1. The decision tree below is a decision tree resulting from running Mergesort on a lift with 3 elements \([a_1, a_2, a_3]\).

2. The tree below cannot be a decision tree of any comparison-based algorithm after adding all required labelling.

3. A decision tree of any comparison-based algorithm has to have all levels full except for the last one.

4. The binary tree below is a binary heap:
5. Let \([10,9,8,8,7,6,5,3,2,1]\) be an array representation of the binary heap. After removing the element of the highest priority you will get the array \([9,8,8,3,7,6,5,1,2]\).

6. The following list \([1,3,1,4,5,2]\) achieves the maximum number of swaps and comparisons possible for a list with 6 elements while recursively heapifying it to construct the heap. Note, that you count only comparisons of two elements of the lists and swaps of two elements of the list in this question.

7. The following tree can be seen at a step of Heapsort algorithm while removing vertices of the highest priority:

```
     9
    / \ \
   7   8
  / \  / \ \
 6   5 1  2
```

8. The sublist \([12,14]\) will be considered while looking for the 5th order statistic on the list \([5,3,9,10,15,14,20,12]\) using Quickselect algorithm.

9. 5 will be chosen as the pivot at some step of Quickselect algorithm to find the 3rd order statistic of the list \([17,9,6,5,10,22,14,5]\).

10. 10 is the 4th order statistic of the list \([14,1,11,2,3,10,7]\).

11. Selection sort achieves the maximum possible number of comparisons on the list \([1,2,3,4,5,6,7,8,9]\). Note, that you count only comparisons of two elements with at least one of them being an element of a list.

12. List \([2,5,1,1,7,8,10]\) can be seen at some step of Selectionsort algorithm while sorting \([10,2,5,7,1,8,1]\).

13. Insertion sort does larger number of elementary operations than Selection sort while sorting the list \([14,6,9,10]\). An elementary operation here is a comparison of two elements with at least one of them being an element of the list.

14. Lists \([10,15,11,17,16]\), \([10,11,12,9,13]\) and \([14,16,18,20,17]\) take the same number of comparisons while sorting by Insertion sort. You count comparisons only between two elements of a list.

15. Insertion sort achieves the minimum possible number of comparisons on the list \([1,2,3,4,5,6]\). You count comparisons only between two elements of a list.

16. List \([23,34,56,2]\) can be seen at some step of Insertionsort algorithm while sorting \([34,23,2,56]\).

17. Mergesort achieves the minimum possible number of comparisons on the list \([7,6,4,2,3,1]\). You count comparisons only between two elements of a list.

18. The sublist \([45,2,5]\) will be considered while running Mergesort algorithm on the list \([8,23,45,2,5,7,3,6,9]\).

19. Mergesort achieves the maximum possible number of comparisons on the list \([5,1,7,3,4,2,6]\). You count comparisons only between two elements of a list.

20. Suppose you need to order the list \([1_a, 3, 6, 8, 10, 2, 5, 18, 1_b, 4, 19, 23]\), where \(1_a, 1_b\) are two copies of 1. After running mergesort on this list you can get the following list:

\[1_b, 1_a, 2, 3, 4, 5, 6, 8, 10, 18, 18, 19, 23]\.

2
21. Quicksort achieves the minimum possible number of comparisons on the list [4, 1, 4, 6]. You count comparisons only between two elements of a list.

22. The sublist [8, 7, 10] will be considered while running Quicksort algorithm on the list [46, 8, 29, 11, 10, 7, 12].

23. Lists [1, 4, 4, 6] and [4, 1, 6] both achieve the largest possible number of comparisons while running Quicksort algorithm. You count comparisons only between two elements of a list.

24. 5 will be chosen as the pivot at some step of Quicksort algorithm while sorting the list [17, 34, 5, 8, 1, 4, 16].

25. 
\[
\begin{align*}
T(n) &= T(n - 1) + \frac{(C + 3)(n - 1)}{2} \\
T(1) &= 0
\end{align*}
\]

is the recurrence relation and the initial condition in the average case of the following piece of code:
Function Blah (array a[0..n-1])
if n=1 then return a
for j from 0 to n-2 do
    if a[j] < a[j + 1] then swap a[j] and a[j+1]
    Else do constant number C elementary operations
Blah(a[0..n-2]) return a
Elementary operations in this question are a comparison of two elements of the list a and a swap of two elements of the list a.

26. The solution of
\[
\begin{align*}
T(n) &= T(n - 2) + 2n + 1 \\
T(0) &= 0 \\
T(1) &= 1
\end{align*}
\]

is the following function:

\[
T(n) = \begin{cases} 
\frac{n(n + 3)}{2}, & \text{if } n \text{ is even,} \\
\frac{(n + 1)(n + 2)}{2} - 2, & \text{if } n \text{ is odd.} 
\end{cases}
\]

27. Solution of
\[
\begin{align*}
T(n) &= 2T\left(\left\lfloor \frac{n}{3} \right\rfloor \right) + T(n - 2 \left\lfloor \frac{n}{3} \right\rfloor) + 4 \\
T(0) &= T(1) = T(2) = 0
\end{align*}
\]

is \(\Theta(3^k)\).

28. Given the recurrence relation below:
\[
\begin{align*}
T(n) &= \frac{1}{n} (T(0) + T(1) + \ldots + T(n - 1)) + 5, \\
T(0) &= 0.
\end{align*}
\]

Then the solution is in \(\Theta(\log_{13}(n))\).

29. The algorithm A with running time \(T(n) = Cn \log_{10}(n^7)\) takes 10 second to process input of size \(10^4\). Then it will take more than 10 years to process an input of size \(10^{16}\).
30. The running time of the following piece of code is \( nC([\log_7(\log_3 n)] + 1) \left[ \frac{n^2 - 2}{6} \right] \)

\[
i \leftarrow 2
j \leftarrow 3
\]

while \( i < n \) do

while \( j \leq n \) do

for \( k \) from 1 to \( n \) do

constant number \( C \) of elementary operations

\[
j \leftarrow j^7
i \leftarrow i + 6
j \leftarrow 3
\]

Note, you count only comparisons of two elements of the list as elementary operations.

31. Algorithm A takes \( n^3 \log_3(n) + 10n^2 \) elementary operations and algorithm B takes \( 17n^6 + n^4 \log_7 n \). Then for large enough input algorithm A is faster that algorithm B.

32. The running time of the following piece of code is \( \Theta(n^2 \log_2(n)) \)

\[
k \leftarrow 1
\]

for \( i \) from 1 to \( n \) do

if \( i \) is a power of 5 then

while \( k \leq n \) do

\( k \) elementary operations

\[
k \leftarrow k + 1
\]

33. \( f(n) = 457 \cdot n^n + 67^n + n \log_2 n \) is \( O(n!) \).

34. \( f(n) = n^2 \log_2(\log_2(\log_2 n)) \) is \( \Omega(n^2 \log_2(\log_2 n)) \).

35. \( f(x) = \frac{45 \cdot 3^n + n^{1000000} \log_2(\log_2 n)}{n^7 + 1} \) is \( \Theta(3^n) \).
36. **Free-response:** Let \( a = [0..n-1] \) be a list with \( n \) elements, such that \( n \) is a power of 2. You know that this list has the following property:

if you run the mergesort on this list every time you split into two sublists one of them is ordered and the other one has elements which are strictly larger than any element of the sorted sublist.

a) Write the recurrence relation and initial conditions for the mergesort algorithm on lists satisfying the property above. In this question you count as elementary operations comparisons of two elements of a list and writing in a list.

b) Solve the recurrence relation from a)

c) give the best asymptotic bound of the solution from b)