50-Step Unix

1. man ls

2. apropos git

3. A prompt that asked: "Display all 176 possibilities? (y or n)"

4. pwd

5. Use Ctrl+P or the up arrow key.

6. nano ~/.bash_profile

7. hostname

8. ifconfig

9. dig google.com

10. hostnamectl (then look at the section that said "Operating System").

11. ps aux | wc -l

12. top -o +%MEM

13. top -o -%CPU

14. ps --no-headers

15. Ctrl+C

16. python -c 'while True: 1*1' &
    Then type: ps aux | grep python

17. ps aux | grep -E 'python.*while' | grep -v grep

18. pkill -f "python.*while"

19. whoami

20. groups or groups hongkng5 (hongkng5 is my username)

21. echo $SHELL

22. touch funny

23. ls -l funny

24. chmod u=rwx,g=rx,o=r funny

25. echo "ls -al /" >> funny

26. ./funny

27. echo "sleep 10" >> funny

28. cp -p funny not-so-funny

29. chmod o=x not-so-funny

30. chmod a=x not-so-funny
    rm -f not-so-funny

31. scp funny hongkng5@csslab9.uwb.edu:/tmp/funny-hongkng5

32. ssh csslab9.uwb.edu "ls /tmp"

33. ssh csslab9.uwb.edu "/tmp/funny-hongkng5"

34. find /usr/bin -type f -name '*zip*' | wc -l

35. find /usr/bin -name '*yum*'

36. mkdir yummies

37. cp /usr/bin/*yum* yummies

38. zip my-yummies.zip yummies

39. tar -czvf my-yummies.tar.gz yummies

40. for i in {1..3}; do
        cat my-yummies.tar.gz >> big-file
    done [?]

41. ls -lh big-file

42. wget
https://gist.githubusercontent.com/jclosure/63820f48cb6551a43a0b272c3542389f/raw/093
eff95fcfc6c3c547d9854ebb9a027bb285558/.nanorc

43. git clone https://github.com/pisan343/hello

44. cd hello
    then type: ./simplecompile.sh

45. There is no memory leak in main.cpp

46. cat main.cpp

47. less main.cpp

48. time sleep 1

49. which ls

50. This is funny because when you try to demand or install anything from a different OS, it'll make you go and do it yourself. While when you run sudo make in Unix, the OS will go and find whatever it is and do that for you. You can just sit back and relax while the OS will do all the work and won't even complain.

RUN A...   (gdb) Laun   main.cpp

main.cpp > printFun(int)

```cpp
1    // A C++ program to demonstrate working of
2    // recursion
3    #include <bits/stdc++.h>
4    using namespace std;
5
6    void printFun(int test)
7    {
8        if (test < 1)
9            return;
10       else {
11           cout << test << " ";
12           printFun(test - 1); // statement 2
13           cout << test << " ";
14           return;
15       }
16   }
17
18   // Driver Code
19   int main()
20   {
21       int test = 3;
22       printFun(test);
```

CALL STACK   Paused on breakpoint
    printFun(int test)   main.cpp
    main()   main.cpp   22:1

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

[cssuwbstudent@LinuxMachine printfun]$

bash printfun
bash print...
cppdbg: ...