

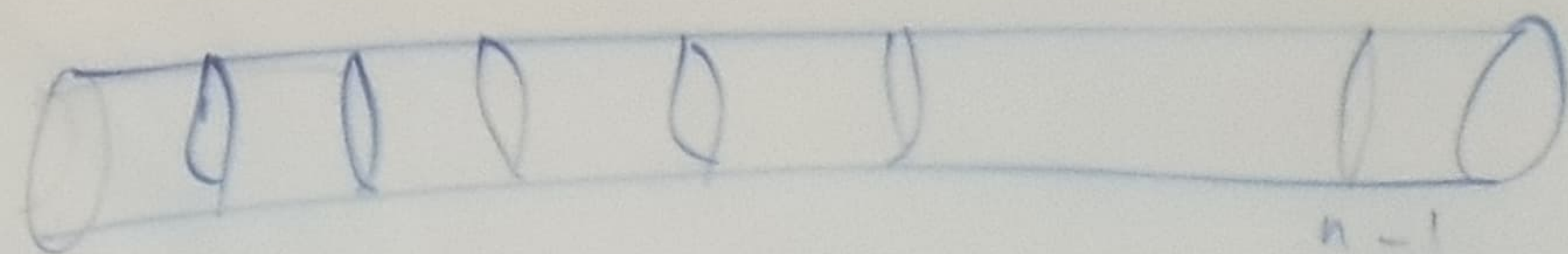
n

$$r[i] = \max \{ p_i, \quad$$

$$p_{i+r(i-1)} \quad$$

$$p_{i+r(i-2)} \quad$$

$$p_{i+r(i-3)} \}.$$



$n-1$

طول	1	2	3	4	5	6	7	8	9	10
قیمت p_i	1	5	8	9	10	17	17	20	24	30
r_i	1	5	8	10						

$$n = x_1 + x_2 + \dots + x_k$$

$$r = p_{x_1} + p_{x_2} + \dots + p_{x_k}$$

به ناک صدا

$$T[n] = 1 + \sum_{j=1}^{n-1} T[j]$$

f (int p, int n):

