Ngôn ngữ lập trình C++

§05§ BÀI TẬP LẬP TRÌNH CƠ BẢN VÀ NÂNG CAO

1☼. (*Easy*) – Ghép chữ số - ConCatDigit.Cpp

Cho 3 chữ số a, b, c ($a > 0$). Gọi N là số được tạo bởi 3 chữ số a, b, c và có dạng $N = \overline{abc}$.

Yêu cầu: Hãy đưa ra $2 \times N$.

Dữ liệu cho trong file **ConCatDigit.Inp** gồm 3 chữ số a, b, c.

Kết quả ghi ra file **ConCatDigit.Out** là giá trị $2 \times N$.

Ví dụ:

ConCatDigit.Inp	ConCatDigit.Out
123	246

2☼. (*Easy*) – Chẵn nhân 2, lẻ nhân 3 – C2L3.Cpp

Cho dãy số nguyên $A_1, A_2, .. , A_N$. Với mỗi số hạng A_i ($i = 1, 2, .., N$), ta đặt $B_i = 2A_i$ nếu A_i là một số chẵn, $B_i = 3A_i$ nếu A_i là một số lẻ. Gọi $S = B_1 + B_2 + .. + B_N$.

Yêu cầu: Tính giá trị S.

Dữ liệu cho trong file **C2L3.INP** gồm:

- Dòng đầu ghi số nguyên dương N ($N \leq 100000$).
- Dòng thứ 2 ghi N số nguyên $A_1, A_2, .. , A_N$ ($|A_i| \leq 10^5$).

Kết quả ghi ra file **C2L3.OUT** là giá trị của tổng S.

Ví dụ:

C2L3.INP	C2L3.OUT	Giải thích
3 1 2 3	16	$A_1 = 1 \rightarrow B_1 = 3A_1 = 3$ $A_2 = 2 \rightarrow B_2 = 2A_2 = 4$ $A_3 = 3 \rightarrow B_3 = 3A_3 = 9$ $S = B_1 + B_2 + B_3 = 16$

3☼. (*Easy*) - Chia kẹo cho 3 anh em – DivideCandy3.Cpp

Hiện tại, 3 anh em Tôm, Tép, Cua có số kẹo lần lượt là a, b, c. Bố X viên kẹo. Tính xem, liệu Bố có thể cho 3 anh em X viên kẹo để số kẹo mỗi người sau khi cho là bằng nhau.

Dữ liệu cho trong file **DivideCandy3.Inp** gồm:

- Dòng đầu ghi 3 số nguyên dương a, b, c ($a, b, c \leq 100$) tương ứng là số kẹo của ba anh em Tôm, Tép, Cua.
- Dòng thứ 2 ghi số nguyên dương X là số kẹo của Bố sẽ cho cả ba anh em.

Kết quả ghi trong file **DivideCandy3.Out** là 'Yes' nếu có cách chia, ngược lại ghi 'No'.

Ví dụ:

DivideCandy3.Inp	DivideCandy3.Out	Giải thích
3 4 5 3	Yes	Cho Tôm 2 cái: Tép 1 cái và Cua 0 cái.

4☼. (*Medium*) Xóa số tính điểm – EraseMark1.Cpp

Em Tép viết lên bảng một dãy số gồm $2N$ số nguyên không âm $A_1, A_2, .., A_{2N}$. Anh Tôm sẽ thực hiện N lần xóa, mỗi lần xóa, Tôm chọn 2 số ở 2 đầu dãy và xóa đi hai số đó. Nếu 2 số bị xóa bằng nhau thì điểm của lần xóa là tổng 2 số bị xóa, nếu 2 số bị xóa khác nhau thì lần xóa này không được điểm.

Yêu cầu: Cho dãy số mà em Tép đã viết lên bảng. Tính điểm nhận được của Anh Tôm.

