

## Solution of Exercise Sheet 6

### Exercise 1 (File Systems)

1. Describe the information inodes store.

*An inode stores a file's metadata, except the file name.*

2. Name three examples of metadata in the file system.

*Metadata are among others the size, UID/GID, permissions and date.*

3. Describe what a cluster in the file system is.

*File systems address clusters and not blocks of the storage device. Each file occupies an integer number of clusters.*

4. Describe how a UNIX file system (e.g. ext2/3), which does not implement extents, can address more than 12 clusters.

*If a file requires more than 12 clusters, these clusters are indirectly addressed via clusters that contain only cluster numbers.*

5. Describe how directories in the Linux file systems are technically implemented.

*Directories are just text files, which contain the names and inodes of files.*

6. Explain why moving a large file within a file system is always faster than copying it.

*Directories are just text files containing the names and inode numbers of files. Moving a file within a file system implies removing a line in the old directory and adding a line in the new directory. On the other hand, copying a file creates a new inode and duplicates the file contents.*

7. Name one advantage and one drawback of small clusters in the file system compared with large clusters.

*Advantage: Decreasing capacity loss due to internal fragmentation.*

*Drawback: Rising overhead for large files.*

8. Do DOS/Windows file systems differentiate between uppercase and lowercase?

☐ Yes      ☒ No

9. Do UNIX file systems differentiate between uppercase and lowercase?

☒ Yes      ☐ No

10. Do modern operating systems accelerate requests to stored data with a cache in the main memory.

☒ Yes      ☐ No

11. Most operating systems operate according to the principle...

☒ write-back      ☐ write-through

12. Name one advantage and one disadvantage of a cache in the main memory, which is used by the operating system to accelerate the requests to stored data.

*Benefit: Better system performance.*

*Drawback: System crashes may cause inconsistencies.*

13. Explain what an absolute path name is.

*It is a path name, which describes the complete path from the root to the file.*

14. Explain what a relative path name is.

*It is a path name, which does not begin with the root.*

15. `/var/log/messages` is an/a...

☒ absolute path name      ☐ relative path name

16. `BTS_Vorlesung/Vorlesung_05/bts_slides_05_en.tex` is an/a...

☐ absolute path name      ☒ relative path name

17. `Documents/MasterThesis/thesis.tex` is an/a...

☐ absolute path name      ☒ relative path name

18. `/home/<username>/Mail/inbox/` is an/a...

☒ absolute path name      ☐ relative path name

19. Describe what information the boot sector (also called boot block) of a file system stores.

*The boot sector (boot block) contains executable machine code („boot loader“), which starts the operating system, and in some file systems like FAT12/16/32 some information about the file system too.*

20. Describe what information the super block of a file system stores.

*It contains information about the file system, e.g. number of inodes and clusters.*

21. Explain why some file systems (e.g. ext2/3) do combine the clusters of the file system to block groups.

*Inodes (metadata) are physically located close to the clusters, they address.*

22. Describe what the File Allocation Table (FAT) is and the information it stores.

*The FAT (File Allocation Table) is a table of fixed size. For each cluster in the file system, an entry exists in the FAT with the following information about the cluster:*

- Cluster is free or the storage medium is damaged at this point.*
- Cluster is occupied by a file and stores the address of the next cluster, which belongs to the file or it is the last cluster of the file.*

23. Describe the objective of the journal in a journaling file system.

*In the journal, write operations are collected before being committed to the file system.*

24. Describe a benefit of using a journaling file system compared with using a file system without a journal.

*After a crash, only the files (clusters) and metadata must be checked, for which a record exists in the journal.*

25. Name the three values that are required to store an extent.

*Start (cluster number) of the area (extent) in the file.*

*Size of the area in the file (in clusters).*

*Number of the first cluster on the storage device.*

26. Describe the benefit of using extents compared with direct addressing of the clusters.

*Instead of multiple individual clusters numbers, only 3 values are required: Lesser overhead.*

27. Name one Linux file system that implements extents.

*JFS, XFS, btrfs, ext4*

28. Name one Windows file system that implements extents.

*NTFS*

29. When a file within a copy-on-write (COW) file system gets modified, the old clusters in the file system that need to be modified are...

- ☐ are overwritten with the new modifications.
- ☒ are preserved (not changed).
- ☒ are copied into new clusters, where the modifications are made.
- ☐ are erased, by removing the cluster address in the inode.

(Hint: Two answers are correct!)

30. Describe the result of defragmenting a file system.

*Logically related clusters are placed physically close to each other on the storage device.*

31. Describe the sort of data processing that is maximum accelerated by defragmenting.

*A continuous arrangement would maximum accelerate continuous forward reading of data because no more seek times occur (when using Hard Disk Drives).*

32. Describe the scenario where defragmenting is useful.

*Only if the seek times are huge, defragmentation makes sense.*

## Exercise 2 (File Systems)

Please mark for each statement about file systems, whether the statement is true or false.

