ECE 331

Homework 2

See course web site for due date

Place your typed homework answers in vim. Print single sided with your name using a **mono space font**. No need to restate questions. Fully investigating questions is required for a higher grade. Please use the kernel coding style for all code. Please use your RPi for developing answers. Although code should be written and run on a RPi, it should run on ANY POSIX compliant OS. As always, all code shall be comment, conform to the Linux Kernel Coding Style, and error conditions shall be checked and appropriately handled.

1.
   a) apt-get update; apt-cache search weather
   b) apt-get install weather-util
   c) dpkg -L weather-util
   d) apt-get remove weather-util

2. Give the concise command(s) to
   a) apt-get source avrdude
   b) wget 'http://savannah.nongnu.org/bugs/download.php?file_id=32171'; mv 'download.php?file_id=32171' endpoint.patch
   c) cd avrdude-6.2; patch < ../endpoint.patch
   d) apt-get build-dep avrdude
   e) dpkg-buildpackage -uc -us; dpkg-source --commit
   f) dpkg-buildpackage -uc -us
   g) cd ..; dpkg -i avrdude_6.2-5_arm.deb
   h) patch name: endpoint patch
      Description: Fix for LUFA code
      Summary: AVRdude does not detect LUFA devices due to an endpoint error

3. ln -s /sys/var/adm/armv7/hf skywalker

4.
```
// A. Sheaff 2/3/2017
// A program to sum file sizes using the block count
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <stdint.h>

int main(int argc, char * argv[])
{
    unsigned int i;     // Counter
    blkcnt_t sum=0;     // Sum
    struct stat s;      // Stat buffer for count
    int ret;            // Error checking

    // For each command line arg
    for (i=1;i<argc;i++) {
        ret=stat(argv[i],&s);      // Get the inode metadata
        if (ret<0) {               // Check for errors
            perror("stat");
            continue;
        }
        printf("%s %lu\n",argv[i],s.st_blocks);      // Print this file
        sum+=s.st_blocks;                            // Sum
    }

    printf("%lu\n",sum);                             // Total

    return 0;
}
```

5. On your RPi, write a C program that prints the total number of BLOCKS used by all filenames that match a **single glob** passed on the command line. Have **your** program do the globbing (not the shell) by calling `glob(3)` in your program. As always, check for and act appropriately upon error.
   a) `./gbsum ‘/usr/local/images/*.jpg’`
   b)

```
// A. Sheaff 2/2/2017
// Block size sum program modified to use a glob passed on cmd line
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <stdint.h>
#include <glob.h>

int main(int argc, char * argv[])
{
    unsigned int i;     // Count
    blkcnt_t sum=0;     // Block Sum
    struct stat s;      // Inode metadata
    int ret;            // Error checking
    glob_t pglob;       // Glob return

    // Check for 1 arg - namely glob pattern
    if (argc!=2) {
        printf("Usage: %s glob\n",argv[0]);
        return 1;
    }

    // Glob the passed arg
    ret=glob(argv[1],0,NULL,&pglob);
    if (ret<0) {
        perror("glob");
        return errno;
    }

    // Iterate over globs
    for (i=0;i<pglob.gl_pathc;i++) {
        ret=stat(pglob.gl_pathv[i],&s);  // Get metadata
        if (ret<0) {
            perror("stat");
            continue;
        }
        printf("%s %lu\n",pglob.gl_pathv[i],s.st_blocks);  // Print each
        sum+=s.st_blocks;    // Sum
    }
    // Print total
    printf("%lu\n",sum);
    globfree(&pglob);   // Clean up
    return 0;
}
```

6.

a) `debugfs /dev/mmblk0p2`
b) `>> stats # take the last group number and add one. 238`
c) `>> stat /bin/ls`
d) `>> ncheck 11`