

**Some important points:**

- (i) Read the entire exam thoroughly *before* you start!
- (ii) The teacher will normally take one round through the exam hall. Have your questions ready!
- (iii) Write your answers in the answer boxes and hand in the problem sheet. Feel free to use a pencil! Or draft your answers on a separate piece of paper, to avoid strikeouts, and to get a copy for yourself
- (iv) You are permitted to hand in extra sheets if need be, but the intention is that the answers should fit in the boxes on the problem sheets. Long answers do not count positively.

The exam has 20 problem, worth 100 points in total. The point value is given next to each problem.

\* \* \*

For each of the first 5 problems, you are to select your answer from among these algorithms:

- BFS
- BELLMAN-FORD
- DFS
- DAG-SHORTEST-PATH
- DIJKSTRA
- FASTER-ALL-PAIRS-SHORTEST-PATHS
- FLOYD-WARSHALL
- SLOW-ALL-PAIRS-SHORTEST-PATHS

(5 p) 1. Which algorithm on page 1 does not permit cycles?

(5 p) 2. Which algorithm on page 1 permits arbitrary positive but no negative weights?

(5 p) 3. Which algorithm on page 1 might not find the shortest path in a directed acyclic graph where all edge weights are 1?

(5 p) 4. Which algorithm on page 1 should you use to find the shortest paths from every node to every other in an arbitrary graph with positive edge weights, if you have  $\Theta(V^2)$  edges?

(5 p) 5. Which algorithm on page 1 should you use to find an augmenting path?

(5 p) 6. In the machine model of the textbook, integers normally have  $c \lg n$  bits. What is the requirement placed on  $c$ ?

(5 p) 7. You have found the best-case running time for an algorithm. Which one of  $\mathcal{O}$ ,  $\Omega$  or  $\Theta$  would you use to describe it?

(5 p) 8. What is the worst-case running time of MERGE-SORT?

(5 p) 9. What has BELLMAN-FORD discovered if it returns FALSE?

(5 p) 10. Let  $A = [3, 3, 1, 4, 4, 3, 1, 2, 3, 5]$ . How does  $C[0..6]$  look when COUNTING-SORT( $A, B, 6$ ) returns?

- (5 p) 11. Let  $A = [0, 9, 2, 8, 1, 5, 3, 4, 7, 6]$ . Assume that PARTITION chooses the last element as *pivot* (as in the textbook). Which index does PARTITION( $A, 1, 10$ ) return then?

- (5 p) 12. Let  $G = (V, E)$  be a graph with nodes  $V = \{0..9\}$  and with positive edge weights  $w$ . Let  $D[0..9]$  be a table of distance estimates, where  $D[u] = u.d$ , for  $u = 0..9$ . After DIJKSTRA( $G, w, 0$ ) is executed, we have  $D = [0, 2, 3, 5, 8, 6, 9, 1, 7, 4]$ . In which order have the nodes  $0..9$  been selected and visited?

- (5 p) 13. Let  $C = \{a, b, \dots, j\}$ , where  $a.freq = 11, b.freq = 12, \dots, j.freq = 20$ . Execute HUFFMAN( $C$ ). Let the left child be smaller (like the book), and let the left child be 0. What is the Huffman code for  $g$ ?

- (5 p) 14. Let  $A = [5, 0, 2, 7, 3, 9, 1, 6, 8, 4]$ . Execute BUILD-MAX-HEAP( $A$ ). How does  $A$  look afterward?

- (5 p) 15. Let  $A = [0, 1, 2, 3, 6, 4, 9, 8, 5, 7]$ . Execute EXTRACT-MIN( $A$ ). How does  $A$  look afterward?

- (5 p) 16. Let  $T(0) = 0$  and  $T(n) = T(n-1) + 2^n + n$  for  $n > 0$ . Solve the recurrence. Use asymptotic notation.

- (5 p) 17. Let  $T(0) = 0$  and  $T(n) = \pi^2 T(n/\pi) + n^2$  when  $n > 0$ , where  $\pi = 3.14159\dots$ . Solve the recurrence. Use asymptotic notation.

