Write-up cho bài 6:

Đầu tiên đặt res là tổng lớn nhất có thể, đặt mặc định = 0; L và R là 2 vị trí đang xét, đặt mặc định = 0.

Bước 1: Tạo mảng S chứa toàn bộ các tổng có thể của mảng B (Eg: S[0] = B[0], S[1] = B[0]+B[1], .., S[N-1] = B[0] + B[1] +.. + B[N-1].

Bước 2: Xét từ đầu dãy A (mình sẽ kiểm tra điều kiện 1 trước): Có thể sử dụng biến đánh dấu vị trí hiện thời đang xét của dãy A (mình sẽ coi là vị trí i), sau đó kiểm tra lần lượt từ vị trí bằng cách xét trở ngược lại, tới khi nào gặp phần tử trước lớn hơn sau (làm không thoả mãn điều kiện A[R] là max) thì dừng lại. Ta coi vị trí dừng lại là j. Từ đây ta sẽ có 1 cặp L và R thảo mãn điều kiện 1 với R là vị trí đang xét của dãy A (vị trí i) còn L là vị trí đã xét trở ngược lại (vị trí j). Ta sẽ có 2 trường hợp cho L:

1. L = 0 => Tổng từ L tới R sẽ chính bằng S[i] (vì S[i] = B[0]+..B[i]
2. L 0 => Tổng từ L tới R sẽ bằng S[i] - S[j-1] (hay tương tự là từ B[j]+…B[i])

Bước 3: So sánh tổng của dãy L với R đang xét với res, nếu > hơn hẳn thì lưu tổng dãy đó vào res, đồng thời cập nhật vị trí L và R thành dãy đang xét (chú ý phải > hơn hẳn, không phải vì đề bài cho là in dãy đầu tiên tìm được)

Bước 4: Thực hiện vòng lặp tới khi xét hết dãy A thì thôi, kết quả cuối cùng là cặp L và R thoả mã

Sol: <https://ideone.com/D13t2K>