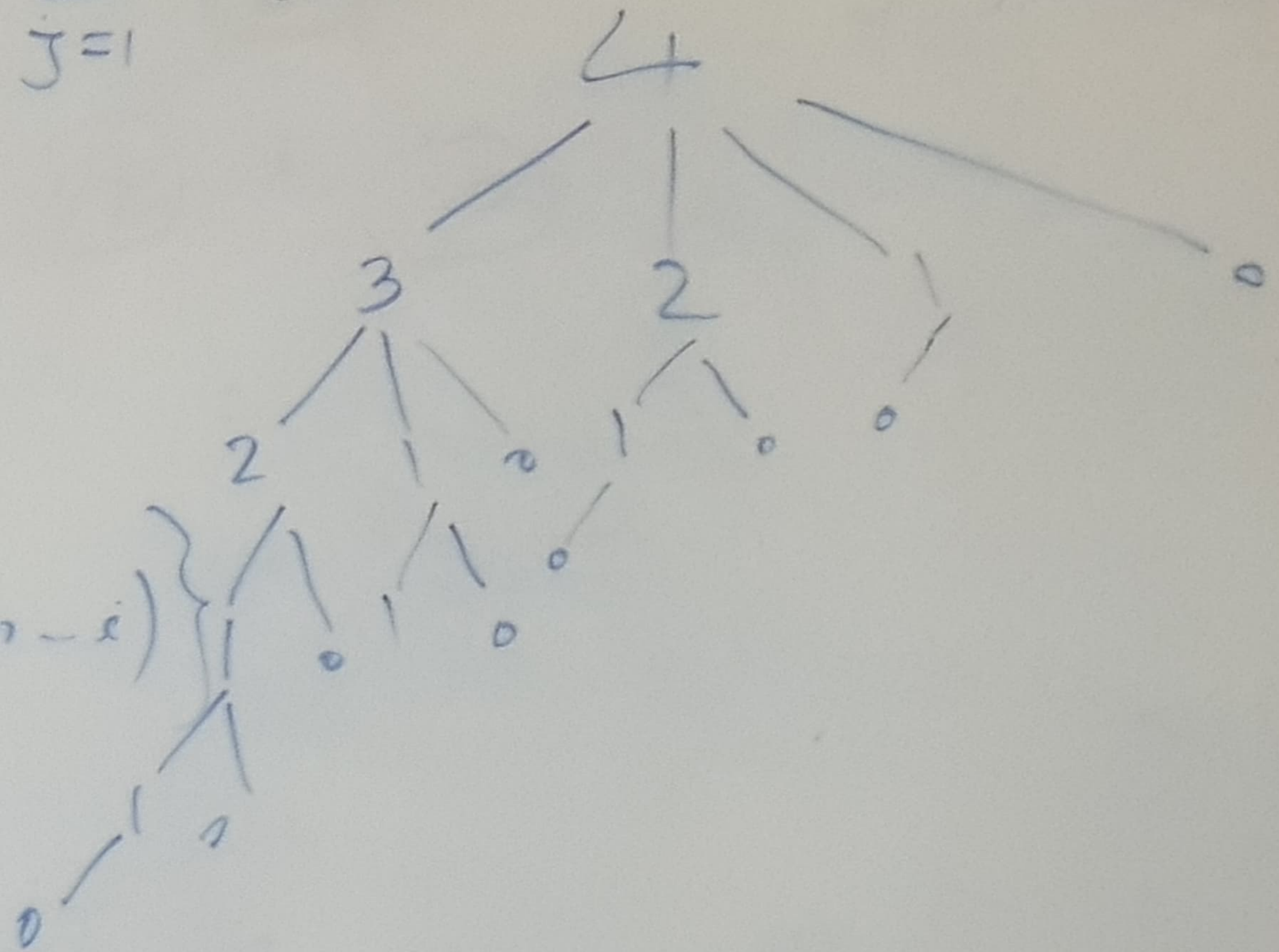
if n == 0:

