Назив проблема: Упити

Временско ограничење: 0.5 секунди
Меморијско ограничење: 64 MB

Текст проблема

Мали Жика има N кутија бомбона које је поређао у ред и нумерисао, редом, бројевима од 1 до N. Нажалост, све кутије су на почетку празне. С времена на време, мајка му дода одређени број бомбона у одређену кутију. Такође, с времена на време, он се запита следеће питање: колико у датом тренутку укупно има бомбона у кутијама чији су редни бројеви између нека два одређена броја која је у том тренутку замислио.

Помозите му да успешно одговори на сва своја питања.

Описи функција

Потребно је да имплементирате следеће 3 функције:

- $Init(N)$ – ова функција се позива само једном на почетку програма и означава да мали Жика има тачно N кутија. Можете је искористити да поставите почетне вредности својих глобалних променљивих/низова.
- $AddValue(pos, val)$ – ова функција означава да се кутији са редним бројем pos додаје val бомбона. Ова функција се позива више пута у току програма са потенцијално различитим параметрима.
- $CalculateSum(L, R)$ – ова функција представља Жикино питање "колико се бомбона у овом тренутку укупно налази у кутијама са редним бројевима $L, L+1, L+2, \ldots, R$"; потребно је одговорити на питање тј. одговор вратити као вредност функције. Ова функција се позива више пута у току програма са потенцијално различитим параметрима.

Пример:

У следећој табели је приказан могући редослед позива ваших функција:

Позив функције	***Опис***
Init(10);	Добијате информацију (само једном на почетку) да Жика има 10 кутија бомбона.
CalculateSum(2, 5);	Потребно је пронаћи укупан број бомбона међу кутијама из сегмента [2,5]. Треба вратити 0.
AddValue(3, 1);	Додајемо 1 бомбону у кутију број 3
CalculateSum(2, 5);	Сада је укупан број бомбона међу кутијама из сегмента [2, 5] једнак 1 и ту вредност треба вратити.
AddValue(7, 100);	Додајемо 100 бомобна у кутију број 7.
CalculateSum(3, 9);	Потребно је вратити 101 јер толико бомбона има у сегменту [3, 9].

Ограничења

- Функција $Init(N)$ се позива тачно једном (на почетку) и важи $1 \leq N \leq 100.000$
- Функције $AddValue$ и $CalculateSum$ се **укупно** позивају не више од 100.000 пута.
- Приликом сваког позива $AddValue(pos, val)$ важи $1 \leq pos \leq N$ и $1 \leq val \leq 10^9$
- Приликом сваког позива $CalculateSum(L, R)$ важи $1 \leq L \leq R \leq N$.

Подзадаци

- **ПОДЗАДАТАК 1 [9 ПОЕНА]**: $N = 1$, функције $AddValue$ и $CalculateSum$ се укупно позивају не више од 10 пута, у свакој кутији у било ком тренутку неће бити више од 1.000 бомбона.

- **ПОДЗАДАТАК 2 [16 ПОЕНА]**: Функција $CalculateSum$ се позива највише 100 пута.

- **ПОДЗАДАТАК 3 [27 ПОЕНА]:** Функција $AddValue$ се позива највише 100 пута.

- **ПОДЗАДАТАК 4 [48 ПОЕНА]:** Нема додатних ограничења.

Детаљи имплементације

Потребно је да пошаљете тачно један фајл, под називом **upiti.c**, **upiti.cpp** или **upiti.pas**, који имплементира горе поменуте функције. Осим тражених функција, ваш фајл може садржати и додатне глобалне променљиве, помоћне функције и додатне библиотеке. Уколико радите у C/C++-у, потребно је на почетку фајла ставити **#include "upiti.h"** а уколико радите у Pascal-у, потребно је на почетку фајла ставити **Unit upiti;** (**ово је већ додато у фајловима који су вам обезбеђени**).

Зависно од програмског језика који користите, ваше функције/процедуре морају бити следећег облика:

C/C++	void Init(int N);
	void AddValue(int pos, int val);
	long long CalculateSum(int L, int R);

Pascal	procedure Init(N : longint);
	procedure AddValue(pos, val : longint);
	function ClaculateSum(L, R : longint) : int64;

Параметри функција/процедура су раније описани. Обратити пажњу да функција $CalculateSum$ враћа 64-битну вредност.

Тестирање и експериментисање

Уз задатак, обезбеђени су вам "template" фајлови (upiti.c, upiti.cpp, upiti.pas) које можете користити и мењати по потреби. Такође су вам обезбеђени програми (grader.c, grader.cpp, grader.pas) који служе да лакше тестирате кодове. Ови програми учитавају са стандардног улаза следеће податке:

- У првом реду бројеве N и Q, раздвојене размаком; N је број кутија а Q укупни број упита (не рачунајући $Init$).

- У наредних Q редова налазе се по три природна броја t, x, y; уколико је $t = 1$, тада тренутни ред представља упит $AddValue(x, y)$ а уколико је $t = 2$, тада тренутни ред представља упит $CalculateSum(x, y)$.

На почетку се позива ваша функција $Init$ са параметром N а затим се за сваки учитани упит позива ваша одговарајућа функција из одговарајућег фајла (upiti.c, upiti.cpp, upiti.pas) са учитаним параметрима. После сваког позива вашој функцији $CalculateSum$, резултат који она враћа се исписује на стандардни излаз. И кодове ових програма можете мењати по потреби.