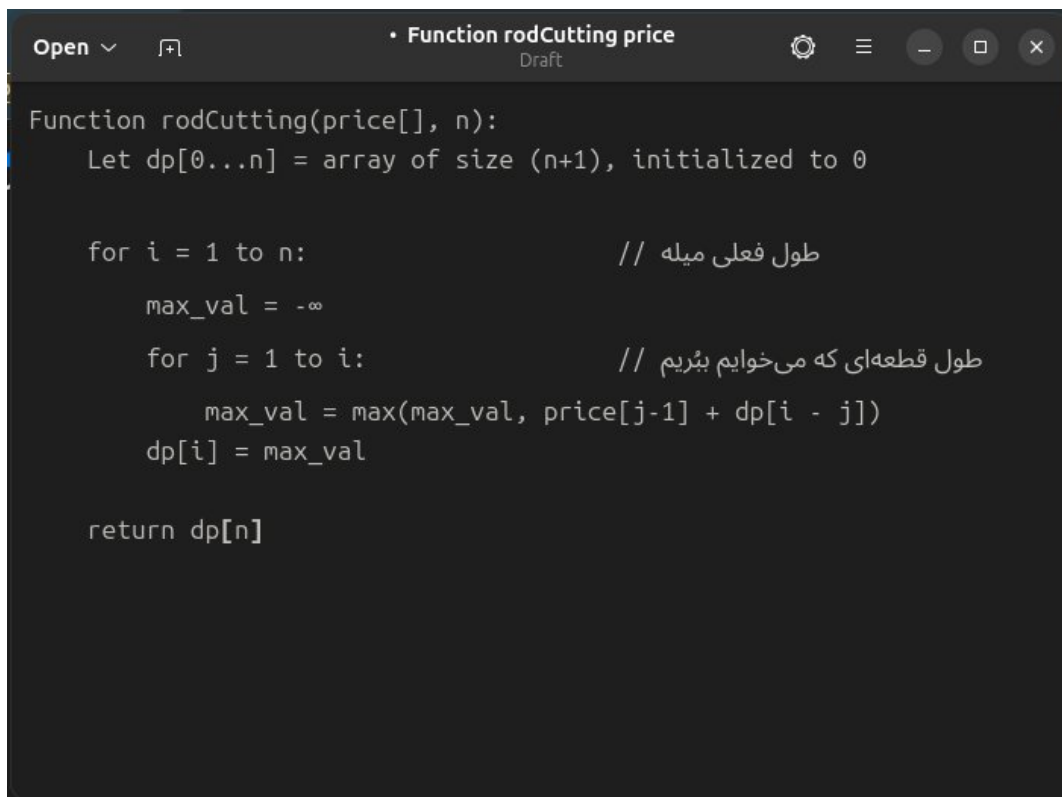


مسئله‌ی برش میله (rod cutting problem)

مسئله به ما یه میله به طول  $n$  اینچ میده و یه آرایه  $price$  که قیمت قطعات را به صورتی که  $price[i]$  قیمت قطعه‌ای با طول  $i$  را مشخص میکند.

هدف از مسئله اینه که میله را به چند قسمت تقسیم کنیم (ببریم) طوری که مجموع قیمت قطعات ماکزیمم مقدار شود.

با رویکرد بازگشتی  $O(n^n)$  حاصل میشه. ایده‌ی حل با استفاده از رویکرد دی‌پی با استفاده از tabulation با پیچیدگی  $2^n$  است. که ایده اینه که جدول  $dp$  را از پایین به بالا پر کنیم. که بیشترین درآمد ممکن از بریدن میله با طول  $i$  بدست میاد را نگهداری میکنیم.  $Dp[0] = 0$  است. این حالت بیس است. طول صفر یعنی هیچی.



```
• Function rodCutting price
Draft

Function rodCutting(price[], n):
    Let dp[0...n] = array of size (n+1), initialized to 0

    for i = 1 to n:                // طول فعلی میله
        max_val = -∞
        for j = 1 to i:            // طول قطعه‌ای که می‌خواهیم ببریم
            max_val = max(max_val, price[j-1] + dp[i - j])
            dp[i] = max_val

    return dp[n]
```

برای هر طول میله از ۱ تا  $n$  تمام برش‌های ممکن را بررسی میکنیم. از ۱ تا  $i$  میره که قطعه‌هایی با طول ۱ تا  $i$  را امتحان کنیم.  $price[j]$  چون ایندکس‌های آرایه از صفر شروع میشن. و  $dp[i-j]$  یعنی حداکثر قیمتی که برای باقیمانده‌ی میله بدست میاد.

```
Open ▾  price 1 5 8 9
Draft

price = [1, 5, 8, 9]

i = 4
j = 1 → price[0] + dp[3] = 1 + 8 = 9
j = 2 → price[1] + dp[2] = 5 + 5 = 10
j = 3 → price[2] + dp[1] = 8 + 1 = 9
j = 4 → price[3] + dp[0] = 9 + 0 = 9

dp[4] = max(9, 10, 9, 9) = 10 پس
جدول dp رو تا dp[n] می‌سازیم و در پایان:
dp[n] = جواب
```

محمد کریمی ۴۰۲۳۶۱۳۰۶۰

امیررضا محمدی یگانه ۴۰۲۳۶۱۳۰۶۸