Bài 1. Tongdao.cpp

Cho một mảng gồm n số nguyên, các chỉ số của mảng bắt đầu từ 0. Nghĩa là a0,a1,..,a_{n-1}, giữa các số có một khoảng trắng.

Bạn có quyền đảo ngược tối đa một đoạn con liên tục của mảng trên để tổng các số ở vị trí chẵn thu được là lớn nhất.

Một đoạn con liên tục là đoạn a[l,r] sao cho ta có a_l,a_{l+1},..,ar.

Đầu vào **Tongdao.inp**

Dòng đầu là số t là số test. ($1 \le t \le 2.10^4$).

Dòng đầu tiên trong mỗi test là số nguyên n ($1 \le n \le 2.10^5$) là số phần tử trong mỗi test.

Dòng tiếp theo trong test là giá trị các phần tử a0,a1,..,an_{n-1} . ($1 \le ai \le 10^9$).

Đầu ra **Tongdao.out**

Ghi kết quả của mỗi test trên một dòng.

Ví dụ

Input	output
4 8 1 7 3 4 7 6 2 9 5 1 2 1 2 1 10 7 8 4 5 7 6 8 9 7 3 4 3 1 2 1	26 5 37 5

Bài 2. Groups.cpp

Có n lập trình viên và bạn muốn chia thành nhiều nhóm nhất có thể (nhóm phải có ít nhất 1 người). Kỹ năng của lập trình viên thứ i là ai. Có một yêu cầu đối với mỗi nhóm khi thành lập là: số lượng lập trình viên trong nhóm nhân với kỹ năng tối thiểu trong số tất cả các lập trình viên trong nhóm phải có ít nhất là x.

Mỗi lập trình viên chỉ được tham gia vào một nhóm. Một số lập trình viên có thể không được chọn vào nhóm nào.

Tính số lượng nhóm tối đa mà bạn có thể tạo ra.

Đầu vào Groups.inp

Dòng đầu tiên chứa số nguyên t ($1 \le t \le 1000$) - số lượng test.

Dòng đầu tiên của mỗi test chứa hai số nguyên n và x ($1 \le n \le 10^5$; $1 \le x \le 10^9$) - số lượng lập trình viên và hạn chế kỹ năng nhóm tương ứng.

Dòng thứ hai của mỗi test chứa n số nguyên a1, a2, ,an. ($1 \le ai \le 10^9$), trong đó ai là kỹ năng của lập trình viên thứ i.

Tổng n trên tất cả các đầu vào không vượt quá 10^5.

Đầu ra Groups.out

Đối với mỗi test, in một số nguyên là số lượng nhóm tối đa mà bạn có thể tạo ra.

Groups.inp	Groups.out
3 5 10 7 11 2 9 5 4 8 2 4 2 3 4 11 1 3 3 7	2 1 0

Bài 3 **fullbox.cpp**

Bạn được cho một cái túi kích thước n ngoài ra bạn có m hộp. Kích thước của hộp thứ i là ai, trong đó mỗi ai là một số nguyên không âm lũy thừa của 2.

Bạn có thể chia hộp đã cho thành hai phần có kích thước bằng nhau. Nhiệm vụ của bạn là phải chia làm sao để bỏ đầy vào túi.

Ví dụ: nếu n = 10 và a = [1,1,32] thì bạn phải chia hộp có kích thước 32 thành hai hộp có kích thước 16, sau đó chia hộp có kích thước 16 thành hai hộp kích thước 8. Vì vậy, bạn có thể bỏ đầy vào túi các hộp có kích thước 1, 1 và 8.

Tính số lượng cách phân chia ít nhất để bỏ đầy vào túi có kích thước n.

Đầu vào **fullbox.inp**

Dòng đầu tiên chứa một số nguyên t ($1 \le t \le 1000$) - số lượng test.

Dòng đầu tiên của mỗi trường hợp thử nghiệm chứa hai số nguyên n và m ($1 \le n \le 10^{18}, 1 \le m \le 10^5$) - kích thước của túi và số lượng hộp tương ứng.

Dòng thứ hai của mỗi test chứa m số nguyên a1, a2,..,am ($1 \le ai \le 10^9$) - kích thước của các hộp. Nó được đảm bảo rằng mỗi ai là một số lũy thừa của 2.

Đầu ra **fullbox.out**

Đối với mỗi test, in ra một số nguyên là số lượng phân chia tối thiểu để đổ đầy túi có kích thước n (hoặc ghi vào file −1, nếu không thể).

Ví dụ:

fullbox.inp	fullbox.out
3 10 3 1 32 1 23 4 16 1 4 1 20 5 2 1 16 1 8	2 -1 0

Hết