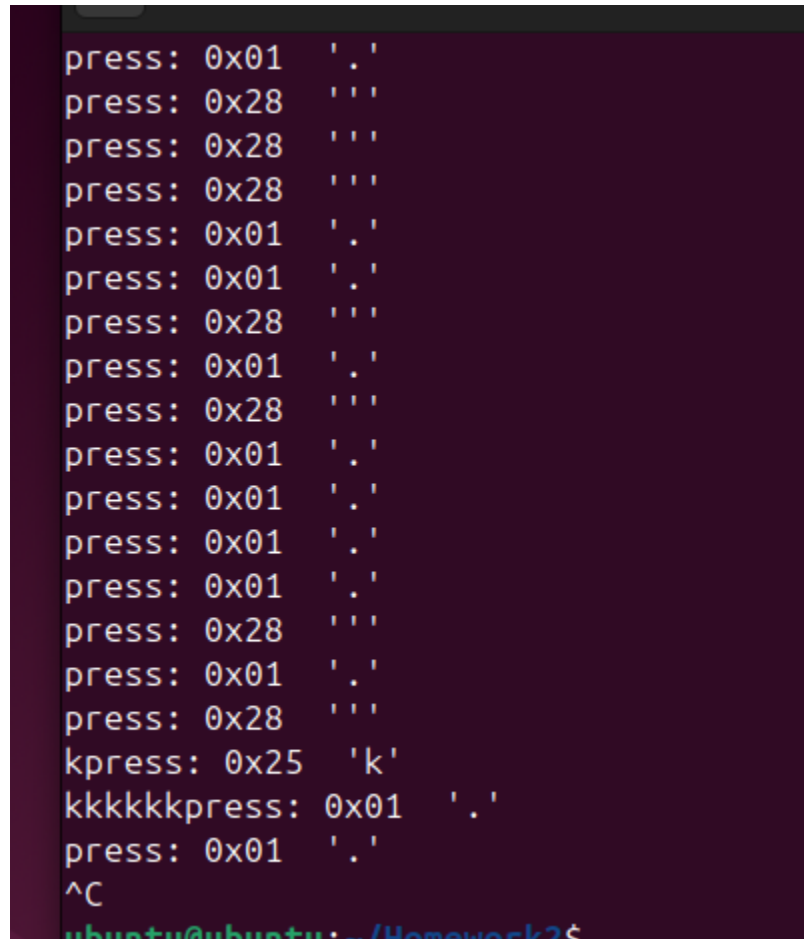


Walter Mikula  
Dr. Klingensmith  
COMP 310 OS  
October 8 2025  
Homework 2:

Deliverables:

I will also submit my .c file I created in my Ubuntu VM

This is the output that i have for my terminal

A terminal window with a dark purple background and light green text. The output shows a sequence of key presses and their corresponding scancodes. The first line is 'press: 0x01 '.''. This is followed by several lines of 'press: 0x28 ''' and 'press: 0x01 '.''. Then, there is a 'kpress: 0x25 'k'' line, followed by 'kkkkkkpress: 0x01 '.'' and 'press: 0x01 '.''. The sequence ends with '^C' and the prompt 'ubuntu@ubuntu:~/Homework2\$'.

Write C code to do the following in your

OS: 1. Read the PS/2 status register and check the LSB.

- In the continuous loop in my code, because I have it polling until Ctrl + C is hit, I call inb(0x64) to read the byte and the LSB.

2. If the LSB is 1, then that means that the PS/2 controller's output buffer contains a scancode.

- In my code, when the LSB ==1, the buffer becomes full, and the scancode can be read

3. To get the scancode, read from IO port 0x60.

- I used an unsigned char to read it from the scancode.

#### 4. Print the scancode to the terminal

- It kept printing a flow of '.'s to the terminal, so I was very confused, but I guess that is normal for new Linux environments since I'm using Ubuntu, not Gentoo.

Question: Is this a good approach? What happens when you have more than one I/O device?

What about if you have some other tasks running that don't involve I/O? Will this approach affect responsiveness?

This approach is pretty shit, but it does work. The CPU constantly checks the PS/2 register, so it's always busy waiting for data. When there is more than one I/O device, then it would have to continually check each I/O device, which would make it super slow and would tank the performance. If other tasks that don't involve I/O are running will not run smoothly because it's still checking for input. And the input might not even exist if we don't enter a key for it to print out, so it will just be checking for nothing. This approach will affect responsiveness in a negative way. Either the I/O polling is running and wasting CPU.