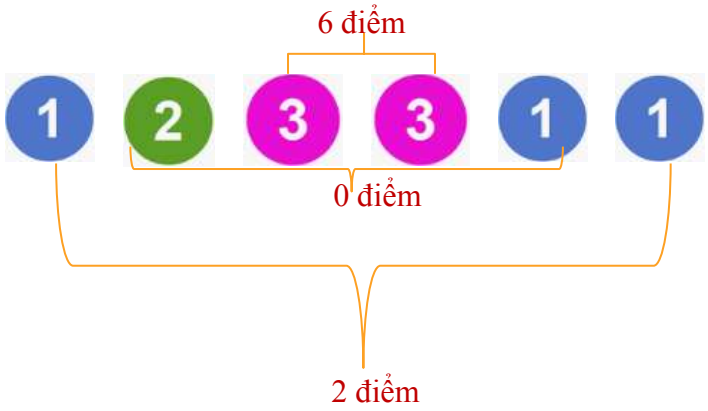
Dữ liệu cho trong file **EraseMark1.Inp** gồm:

- Dòng đầu ghi số nguyên dương N ($N \le 100000$).
- Dòng thứ 2 ghi $2N$ số nguyên không âm $A_1, A_2, .., A_{2N}$ ($0 \le A_i \le 10^6$).

Kết quả ghi ra file **EraseMark1.Out** là số điểm mà anh Tôm khi xóa dãy nhận được.

Ví dụ:

EraseMark1.Inp	EraseMark1.Out
3 1 2 3 3 1 1	8



5☼. (*Hard*) Xóa số trên đoạn và tính điểm – EraseMark2.Cpp

Em Tép viết lên bảng một dãy số gồm $2N$ số nguyên không âm $A_1, A_2, .., A_{2N}$.

Anh Tôm sẽ chọn $2K$ số liên tiếp trong dãy gồm $2N$ số mà em Tép đã viết ($K \le N$). Sau đó thực hiện K lần xóa (trên dãy gồm $2K$ số được chọn), mỗi lần xóa, Tôm chọn 2 số ở 2 đầu dãy và xóa đi hai số đó. Nếu 2 số bị xóa bằng nhau thì điểm của lần xóa là tổng 2 số bị xóa, nếu 2 số bị xóa khác nhau thì lần xóa này không được điểm.

Yêu cầu: Cho dãy số mà em Tép đã viết lên bảng. Tính xem, anh Tôm có thể nhận được số điểm lớn nhất bằng bao nhiêu.

Dữ liệu cho trong file **EraseMark2.Inp** gồm:

- Dòng đầu ghi 2 số nguyên dương N và K ($1 \le K \le N \le 10000$; $K \le 100$).
- Dòng thứ 2 ghi $2N$ số nguyên không âm $A_1, A_2, .., A_{2N}$ ($0 \le A_i \le 10^6$).

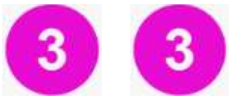
Kết quả ghi ra file **EraseMark2.Out** là số điểm mà anh Tôm khi xóa dãy nhận được.

Ví dụ:

EraseMark2.Inp	EraseMark2.Out
3 1 1 2 3 3 1 1	6

Dãy ban đầu:

Tôm chọn 2 số liên tiếp:



Điểm nhận được: 3 + 3 = 6.

6☼. Tổng Số hoàn hảo – SumPerfectNum.Cpp

Số tự nhiên N được gọi là số hoàn hảo nếu tổng các ước thực sự (các ước nhỏ hơn N) lớn hơn N. Ví dụ $N = 6$; các ước thực sự là: 1, 2, 3, tổng bằng 1 + 2 + 3 = 6, vậy 6 không phải là số hoàn hảo. $N = 12$, các ước thực sự là 1, 2, 3, 4, 6, tổng bằng 16 > 12, vậy 12 là số hoàn hảo.

Yêu cầu: Tính tổng các số hoàn hảo trong đoạn $[a;\ b]$.

