Prelab

Shiyu Wang

Pre 4.1

```
#!/bin/bash

# hello.bash


echo -n "Please tell me your name (first, Last):"

read -r fullname

echo "Hello $fullname"
```

Pre 4.2

This is a program for wiimote that reads values from a button event, event2 and prints the code and the value. First it tries to open the file WIIMOTE_EVT2_FNAME in read only mode, return -1 and print error if fails. Then it applys read system call, “read(int fd, void *buf, size_t count)” to read up to WIIMOTE_EVT2_PKT_SIZE(32) bytes from file into the buffer starting at buf. After that, it just simply keep extracting code, buf[10] and value, buf[12] from the buf and print them out until file ends. Finally, it closes the file descriptor after reading.

Pre4.3

This is a program for wiimote that reads values from event0 and print values of x, y and z. First it tries to open the file `WIIMOTE_EVT0_FNAME` in read only mode, return -1 and print error if fails. Then it did the similar thing like in wiimote_tbns.c to read up to WIIMOTE_EVT0_PKT_SIZE(16) bytes from file into the buffer starting at buf. After that, it keeps extracting code byte, buf[10], high byte of accel, buf[13] and low byte of accel, buf[12] and print them our in order of x (code byte), y (high byte of accel) and z (low byte of accel) until the file ends. Finally it closes the file descriptor after reading

Pre4.4

According to the idea of modular programming, it is better to write less code and wite reusable code. Comparing with “wiimote_btns.c” and “wiimote_accl.c”, there are many similar parts of two program. For example, they are both read from the file and print the code byte and value from it. We can easily create cross file function to apply on both programs to improve modularity.