

----- LINEAR -----

● Output

```

523.694957] Removing Module
526.925998] Loading Module
526.926001] pid: 1 |pname: systemd |state: 1
526.926003] pid: 2 |pname: kthreadd |state: 1
526.926004] pid: 3 |pname: rcu_gp |state: 1026
526.926006] pid: 4 |pname: rcu_par_gp |state: 1026
526.926007] pid: 6 |pname: kworker/0:0H |state: 1026
526.926008] pid: 8 |pname: kworker/u256:0 |state: 1026
526.926009] pid: 9 |pname: mm_percpu_wq |state: 1026
526.926010] pid: 10 |pname: ksoftirqd/0 |state: 1
526.926011] pid: 11 |pname: rcu_sched |state: 1026
526.926012] pid: 12 |pname: migration/0 |state: 1
526.926013] pid: 13 |pname: idle_inject/0 |state: 1
526.926015] pid: 14 |pname: cpuhp/0 |state: 1
526.926016] pid: 15 |pname: kdevtmpfs |state: 1
526.926017] pid: 16 |pname: netns |state: 1026
526.926018] pid: 17 |pname: rcu_tasks_kthre |state: 1
526.926019] pid: 18 |pname: kauditd |state: 1
526.926020] pid: 19 |pname: khungtaskd |state: 1
526.926021] pid: 20 |pname: oom_reaper |state: 1
526.926023] pid: 21 |pname: writeback |state: 1026
526.926024] pid: 22 |pname: kcompactd0 |state: 1
526.926024] pid: 23 |pname: ksmd |state: 1
526.926026] pid: 24 |pname: khugepaged |state: 1
526.926027] pid: 116 |pname: kintegrityd |state: 1026
526.926028] pid: 117 |pname: kblockd |state: 1026
526.926029] pid: 118 |pname: blkcg_punt_blo |state: 1026
526.926031] pid: 119 |pname: tpm_dev_wq |state: 1026
526.926032] pid: 120 |pname: ata_sff |state: 1026
526.926033] pid: 121 |pname: md |state: 1026
526.926034] pid: 122 |pname: edac-poller |state: 1026
526.926035] pid: 123 |pname: devfreq_wq |state: 1026
526.926036] pid: 124 |pname: watchdogd |state: 1
526.926038] pid: 127 |pname: kswapd0 |state: 1
526.926039] pid: 128 |pname: kworker/u257:0 |state: 1026
526.926040] pid: 129 |pname: ecryptfs-kthrea |state: 1
526.926041] pid: 132 |pname: kthrotld |state: 1026
526.926042] pid: 133 |pname: irq/24-pciehp |state: 1
526.926043] pid: 134 |pname: irq/25-pciehp |state: 1
526.926044] pid: 135 |pname: irq/26-pciehp |state: 1
526.926045] pid: 136 |pname: irq/27-pciehp |state: 1
526.926046] pid: 137 |pname: irq/28-pciehp |state: 1
526.926047] pid: 138 |pname: irq/29-pciehp |state: 1
526.926048] pid: 139 |pname: irq/30-pciehp |state: 1
526.926048] pid: 140 |pname: irq/31-pciehp |state: 1
526.926049] pid: 141 |pname: irq/32-pciehp |state: 1

```

● Code

```

#include <linux/sched/signal.h>
#include <linux/sched.h>
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/module.h>

/* This function is called when the module is loaded. */
int entryPoint(void) {
    struct task_struct *task;

    printk(KERN_INFO "Loading Module\n");

    for_each_process(task) {
        printk(KERN_INFO "pid: %d |pname: %s |state: %d\n", task->pid, task->comm, task->state);
    }

    printk(KERN_INFO "-----END-----\n");
    return 0;
}

/* This function is called when the module is removed. */
void exitPoint(void) {
    printk(KERN_INFO "Removing Module\n");
}

/* Macros for registering module entry and exit points. */
module_init(entryPoint);
module_exit(exitPoint);

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("List tasks linearly");
MODULE_AUTHOR("105072123");

```

首先 include 所有會用到的 header 檔，然後建立一個<linux/sched.h>內建的 structure task_struct *task，作為要被印出 info 的 task 的 pointer，包含 prev 和 next。