Dữ liệu cho trong file **SumPerfectNum.Inp** gồm 2 số nguyên dương a, b $(1 \le a \le b \le 10^5)$.

Kết quả ghi ra file **SumPerfectNum.Out** là tổng các số hoàn hảo trong đoạn $[a;\ b]$.

Ví dụ:

SumPerfectNum.Inp	SumPerfectNum.Out
10 13	12

7☼. Tổng bình phương (*Đề thi vào Chuyên Tin Phan Bội Châu - 2020*)

Lại là tính tổng! Cô giáo yêu cầu Thắng phải hoàn thành tính tổng bình phương, nhưng do thời gian nghỉ dịch CoVid-19 kéo dài, Thắng quên kiến thức về lĩnh vực này nên nhờ đến tài năng của các bạn với bài toán như sau:

Cho số nguyên dương N ($N \le 10^5$).

Yêu cầu: Tính tổng $S(N) = 1^2 + 2^2 + .. + N^2$.

Dữ liệu cho trong file **TONGBP.INP** gồm:

- Dòng đầu ghi số nguyên dương T là số test.
- Tiếp theo là T dòng, mỗi dòng ghi một số nguyên dương N.

Kết quả ghi ra file **TONGBP.OUT** gồm T dòng tương ứng là T tổng $S(N)$ ứng với T giá trị N.

Ví dụ:

TONGBP.INP	TONGBP.OUT
2 3 14	14 1015

Giới hạn:

- Có 80% số test ứng với $T, N \le 1000$.
- Có 20% số test ứng với $T, N \le 100000$.

8☼. (*Đề thi Tin học trẻ Nghệ An 2020 – THCS*) **Số các số chính phương – Numsquare.Cpp**

Cho hai số nguyên dương a, b. Tìm số các số chính phương x thuộc đoạn $[a;\ b]$.

Dữ liệu cho trong file **Numsquare.Inp** gồm hai số nguyên dương a, b ($1 \le a \le b \le 10^{18}$).

Kết quả ghi ra file **Numsquare.Out** là số các số chính phương x thuộc đoạn $[a;\ b]$.

Ví dụ:

Numsquare.Inp	Numsquare.Out
1 5	2

9☼. Chia đoạn có chứa số nguyên tố

Cho dãy số A gồm N nguyên dương $A_1, A_2, .., A_N$ ($A_i \le 2 \times 10^9$). Ta nói một cách chia dãy A được gọi là hợp lệ nếu dãy A được chia làm hai dãy $[A_1, A_2, .., A_k]$, $[A_{k+1}, .., A_N]$ và mỗi dãy phải có ít nhất một số nguyên tố.

Yêu cầu: Tính xem có bao nhiêu cách chia hợp lệ.

Dữ liệu cho trong file **SplitTwoArr.Inp** gồm:

- Dòng 1 ghi số nguyên dương N là số các số hạng của dãy A.
- Dòng 2 ghi N số nguyên dương $A_1, A_2, .., A_N$ ($A_i \le 2 \times 10^9$).

Kết quả ghi ra file **SplitTwoArr.Out** là số cách chia hợp lệ.

Ví dụ:

SplitTwoArr.Inp	SplitTwoArr.Out	Giải thích
4 1 3 4 7	2	Có 2 cách chia hợp lệ. - [1,3], [4, 7]. - [1,3, 4], [7].

10☼. Cắt gỗ

Anh Tôm có một khúc gỗ chiều dài L (mm). Em Tép đã cắt khúc gỗ này thành 2 đoạn có độ dài X và Y ($L = X + Y$). Anh Tôm thấy làm không hài lòng với việc em Tép cắt khúc gỗ ra.

Nhưng sự việc đã xảy ra nên Tôm chẳng làm gì khác được. Sau một hồi bực tức, Tôm liền đố Tép, bây giờ làm sao cắt 2 đoạn này thành các đoạn có độ dài bằng nhau và độ dài của mỗi đoạn sau khi cắt là dài nhất.

