Spring 2024

CS 412 (Algorithms: Design and Analysis)

Weekly Challenge 08: Dynamic Programming

Announced: Friday, March 15, 2024.
Deadline: Friday, March 29, 2024 (11:59 pm PKT).
Total marks: 2.

Instructions: Submit **individually** your solution as a PDF (*studentID.pdf*) typeset in LaTeX and Zip (*studentID_WC8_DP.zip*) for the code. You must submit your solution on Canvas.

---

1. (1 point) We are going to implement the rod cutting algorithm from CLRS 15.1 in a file `cutrod.py`. We will implement the following three versions of the solution: *top-down recursive*, *top-down memoized*, and *bottom-up*. Your code will be tested by `pytest` using the file, `test_cutrod.py`, given in `WC8_DP.zip`. To test your code implementation, open the directory containing `test_cutrod.py` and `cutrod.py` in the terminal, and run the following command:

```
pytest test_cutrod.py
```

TASKS:

(a) Write the top-down recursive version in a function, `cut_rod`.

(b) Write the top-down memoized version in a function, `cut_rod_memoized`.

(c) Write the bottom-up version in a function, `cut_rod_bottom_up`.

(d) All functions take two arguments, `p` and `n`, where `p` is the price array and `n` is the length of the rod. `p[i]` is the price of a rod of length `i`. All prices are positive and increase with length.

(e) Write all functions in the file, `cutrod.py`.

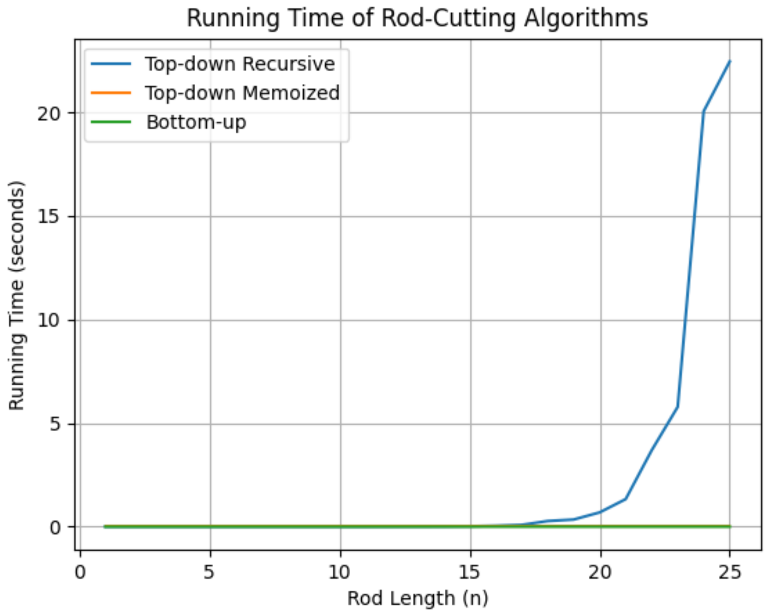
(f) Ensure that all tests pass by running `pytest` locally.

(g) Do not include any external packages.

(h) You may modify the error messages in `test_cutrod.py` to convey more information if you wish, but you may not alter any other functionality in it.

(i) Plot the running time of the three versions against `n` and include them below along with any relevant observations. Your code for plotting should be in `plot.py`.

Solution: The codes are in the ZIP file attached.



It is important to note that the running time complexities of these algorithms where the Top-down recursive function has exponential time complexity The exponential term in the top-down recursive solution makes it significantly slower than the other two for larger values of n. Even though both memoized and bottom-up solutions have a n^2 term, the constant factors involved in each algorithm's implementation can affect the observed runtime.