再用<linux/sched/signal.h>內建的 for_each_process 這個 macro 依序拜訪所有 process。task->pid 是 task 的 process ID，task->comm 是 task 的 name，task->state 是 task 的 state，而-1 表示 unrunnable，0 表示 runnable，>0 表示 stopped。

DFS

● Output

```
5040.261028] pid: 1996 | pname: gsd-color | state: 1
5040.261029] pid: 1997 | pname: gsd-datetime | state: 1
5040.261030] pid: 2001 | pname: gsd-housekeepin | state: 1
5040.261031] pid: 2002 | pname: gsd-keyboard | state: 1
5040.261033] pid: 2004 | pname: gsd-media-keys | state: 1
5040.261034] pid: 2005 | pname: gsd-mouse | state: 1
5040.261035] pid: 2070 | pname: gsd-disk-utilit | state: 1
5040.261036] pid: 2076 | pname: nautilus-deskto | state: 1
5040.261037] pid: 2324 | pname: gnome-software | state: 1
5040.261039] pid: 2326 | pname: update-notifier | state: 1
5040.261040] pid: 3360 | pname: deja-dup-monito | state: 1
5040.261041] pid: 795 | pname: systemd | state: 1
5040.261042] pid: 797 | pname: (sd-pam) | state: 1
5040.261043] pid: 810 | pname: dbus-daemon | state: 1
5040.261045] pid: 1023 | pname: at-spi-bus-laun | state: 1
5040.261046] pid: 1033 | pname: dbus-daemon | state: 1
5040.261047] pid: 1040 | pname: at-spi2-registr | state: 1
5040.261048] pid: 1126 | pname: pulseaudio | state: 1
5040.261049] pid: 1287 | pname: ibus-portal | state: 1
5040.261050] pid: 1299 | pname: xdg-permission- | state: 1
5040.261052] pid: 876 | pname: upowerd | state: 1
5040.261053] pid: 977 | pname: whoopsie | state: 1
5040.261054] pid: 983 | pname: kerneloops | state: 1
5040.261055] pid: 994 | pname: kerneloops | state: 1
5040.261056] pid: 1144 | pname: rtkit-daemon | state: 1
5040.261058] pid: 1285 | pname: ibus-x11 | state: 1
5040.261059] pid: 1430 | pname: bolt | state: 1
5040.261060] pid: 1441 | pname: packagekitd | state: 1
5040.261061] pid: 1627 | pname: colord | state: 1
5040.261062] pid: 1655 | pname: systemd | state: 1
5040.261063] pid: 1656 | pname: (sd-pam) | state: 1
5040.261065] pid: 1680 | pname: dbus-daemon | state: 1
5040.261066] pid: 1781 | pname: at-spi-bus-laun | state: 1
5040.261067] pid: 1786 | pname: dbus-daemon | state: 1
5040.261068] pid: 1789 | pname: at-spi2-registr | state: 1
5040.261069] pid: 1812 | pname: gvfsd | state: 1
5040.261070] pid: 2096 | pname: gvfsd-trash | state: 1
5040.261072] pid: 1817 | pname: gvfsd-fuse | state: 1
5040.261073] pid: 1846 | pname: ibus-portal | state: 1
5040.261074] pid: 1854 | pname: xdg-permission- | state: 1
5040.261075] pid: 1904 | pname: evolution-sourc | state: 1
5040.261076] pid: 1888 | pname: gnome-shell-cal | state: 1
5040.261077] pid: 1929 | pname: dconf-service | state: 1
5040.261079] pid: 1932 | pname: gvfs-udisks2-vo | state: 1
5040.261080] pid: 1920 | pname: goa-daemon | state: 1
5040.261081] pid: 1943 | pname: gvfs-goa-volume | state: 1
```

```
0 S 121 1519 812 0 80 0 - 76392 - tty1 00:00:00 gsd-sharing
0 S 121 1522 812 0 80 0 - 94564 - tty1 00:00:00 gsd-smartcar
0 S 121 1526 812 0 80 0 - 83297 - tty1 00:00:00 gsd-sound
0 S 121 1532 812 0 80 0 - 147539 - tty1 00:00:00 gsd-wacom
0 S 121 1589 1265 0 80 0 - 51289 - tty1 00:00:00 ibus-engine-
4 S 117 1627 1 0 80 0 - 81307 - ? 00:00:00 colord
4 S 0 1652 776 0 80 0 - 103901 - ? 00:00:00 gdm-session-
4 S 1000 1655 1 0 80 0 - 19266 ep_pol ? 00:00:00 systemd
5 S 1000 1656 1655 0 80 0 - 48978 - ? 00:00:00 (sd-pam)
1 S 1000 1669 1 0 80 0 - 72168 - ? 00:00:00 gnome-keyrin
4 S 1000 1673 1652 0 80 0 - 53071 poll_s tty2 00:00:00 gdm-x-sessio
4 S 1000 1675 1673 0 80 0 - 119049 ep_pol tty2 00:00:36 Xorg
0 S 1000 1680 1655 0 80 0 - 12839 ep_pol ? 00:00:00 dbus-daemon
0 S 1000 1684 1673 0 80 0 - 158439 poll_s tty2 00:00:00 gnome-sessio
1 S 1000 1779 1684 0 80 0 - 2825 - ? 00:00:00 ssh-agent
0 S 1000 1781 1655 0 80 0 - 87323 poll_s ? 00:00:00 at-spi-bus-l
0 S 1000 1786 1781 0 80 0 - 12480 ep_pol ? 00:00:00 dbus-daemon
0 S 1000 1789 1655 0 80 0 - 55177 poll_s ? 00:00:00 at-spi2-regi
0 S 1000 1806 1684 1 80 0 - 747089 poll_s tty2 00:01:22 gnome-shell
0 S 1000 1812 1655 0 80 0 - 73037 poll_s ? 00:00:00 gvfsd
0 S 1000 1817 1655 0 80 0 - 104028 futex ? 00:00:00 gvfsd-fuse
1 S 1000 1828 1 0 69 -11 - 293075 poll_s ? 00:00:23 pulseaudio
0 S 1000 1838 1806 0 80 0 - 90468 poll_s tty2 00:00:07 ibus-daemon
```