Em Tép nhỏ nhưng lập trình rất giỏi nên đã lập trình một chương trình máy tính để đưa ra độ dài lớn nhất đó.

Yêu cầu: Bạn hãy lập trình để tìm kết quả và đối chiếu với em Tép nhé.

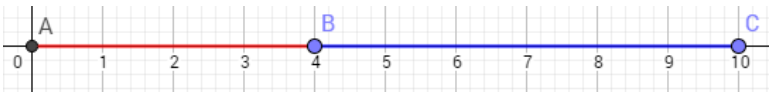
Dữ liệu cho trong file **CutWord.Inp** gồm hai số nguyên dương X, Y ($X\ !=Y$).

Kết quả ghi ra file **CutWord.Out** là độ dài lớn nhất có thể cắt được.

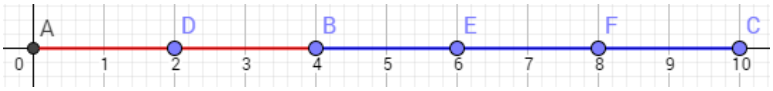
Ví dụ:

CutWord.Inp	CutWord.Out
4 6	2

Trước khi cắt:



Sau khi cắt:



Giới hạn:

- 50% số test ứng với $X, Y \leq 10^6$;
- 50% số test ứng với $X, Y \leq 10^{16}$;

11☼. Chênh lệch trước – DiffPre.Cpp

Cho dãy số nguyên $A_1, A_2, .., A_N$. Với mỗi số hạng A_i (i = 2, 3, .., N), ta gọi độ chênh lệch trước của A_i là giá trị lớn nhất của hiệu $A_i - A_t$ với $t = 1, 2, .., i - 1$.

Yêu cầu: Tính độ chênh lệch trước của $A_2, A_3, .., A_N$.

Dữ liệu cho trong file **DiffPre.Inp** gồm:

- Dòng đầu ghi số nguyên dương N là số các số hạng của dãy.
- Dòng sau ghi N số nguyên $A_1, A_2, .., A_N$ ($|A_i| \leq 10^9$).

Kết quả ghi ra file **DiffPre.Out** gồm $N - 1$ dòng tương ứng là độ lệch trước của $A_2, A_3, .., A_N$. Mỗi giá trị được ghi trên một dòng.

Ví dụ:

DiffPre.Inp	DiffPre.Out
4 1 0 4 2	-1 4 2

Giới hạn:

- 50% số điểm ứng với số test $N \leq 10^3$;
- 50% số điểm ứng với số test $N \leq 10^5$;

12☼. *(Hard)* Lưới số

Cho một bảng gồm 10^{12} dòng và 10^{12} cột. Các dòng được đánh số từ 1 đến 10^{12} (từ trên xuống dưới), các cột được đánh số từ 1 đến 10^{12} (từ trái sang phải). Ô ở dòng i, cột j có ghi một số nguyên $i \times j$. Giả sử bạn đang ở ô (1, 1), bạn cần di chuyển đến ô có ghi một số nguyên N. Tính xem, bạn cần di chuyển ít nhất bao nhiêu bước. Biết rằng khi đang ở ô (i, j), bạn chỉ có thể di chuyển sang ô $(i + 1, j)$ hoặc $(i, j + 1)$.

Dữ liệu ghi trong file **NetNum.Inp** gồm số nguyên dương N.

Kết quả ghi ra file **NetNum.Out** là số bước di chuyển ít nhất để có thể đến được ô có ghi số nguyên dương N.

Ví dụ:

NetNum.Inp	NetNum.Out

10	5

1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

Giải thích: Di chuyển qua các ô: (1, 1) → (1, 2) → (1, 3) → (1, 4) → (1, 5) → (2, 5).