```c
/* ANRC RHKI */
/* Lab13b: Virtual Filesystem Lab */
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/unistd.h>
#include<linux/fs.h>

#include <linux/proc_fs.h>      /* Necessary because we use proc fs */
#include <linux/uaccess.h>      /* for copy_*_user */
#include <linux/seq_file.h>             /* use new seq file */

#define PROCFS_MAX_SIZE         1024
#define PROCFS_NAME             "anrc_vfs"

#define DRIVER_AUTHOR "ANRC"
#define DRIVER_DESC   "Lab13b"

MODULE_LICENSE("GPL");           // Get rid of taint message by declaring code as GPL.

/*  Or with defines, like this: */
MODULE_AUTHOR(DRIVER_AUTHOR);    // Who wrote this module?
MODULE_DESCRIPTION(DRIVER_DESC); // What does this module do?


static char msg[PROCFS_MAX_SIZE];
int len = 0;

static int read_proc(struct seq_file *m, void *v)
{
        seq_printf(m, "%s", msg);
        return 0;
}

static int open_proc(struct inode *inode, struct file *file)
{
        return single_open(file, read_proc, NULL);
}

static int write_proc(struct file *filp,char *buf,size_t count,loff_t *offp )
{

        len = count;
        memset(msg, 0x0, PROCFS_MAX_SIZE);

        /* copy_from_user(to, from, n) */
        /* don't forget to check bounds */

        return len;
}

static const struct file_operations proc_fops = {
                .owner = THIS_MODULE,
                .read = seq_read,
                .open = open_proc,
                .write = write_proc,
};


void create_new_proc_entry()
{
        /* need read and write for root only, read for others */
        proc_create(PROCFS_NAME, 0644, NULL, &proc_fops);
        memset(msg, 0x0, PROCFS_MAX_SIZE);
        memcpy(msg, "Hello World", 12);
        len = strlen(msg);
        printk(KERN_INFO "proc_create msg:%s",msg);
```

```
}


int proc_init (void)
{
        create_new_proc_entry();
        return 0;
}

void proc_cleanup(void)
{
        remove_proc_entry(PROCFS_NAME, NULL);
        printk(KERN_INFO "/proc/%s removed\n", PROCFS_NAME);
}

module_init(proc_init);
module_exit(proc_cleanup);
```