- Code

```
#include <linux/sched/task.h>
#include <linux/sched.h>
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/module.h>

void dfs(struct task_struct *task) {
    struct task_struct *child;
    struct list_head *list;

    printk(KERN_INFO "pid: %d | pname: %s | state: %d\n", task->pid, task->comm, task->state);

    list_for_each(list, &task->children) {
        child = list_entry(list, struct task_struct, sibling);
        dfs(child);
    }
}

/* This function is called when the module is loaded. */
int entryPoint(void) {
    printk(KERN_INFO "Loading Module\n");

    dfs(&init_task);

    printk(KERN_INFO "-----END-----\n");
    return 0;
}

/* This function is called when the module is removed. */
void exitPoint(void) {
    printk(KERN_INFO "Removing Module\n");
}

/* Macros for registering module entry and exit points. */
module_init(entryPoint);
module_exit(exitPoint);

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("List tasks using DFS");
MODULE_AUTHOR("105072123");
```

首先一樣先 include 所有會用到的 header 檔，過程中我發現 init_task 從 linux4.11 起，是在<linux/sched/task.h>中聲明，而非<linux/sched.h>中。所以如果只有 include<linux/sched.h>的話，會產生”init_task undeclared”的 error。

再呼叫 function dfs，然後跟第一題同樣建立一個 linux 內建的 structure task_struct *task，傳入 init_task。init_task 就是 kernel 中 process 0 使用的 process 描述符，也是 Linux 系統中第一個 process 描述符。

在 fuction dfs 中，首先建立一個 struct task_struct *child，作為 next child 的 pointer。再建立一個 struct list_head *list，list_head 在上一次的作業中就有使用過了，是用來儲存 prev 和 next 的 list。然後 printk 的部分就跟第一題是一模一樣的。

再用 list_for_each 來 traverse task 的 list，list 為一個 pointer 並指向第一個節點”task 的 children”，以此為起點一路 traverse 下去，終止條件是當 pointer 指向 head 時，代表已經 traverse 完一圈了。然後在這個 loop 中使用 list_entry 找到 task 的 children 的 sibling，將它們設為 child，最後傳入 dfs 中再次呼叫的 dfs fuction，不斷 iterate。