Algorithms and Data Structures
Jacobs University Bremen
Dr. Kinga Lipskoch

Course: CH-231-A
June 24th, 2021

Homework 14 - Enhancing the Module Achievement

- Submit one ZIP file per homework sheet which contains one PDF file (including pictures, computations, formulas, explanations, etc.) and your source code file(s) with one makefile and without adding executable, object or temporary files.
- The implementations of algorithms has to be done using C, C++, Python or Java.
- The TAs are grading solutions to the problems according to the following criteria: https://grader.eecs.jacobs-university.de/courses/ch_231_a/2021_1/Grading_Criteria_ADS.pdf

Problem 14.1 *Dynamic Programming: Rod Cutting* (5 points)

Given a rod of length n inches and a table of prices p_i for $i = 1, 2, ..., n$, determine the maximum revenue r_n obtainable by cutting up the rod and selling the pieces. Note that if the price p_n for a rod of length n is large enough, an optimal solution may require no cutting at all.
Consider the following example:

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

Consider the case when $n = 4$. Cutting a 4-inch rod into two 2-inch pieces produces revenue $p_2 + p_2 = 5 + 5 = 10$, which is optimal.
Write a program for solving the problem from above such that the time complexity is not higher than $\Theta(n^2)$. Your solution has to determine the optimal revenue without the listing of the cuts.

Problem 14.2 *Dynamic Programming: Print Cuts* (5 points)

Modify your program to the previous problem (**Problem 14.1**) in a way such that you can print the cuts which correspond to the optimal solution.

Problem 14.3 *Other Solution: Rod Cutting* (5 points)

Solve the Rod Cutting problem using a different programming technique/approach than dynamic programming. After writing the program for this other solution, analyse the time complexity of your solution and compare it with the complexity of your previous solution(s) (**Problem 14.1** and **Problem 14.2**).

Problem 14.4 *Backtracking Problem: All Combinations with Constraints* (5 points)

Write a backtracking program which does the following: given a positive number n, find all combinations of $2 * n$ elements such that every element from 1 to n appears exactly twice and the distance between its two appearances is exactly equal to the value of the element.

For example,
Input:
$n = 3$
Output:
$3, 1, 2, 1, 3, 2$
$2, 3, 1, 2, 1, 3$

Problem 14.5 *Explanations* (10 points)

Write detailed textual explanations for each of your solutions for the four previous problems (**Problem 14.1**, **Problem 14.2**, **Problem 14.3** and **Problem 14.4**). Do not concentrate on the syntactical details, but on the algorithmic reasonsing and steps. Place the explanations into a PDF file which you include in the ZIP file together with your source code for the different problems.
Do not forget to list all the sources you have used for all problems.

How to submit your solutions

You can submit your solutions via *Grader* at **`https://grader.eecs.jacobs-university.de`** as a generated PDF file and/or source code files.
If there are problems with *Grader* (but only then), you can submit the file by sending mail to `k.lipskoch@jacobs-university.de` **with a subject line that starts with CH-231-A**.

Please note, that after the deadline it will not be possible to submit solutions. It is useless to send solutions by mail, because they will not be graded.

This homework is due by Monday, July 26th, 23:00.