**HP**

-Đầu tiên sort lại dãy theo đoàn , ta có :

Ans += sum \* presum;

Trong đó sum là tổng a[i] của đoàn đang xét , presum là tổng của cac đoàn trước đó

**ALP**

**-**Tính mảng d[i][j] là đường đi nhỏ nhất từ i đến j bằng Dijkstra

-nếu 2 cặp u v có cùng số hiệu G thì G += d[u][v]

Ngược lại Y += d[u][v]

**PLAN**

* Sub1 : - gọi f[i][j][k] là số cách chọn của nx số từ 1 đến i để có tổng nguyên liệu P là j và tổng nguyên liệu Q là j
* f[i][j][k] = f[i – 1][j][k]
* nếu a[i] >= j và b[i] >= k thì f[i][j][k] += f[i][j – a[i]][k - b[i]]

* Sub2: ta sẽ quy hoạch động chữ số

-dp[i][t1][t2] là số cách đến cột i mà nhớ cột i + 1 là t1 , nhớ cột i + 2 là t2;

- Tối ưu bằng cách chuyển sang hệ nhị phân để xử lý

- mỗi dp i t1 t2 đệ quy để sinh ra n số rồi cập nhật cho cột tiếp

**IJOB**

- Tiếp cận vs bài toán đơn giản hơn là không có điều kiện mảng P

Khi đó công việc a ưu tiên trước công việc b khi

a.w / a.t < b.w / b.t

- Quy ước lại cách tính bài toán theo thời gian bắt đầu làm việc.Vì nó không thay đổi tính chât bài toán mà chỉ thay đổi thời gian bằng hằng số i.w \* i.t , ban đầu ta sẽ cộng tất cả nó vào kết quả

- Quay trở lại bài toán chính , ta thấy hình dạng thứ tự nó khi xây xong sẽ có dạng cây

\_ Với mỗi nhánh ta sẽ cố gắng đưa nó về 1 dãy mà thỏa mãn tính chất số đứng trước luôn tốt ưu hơn số sau

-với mỗi đỉnh u ta sẽ cho các nhanh con của nó vào 1 cái multiset và ta se cho công việc u ở đầu tiên nếu u không tốt hơn công việc j thì ta sẽ hợp 2 công việc

u.w = u.w + j.w , u.t = u.t +j.t

khi đó cộng thêm vào kết quả 1 khoảng j.w + u.t

sau khi gộp lại xong ta sẽ cho công việc được gộp vào multiset

-Cuối cùng sẽ được 1 dãy của bài toán ,

ans += Time \* i.w , Time + i.t;