Exercise Sheet 6

Exercise 1 (File Systems)

1. Which information stores an inode?

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

3. What is a cluster in the file system?

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

5. How are directories in the Linux file systems technically implemented?

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

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

   □ Yes □ No

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

   □ Yes □ No

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

   □ Yes □ No

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

    □ write-back □ write-through

11. 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.

12. What is an absolute path name?

13. What is a relative path name?

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

    □ absolute path name □ relative path name

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

    □ absolute path name □ relative path name

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

    □ absolute path name   □ relative path name

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

    □ absolute path name   □ relative path name

18. What information stores the boot sector of a file system?

19. What information stores the super block of a file system?

20. Why do some file systems (e.g. ext2/3) combine the clusters of the file system to block groups?

21. What is the File Allocation Table (FAT) and what information does it store?

22. What is the objective of the journal in journaling file systems?

23. Name an advantage of journaling file systems compared with file systems without a journal.

24. What is the advantage of using extents compared with direct addressing of the cluster?

25. What is the result of defragmenting?

26. What sort of data processing is maximum accelerated by defragmenting?

27. For which scenario is defragmenting useful?

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

Exercise 3 (Pattern Comparison and Data Analysis)

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

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

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

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

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

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

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

```
<a href="BTS2014/index.html">Betriebssysteme (BTS)</a><p>
<b>This is a <i>HTML file</i></b><br>
<h2>This is a headline<h2>
```

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

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

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

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

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
```

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

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

9. Determine with `awk` all matches, which had more than 30000 spectators.

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

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

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).
    ```