Seksjon 1

1 OPPGAVE

1. (25%)

Write (in Java plus standard library, or in pseudocode) a method

int threeSumNeg(int[] a)

which returns the number of triples *i,j,k* with *0 <= i < j < k < a.length* such that *a[i] + a[j] + a[k] < 0*. You can assume that all elements of the array *a* are different. To get the full score your methode should run in *O(N^2 log N)* time, where *N = a.length*.

Fill in your answer here

2 OPPGAVE

2. (25%)

Let *int[] a* be an array of integer values between 0 and 99. Such an array can be sorted efficiently by using an auxiliary array *int[100] count*. Write a method (in Java or pseudocode, not using a method like *sort()* from a library)

void sort(int[] a)

which sorts *a*. To get the full score your method should run in time *O(N),* where *N = a.length*.

Fill in your answer here

3 OPPGAVE

3. (25%)

Explain what a hash table is. Describe first the functionality by explaining what the methods get(), put() and delete() of the API should do. Then explain what a hash function is and why collision resolution is important. Finally, describe an implementation by giving suitable datastructures plus a description of the implementation of the three methods above.

Fill in your answer here

4 OPPGAVE

4. (25%)

Given a directed graph *G* with nodes *0,...,V-1,* represented by adjacency lists of out-edges. More precisely, for each node *v*, *adj[v]* is a linked list which contains all nodes *w* such that *G* has an arrow from *v* to *w*. Write (in Java or in pseudocode) a method

boolean reachable(int s, int t)

which returns a boolean *true* if and only if there is a path from *s* to *t* in *G*.
(Hint: no information about the possible path from *s* to *t* needs to be returned.)
Analyse the worst-case run-time of your algorithm in terms of the number of nodes and arrows of *G*.

Fill in your answer here