return 0

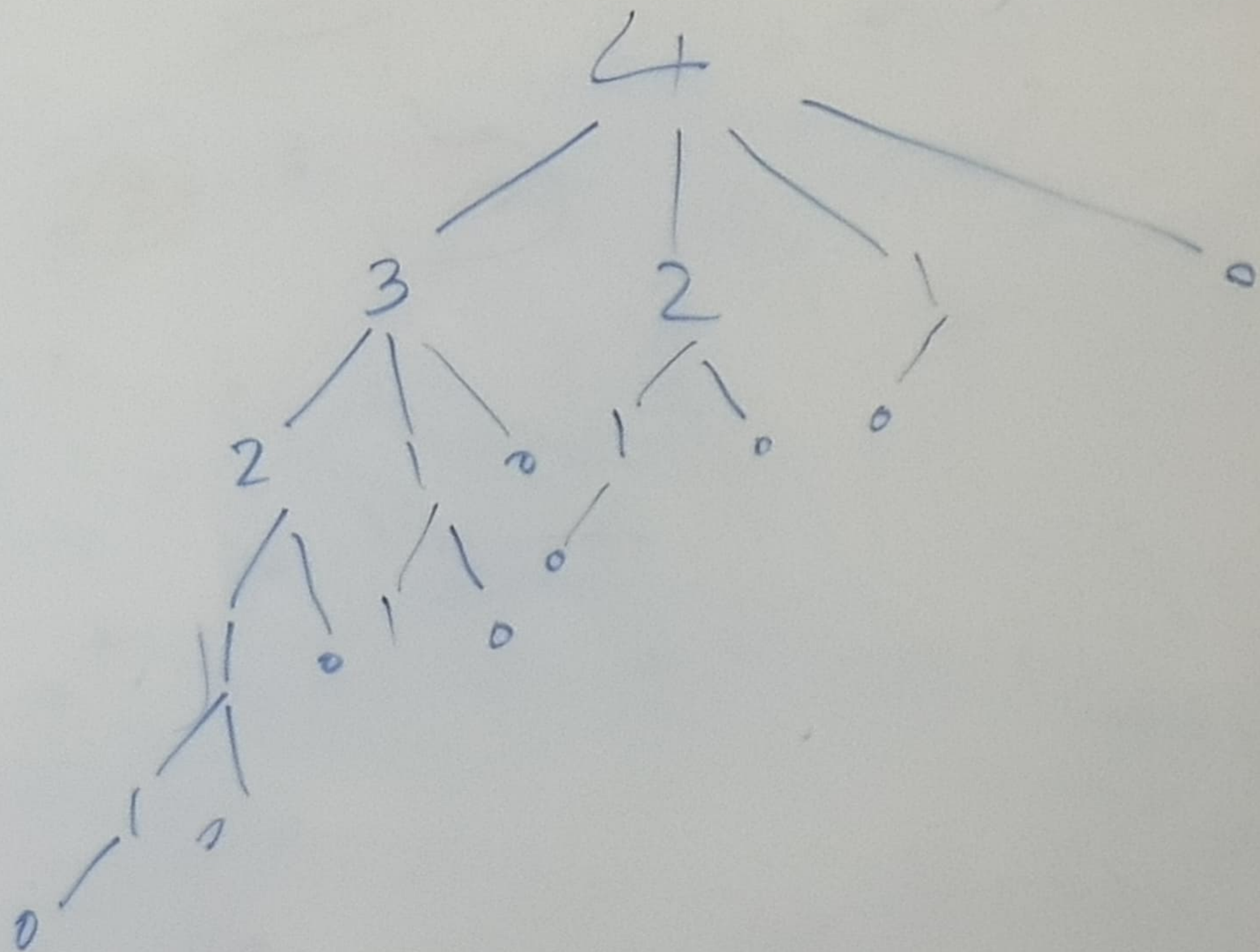
for i = 1 to n

$q = \max \{ q, p[i] + f(p, n-i) \}$

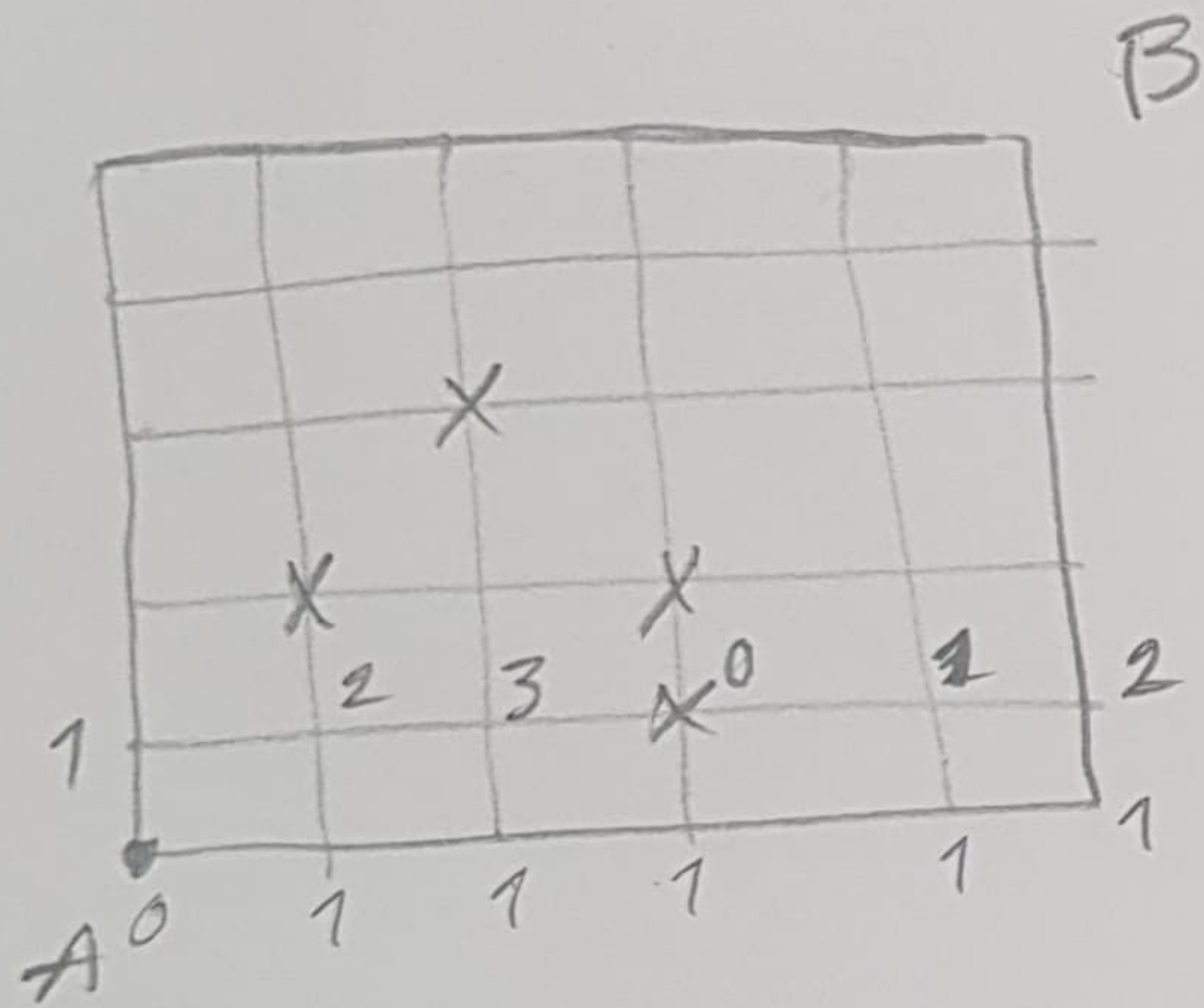
return q



به ناک فدا



تعداد مسیرهایی که از A به B وجود دارد، با این محدودیت که نمی‌توان از نقاط دارای مانع عبور کرد و فقط راست R یا بالا U می‌توانیم برویم.



هر خانه جمع خانه‌های چپ و پایین خود مسیر دارد.
به تعداد

مانع ها صفر مسیر دارند.

تدریس شد، اما یکس گرفته نشد!

به ناکفدا

knapsack ۱-۵

کوله پشتی ۱-۵

تاشی که شش غلام ارزش پیدا و اندازه پیدا
دارد و ظرفیت کل کوله پشتی ۷۰ است.

(فرض کنیم زیاده ها و زیاده ها عدد صحیح باشند).

