1. Pair

* Từ mảng a tạo ra hai nhóm:

+ Nhóm 1: k số bé nhất

+ Nhóm 2: k số lớn nhất

* Lần lượt ghép cặp cho từng phần tử của hai nhóm: 2 phần tử bé nhất của hai nhóm , 2 phần tử bé thứ hai của hai nhóm, …
* Tìm chênh lệch giá trị nhỏ nhất của từng cặp

1. Mdoor

* Sub1: m<=30, n<=30, n<=30

+ QHĐ : f[t][i][j] là số điểm lớn nhất đạt được khi đến ô (i, j) và sd cửa t lần

+ f[t][i][j] = max + w[i][j].

* Độ phức tạp:
* Sub2: m, n<=1000, c=0

+ QHĐ : f[i][j] là số điểm lớn nhất đạt được khi đến ô (i, j).

+ f[i][j] = max( f[i-1][j-1], f[i-1][j], f[i-1][j+1] ) + w[i][j]

* Độ phức tạp:
* Sub3: m, n<=1000 , c<=30

+ Ý tưởng tương tự sub1, tuy nhiên ta có thể tối ưu vong for tìm f[t-1][i’][j’] bằng cách sử dụng mảng g[t] là giá trị lớn nhất trong các f[t][i][j]

* Độ phức tạp:
* Sub4: m, n <=1000, c<=1e9

+ Chia bài toán thành 3 giai đoạn:

\_ Giai đoạn 1: từ hàng 1 đến một ô cửa bất kì -> x

\_ Giai đoạn 2: từ 1 ô cửa đến 1 ô của khác -> y

\_ Giai đoạn 3: từ 1 ô cửa đến hàng m -> z

-> kq: x + (c-1)\*y + z

* Độ phức tạp:

1. Sweet3 ( đặt tính rồi tính )

* Gọi f[i][j] là số cách tạo đến vị trí i (tính từ phỉa sang) mà nhớ j.
* F[i][j] += f[i-1][j’]

Trong đó, ( x\*a+y\*b+z\*c+j’ )%10 = cs thứ i của n tính từ phải sang và ( x\*a+y\*b+z\*c+j’ )/10 = j

* Chúy ý: có thể chuyển xử lí cơ số 10 thành cơ số 2 để tối ưu hơn nếu cần.

1. Table

* Sub1: n<=6: xâu kí tự cchir nằm trông dãy giá trị 1->1e5 nên ta sẽ tạo dãy từ 1->1e5 rồi for tìm lúc xuất hiện của xâu
* Sub2: 6<n<=30: chia ra ba trường hợp:

+ Trường hợp 1: Nhiều số trong t -> làm giống sub3

+ Trường hợp 2: Nó chính là một số.

+ Trường hợp 3: nó chứa hai số: phần cuối số trước và phần đầu số sau. Áp dụng ý tưởng bài sweet

* Sub3: 30 < n <= 150: vì một số có độ dài lớn nhất chỉ khoảng 16 nên chắc chắn trong xâu t có trọn ít nhất 1 số -> tìm đoạn số khớp với xâu và xuất kq.