Operating Systems 2016/17
Assignment 12

Prof. Dr. Frank Bellosa
Dipl.-Inform. Marc Rittinghaus

Submission Deadline: Monday, January 30th, 2017 – 23:59

A new assignment will be published every week, right after the last one was due. It must be completed before its submission deadline.

The assignments must be filled out online in ILIAS. Handwritten solutions are no longer accepted. You will find the online version for each assignment in your tutorial's directory. **P-Questions** are programming assignments. Download the provided template from ILIAS. Do not fiddle with the compiler flags. Submission instructions can be found on the first assignment.

In this assignment you will delve into storage as well as file and directory access.

T-Question 12.1: Storage

a. Name two reasons why the OS has only limited information about the layout of data on hard disks (with regard to locality). **1 T-pt**

b. Why are flash-based solid-state drives (SSD) much faster than hard disks for random access? **1 T-pt**

c. What technique should file systems on SSDs apply to help with write performance? How does that mechanism help the SSD to reduce write overhead? **2 T-pt**

d. Why is a RAID 5 system often preferred over a RAID 4 system? **1 T-pt**

T-Question 12.2: Files

a. Sparse files have unallocated holes which are represented by zeros. When a write is performed to an offset in a hole, a new block is allocated that can hold the written data. How can you determine if a file is a sparse file (i.e., it has holes), using the `stat` system call? *Hint:* Take a look at the man-page and the fields in `struct stat`. **1 T-pt**

b. What are ACLs? **1 T-pt**

c. UNIX defines the **r**ead, **w**rite, and e**x**ecute permissions. What other permissions on files could be of use? Name at least two. **1 T-pt**

P-Question 12.1: Directory Listing

Download the template **p1** for this assignment from ILIAS. You may only modify and upload the file `ls.c`.

In this question you will write a program that prints a directory listing just like the 'ls' and 'dir' commands in Linux and Windows respectively.

a. Write a function that prints a listing of all files in the directory specified by `path`. Ignore the `filterByExtension` parameter for now. Your function should fulfill the following requirements: **3 P-pt**

- Performs an enumeration of all files in the given directory using the `opendir()`, `readdir()`, and `closedir()` system calls.
- For each file, retrieves the size in bytes and the actual size on disk in bytes using `stat()`.
- Prints the gathered information via the template's `_printLine()` function, using the file name without the path (e.g., 'filename.txt' instead of '/home/x/filename.txt').
- Returns 0 on success, -1 otherwise.

```
int list(const char* path, const char *filterByExtension);
```

b. Extend your listing program so that it only displays files with the extension supplied by `filterByExtension` (e.g., 'txt'). Assume extensions to be separated from the file name by a dot (e.g., '/path/filename.txt'). If `filterByExtension` is `NULL`, do not perform extension filtering. However, your program should never show hidden files (name starts with '.') and the entries for the current '.' and the parent directory '..'. **2 P-pt**

P-Question 12.2: File Copy

Download the template **p2** for this assignment from ILIAS. You may only modify and upload the file `copy.c`.

In this question you will write a program that copies a file.

a. Your program should be able to receive its arguments (e.g., the file to copy) via the command line. Write a function that parses the command line and initializes the supplied `CopyArgs` structure with the extracted parameters. Your function should fulfill the following requirements: **2 P-pt**

- Accepts command lines in the form `'-s n -c n src dest'`, where `n` is a positive integer and `src`, and `dest` are the source and destination file names, respectively. The parameter `-s` (skip) is optional and supplies a number of bytes to skip at the beginning of the file. It has a default value of 0, meaning that the file should be copied from the beginning. The parameter `-c` (count) is optional and specifies the number of bytes to copy. Its default value is -1, indicating that the whole file should be copied.
- Uses `getopt()` to parse the command lines arguments.
- Updates the supplied `CopyArgs` structure to reflect the input.
- Returns 0 on success, -1 otherwise.

```
        int parseCopyArgs(int argc, char * const argv[],
            CopyArgs *args);
```

b. Write a function that performs the actual file copy. Your function should fulfill the following requirements: **3 P-pt**

- Performs the copy using the `open()`, `lseek`, `read()`, `write()`, and `close()` system calls.
- Fails if the destination file already exists.
- Uses `buffer` as intermediate buffer.
- Returns 0 on success, -1 otherwise.

Hints: You can assume that no operations on the source file are performed during copying.

```
        int doCopy(CopyArgs *args);
```

Total:
8 T-pt
10 P-pt