هدف انتخاب اشیایی با بهترین ارزش است

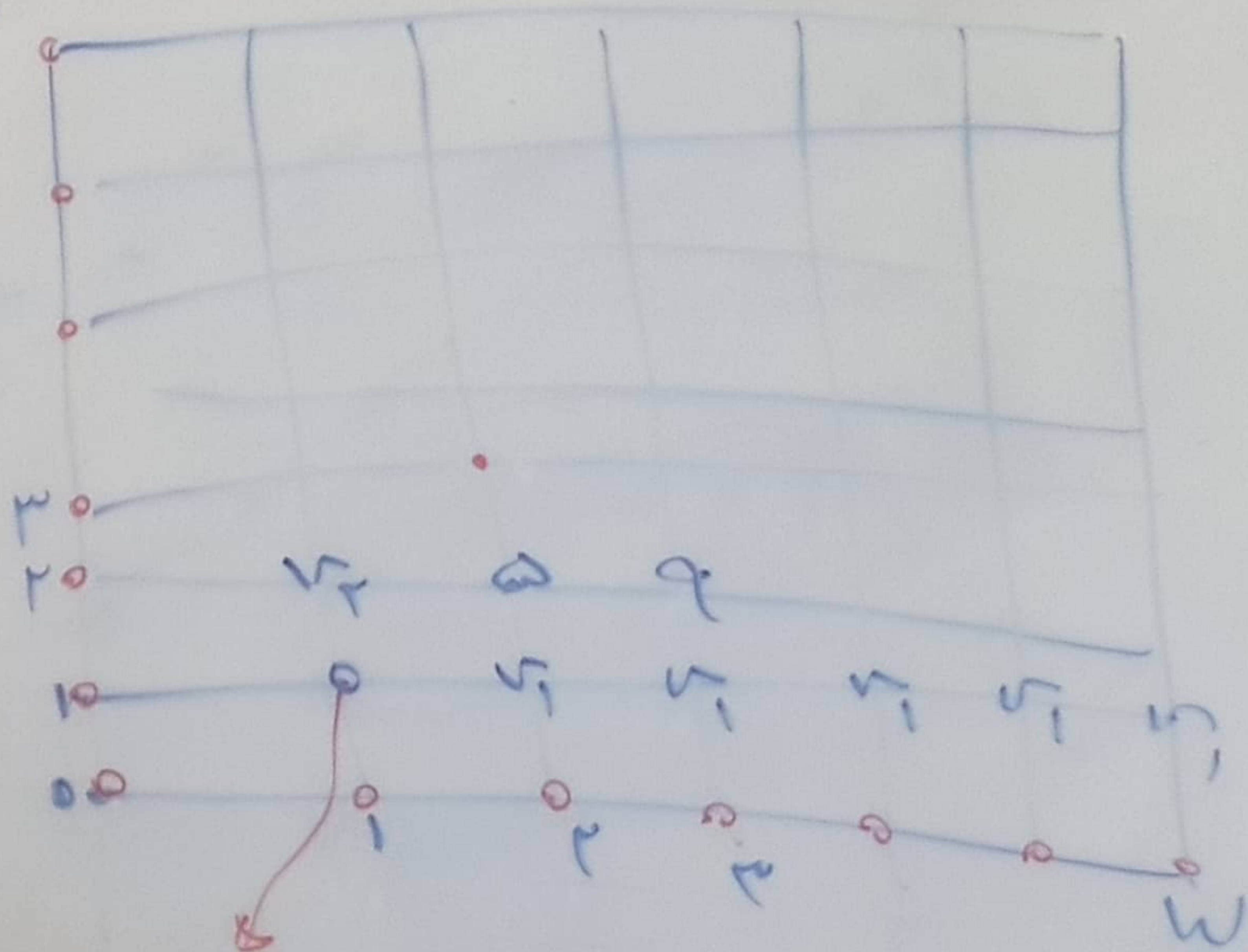
که مجموع اندازه آنها از ظرفیت کوله پشتی بیشتر نشود.

بهترین سودی که از انتخاب اشیاء
اتفاق می افتد می توان به دست آورد
به شرحی که ظرفیت کوله پشتی
ش باشد.