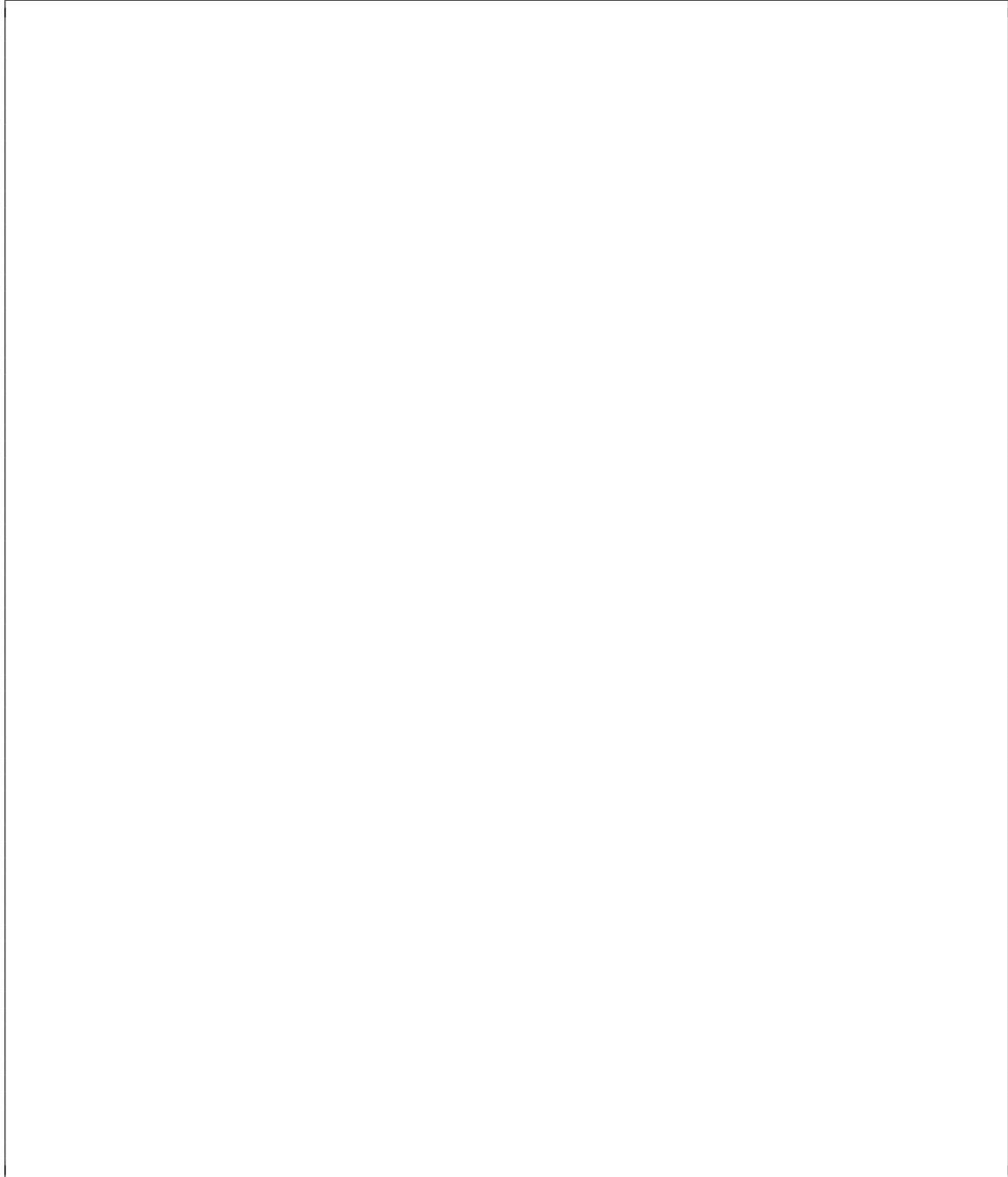
- (5 p) 18. You have found a polynomial-time reduction from A to B, where A and B are decision problems. An optimal algorithm for B has a worst-case running time of  $\Theta(2^n)$ . Can you say anything about the running time of algorithms for A? If yes, describe the running time with asymptotic notation.

- (5 p) 19. You have a memory area of  $n$  bytes, and are dividing it into segments, as shown in fig. 1 on page 5. A segment with a length of  $k$  bytes has a *cost* of  $c_k$ , and you wish to find a segmentation that minimizes the total cost  $f(n)$ . Write a recursive equation that gives the optimal value for  $f(n)$ . The solution should be suitable for memoization. You can assume  $f(0) = 0$ .

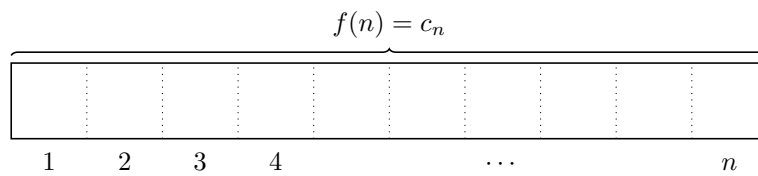
- (5 p) 20. As in the *0-1 knapsack problem*, a burglar has broken into a shop and found  $n$  items. She gives each item  $i$  a positive or negative value  $v_i$ , based on how heavy it is and how much she can sell it for. She wishes to select a subset  $S$  of the items such that  $\sum_{i \in S} v_i$  is maximized.

However: Some of the items depend on others. For example, she can't take the antique sword without also taking the antique scabbard. She can only select a subset where such dependencies are respected. If  $i$  depends on  $j$  and she wants to take  $i$ , then she *must* take  $j$  as well. (See fig. 2 for an example.)

