(g))

#define next_tsg(g, tsg) \
   (typeof(tsg))((u8*)(tsg) + NV_RL_ENTRY_SIZE(g) * (tsg_length(g, tsg) + 1))

struct runlist_iter {
   ... // Pointer to either
   ... void *curr_entry;
   ... // This should be set
   ... // decremented as
   ... // track which channel
   ... // are available;
};

Code available at
Questions?

See office hour calendar on the website for availability.

Assignment 3 will be posted by next class.

Contact:
Email: hacker@unc.edu
Twitter: @JJBakita
Web: https://cs.unc.edu/~jbakita