$P =$
شوند

$$p_{i,j} = \max \left\{ p_{i-1,j}, p_{i-1,j-w_i} + v_i \right\}$$

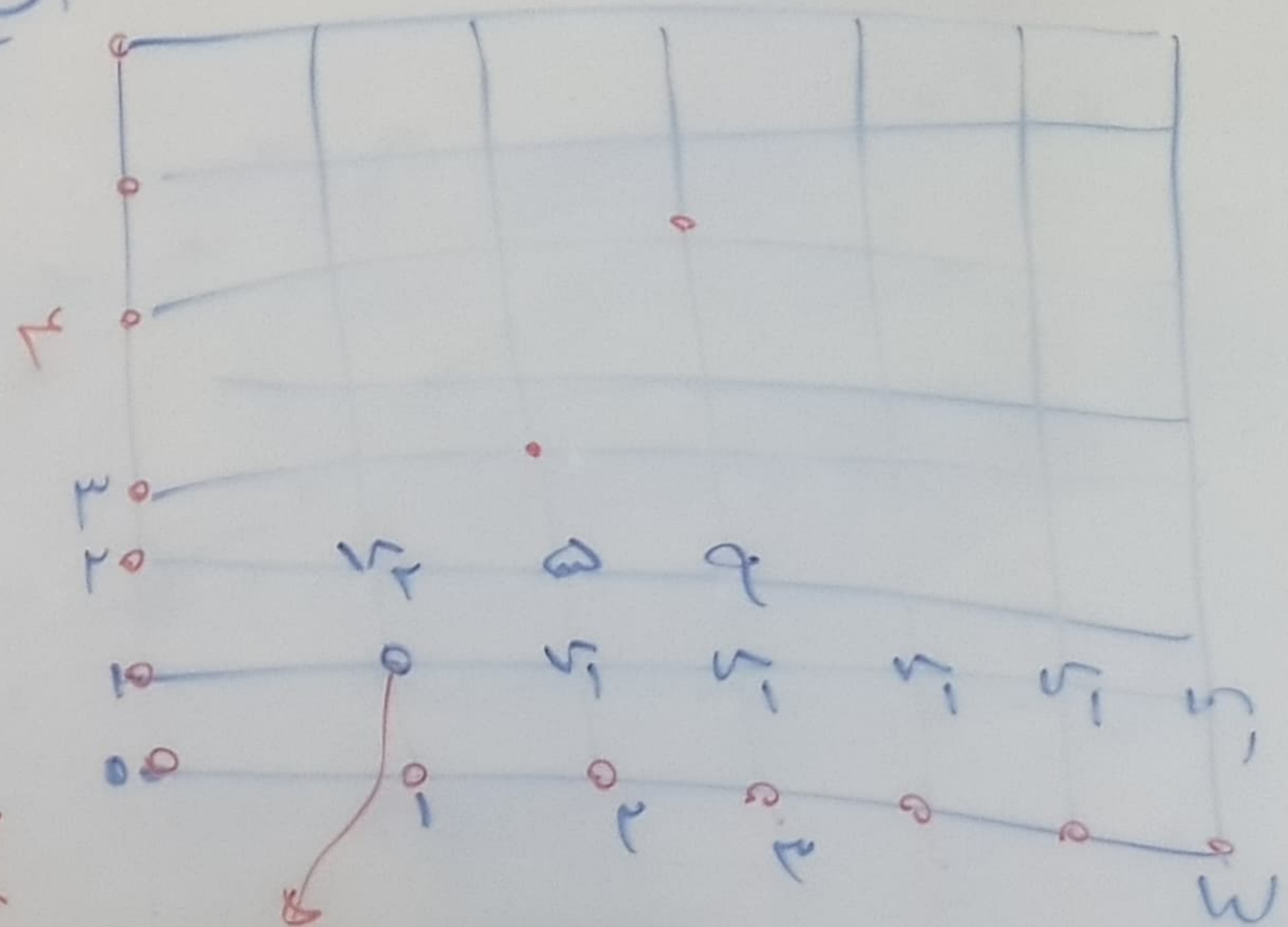
اشیا



ظرفیت
اولیه

شی	۱	۲
وزن	۵	۴
بازار	۲	۱

اسی



$$p_{i,j} = \max \left\{ p_{i-1,j}, p_{i,j-1} + v_i \right\}$$

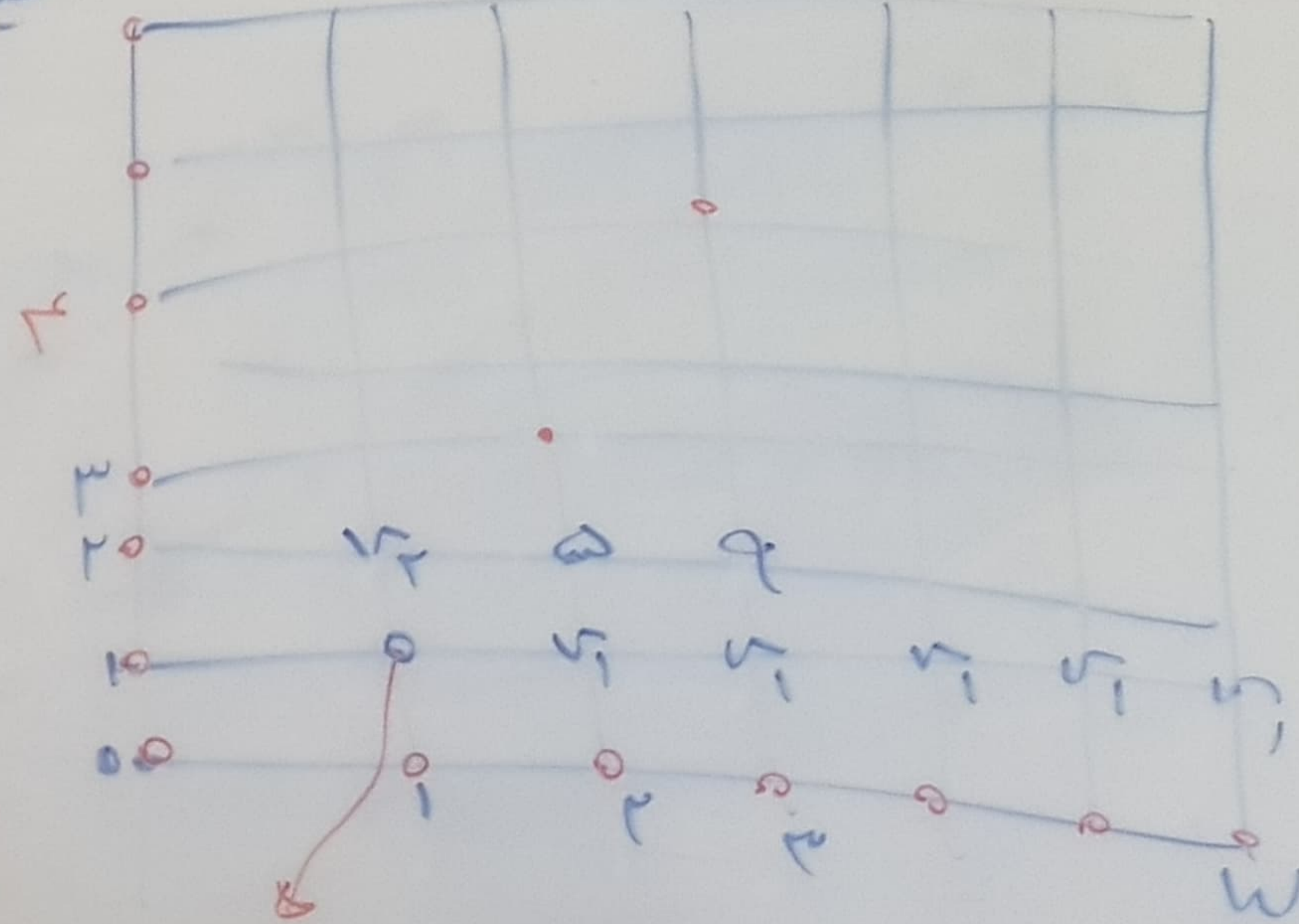
نہ
میں
 $O(nw)$

ظہیر
کوہستانی

سی	۱	۲
د	۱	۲
۵	۱	۲
۲	۱	۲

$$p_{m,n} = p_{m,n-1} + v_m$$

اشیا



$$P_{i,j} = \max \left\{ P_{i-1,j}, P_{i-1,j-w_i} + v_i \right\}$$

نقطه
منجمله

وزن

اثبات: استقر

```

14 def dp_knapsack(items, weight_limit):
15     weight_limit += 1
16     max_value = [[0] * weight_limit for _ in range(len(items))]
17
18     for weight in range(weight_limit):
19         if weight >= items[0].weight:
20             max_value[0][weight] = items[0].value
21
22     for item_number in range(1, len(items)):
23         for weight in range(weight_limit):
24             option1 = max_value[item_number - 1][weight]
25             option2 = 0
26             if items[item_number].weight <= weight:
27                 option2 = max_value[item_number][weight - items[item_number].weight] \
28                     + items[item_number].value
29             max_value[item_number][weight] = max(option1, option2)
30     return max_value[-1][-1]

```

نقطه
اثبات

$$P_{i,j} + v_i$$

شی	۱	۲
وزن	۵	۷
فایده	۱۰	۱۲