Statement	true	false
Inodes store all metadata of files.		X
File systems address clusters and not blocks of the storage medium or storage drive.	X	
The smaller the clusters are, the more overhead for large files occur.	X	
The bigger the clusters are, the lesser capacity is lost due to internal fragmentation.		X
In UNIX, file extensions have always been of great significance.		X
Modern file systems operate so much efficient that buffering by the operating system is no longer common.		X
Absolute path names describe the complete path from the root to the file.	X	
The separator in path names is identical for all operating systems.		X
An advantage of block groups is that the inodes are physically located close to the clusters, they address.	X	
For each cluster in the file system, an entry exists in the FAT.	X	
Because of the Master File Table in NTFS, fragmentation cannot occur.		X
The journal of journaling file systems reduces the number of write operations.		X
Journaling file systems narrow down the data, which need to be checked during the consistency check.	X	
When using journaling file systems, a loss of data is impossible.		X
If metadata and file contents are journaled both, all write operations are carried out twice.	X	
Extents cause lesser overhead compared with block addressing.	X	

## Exercise 3 (Pattern Comparison and Data Analysis)

1. Name (or describe) one useful application for the command `sed`.

*This tool is used for the parsing and transforming of text.*

2. Create a file `sedtest.txt` with the following content:

Line 1  
Line 2  
Line 3  
Line 4  
Line 5  
Line 6

```
$ echo -e "Line 1\nLine 2\nLine 3\nLine 4\nLine 5\nLine 6"  
> ~/sedtest.txt
```

Insert with `sed` 3 blanks at the beginning of each line.

```
$ sed "s/^/   /" ~/sedtest.txt
```

3. Print out with `sed` the lines 2 to 5 of the file `sedtest.txt`.

```
$ sed -n "2,5p" ~/sedtest.txt
```

4. Remove with `sed` each second line of the file `sedtest.txt`.

```
$ sed "n;d;" ~/sedtest.txt
```

5. Create a file `htmlcode.txt` with the following content:

```
<a href="BTSWS2019/index.html">Operating Systems (OpSys)</a><p>
<b>This is a <i>HTML file</i></b><br>
<h2>This is a headline</h2>
```

```
$ echo -e "<a href='BTSWS2019/index.html'>Operating Systems (OpSys)</a><p>" >> ~/htmlcode.html
$ echo -e "<b>This is a <i>HTML file</i></b><br>" >> ~/htmlcode.html
$ echo -e "<h2>This is a headline</h2>" >> ~/htmlcode.html
```

Remove with `sed` all HTML tags from the file `htmlcode.html`.

```
$ sed -e "s/<[^>]*>//g" ~/htmlcode.html
```

6. Create a file `umlaute.txt` with the following content:

```
Bäume, Äpfel, Bücher, Übertreibung
Töpfe, Öffentlichkeit, Straße, Spaß
```

```
$ echo "Bäume, Äpfel, Bücher, Übertreibung" >> ~/umlaute.txt
$ echo "Töpfe, Öffentlichkeit, Straße, Spaß" >> ~/umlaute.txt
```

Modify with `sed` all umlauts in the file `umlaute.txt` into „ae“, „oe“, „ue“, „Ae“, „Oe“, „Ue“ and „ss“.

```
$ sed -e "s/ä/ae/g;s/Ä/Ae/g;s/ö/oe/g;s/Ö/Oe/g;s/ü/ue/g;s/Ü/Ue/g;s/ß/ss/g" ~/umlaute.txt
```

7. Create a file `bundesliga_08_0405.txt` with the results of the 8th match day of the season 2004/2005:

Schalke	- Bochum	3 : 2	61500 spectators
Bielefeld	- Stuttgart	0 : 2	22700 spectators
Dortmund	- Nürnberg	2 : 2	73500 spectators
Leverkusen	- Hamburg	3 : 0	22500 spectators
Freiburg	- Mainz	1 : 2	24000 spectators
Kaiserslautern	- Berlin	0 : 2	30500 spectators
Wolfsburg	- Mönchengladbach	2 : 1	26500 spectators
Rostock	- Hannover	1 : 3	16500 spectators
Bremen	- München	1 : 2	42000 spectators

```
$ echo -e "Schalke      - Bochum      3 : 2 61500 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Bielefeld    - Stuttgart   0 : 2 22700 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Dortmund     - Nürnberg   2 : 2 73500 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Leverkusen   - Hamburg    3 : 0 22500 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Freiburg     - Mainz      1 : 2 24000 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Kaiserslautern - Berlin     0 : 2 30500 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Wolfsburg    - Mönchengladbach 2 : 1 26500 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Rostock      - Hannover    1 : 3 16500 spectators" >> ~/bundesliga_08_0405.txt
$ echo -e "Bremen      - München     1 : 2 42000 spectators" >> ~/bundesliga_08_0405.txt
```

8. Name (or describe) one useful application for the command `awk`.

*This tool is used for text processing and data extraction and reporting.*

9. Determine with `awk` all matches, which had more than 35000 spectators.

```
$ awk '$7 > 35000' ~/bundesliga_08_0405.txt
Schalke - Bochum 3 : 2 61500 spectators
Dortmund - Nürnberg 2 : 2 73500 spectators
Bremen - München 1 : 2 42000 spectators
```

10. Determine with `awk` all matches, which had less than 50000 spectators and where the home team won.

```
$ awk '$4 > $6 && $7 < 50000' ~/bundesliga_08_0405.txt
Leverkusen - Hamburg 3 : 0 22500 spectators
Wolfsburg - Mönchengladbach 2 : 1 26500 spectators
```

11. Determine with `awk` for each game the sum of the scored goals.

```
$ awk '{print $4+$6 " " " $1 $2 $3}' ~/bundesliga_08_0405.txt!
5 Schalke-Bochum
2 Bielefeld-Stuttgart
4 Dortmund-Nürnberg
3 Leverkusen-Hamburg
3 Freiburg-Mainz
2 Kaiserslautern-Berlin
3 Wolfsburg-Mönchengladbach
4 Rostock-Hannover
3 Bremen-München
```

12. Determine with `awk` in which city the most spectators visited the match and print out the result this way:

The most spectators were in CITY (NUMBER).

```
$ awk -v max=0 'max<$7 {max=$7; city=$1} \
END{print"The most spectators were in",city, "("max")"}' \
~/bundesliga_08_0405.txt
The most spectators were in Dortmund (73500)
```