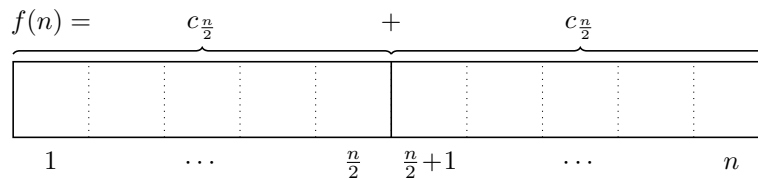
Describe an algorithm that solves the problem efficiently. Keep your explanation as brief and simple as possible. Feel free to draw a figure.



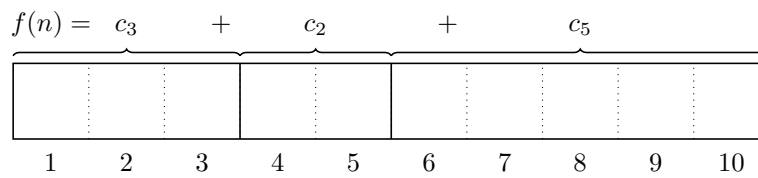
One segment of length  $n$ :



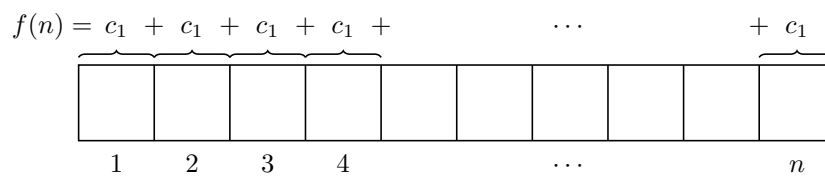
To equal-sized segments:



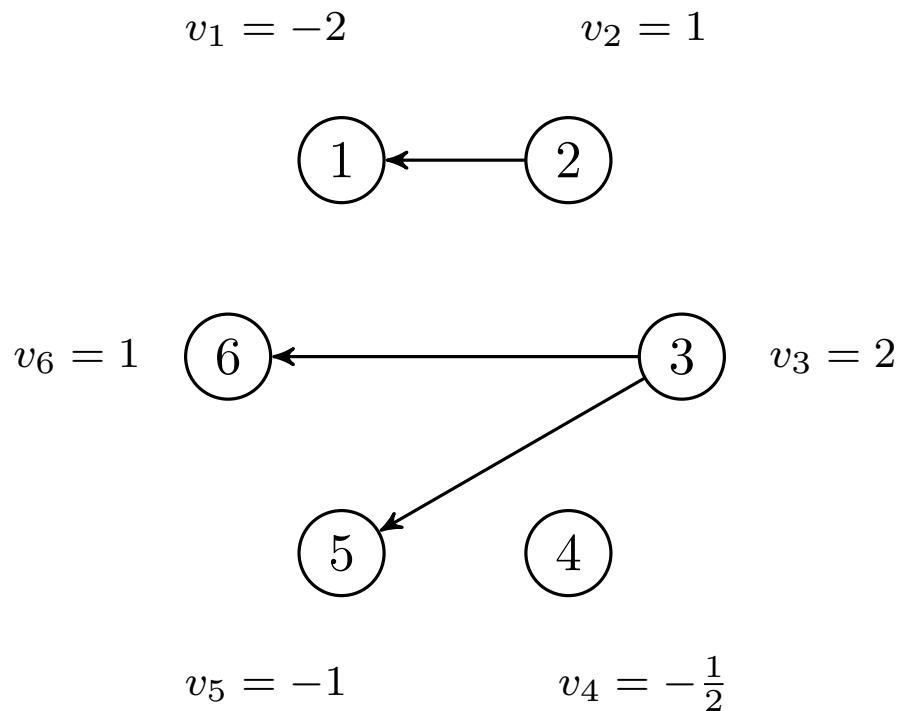
Three segments of varying lengths:



$n$  segments of length 1:



Figur 1: Illustration for problem 19. The memory area can be divided into anything from one to  $n$  segments, and the total solution has a cost of  $f(n)$ , which is the sum of the costs of the segments. You are to find a segmentation that minimizes this total sum of costs.



Figur 2: Illustration for problem 20. In this example, the burglar wants to select a subset of  $\{1, \dots, 6\}$ . Each item has an arrow to each of the objects it depends on. For example: If you want to take item 2, you must also take item 1. Totally, these two items contribute with a value of  $-1$ , so it would not pay to take them. Item 6 may be taken or not, without regard to other items. It has a positive value, so it pays to take it. If you wish to take item 3, you must also take items 5 and 6. Even though  $v_5$  is negative, it would pay to take these three (over simply taking item 6), as  $v_3 + v_5 > 0$ . There is no point in taking item 4.