Кратко обяснение на условието.

Заявка от първи тип: за интервал от масива се изпалнява swap (два последователни елемента на масива)

Заявка от втори тип: сумата на интервала [l; r]

Как решаваме заявки от втори тип:

Използваме стандартна функция void con() (нея си пишете в struct Node{...}; и я викате в краищата на Split(), Merge() и Split\_sz() ) където заедно с изчислението на sz на дървото се изчислява и сумата, която е отговорът на заявката).

Примерен код:

**void con(){**

**sz = 1;**

**sum = key;**

**if (l){**

**sz += l->sz;**

**sum += l->sum;**

**}**

**if (r){**

**sz += r->sz;**

**sum += r->sum;**

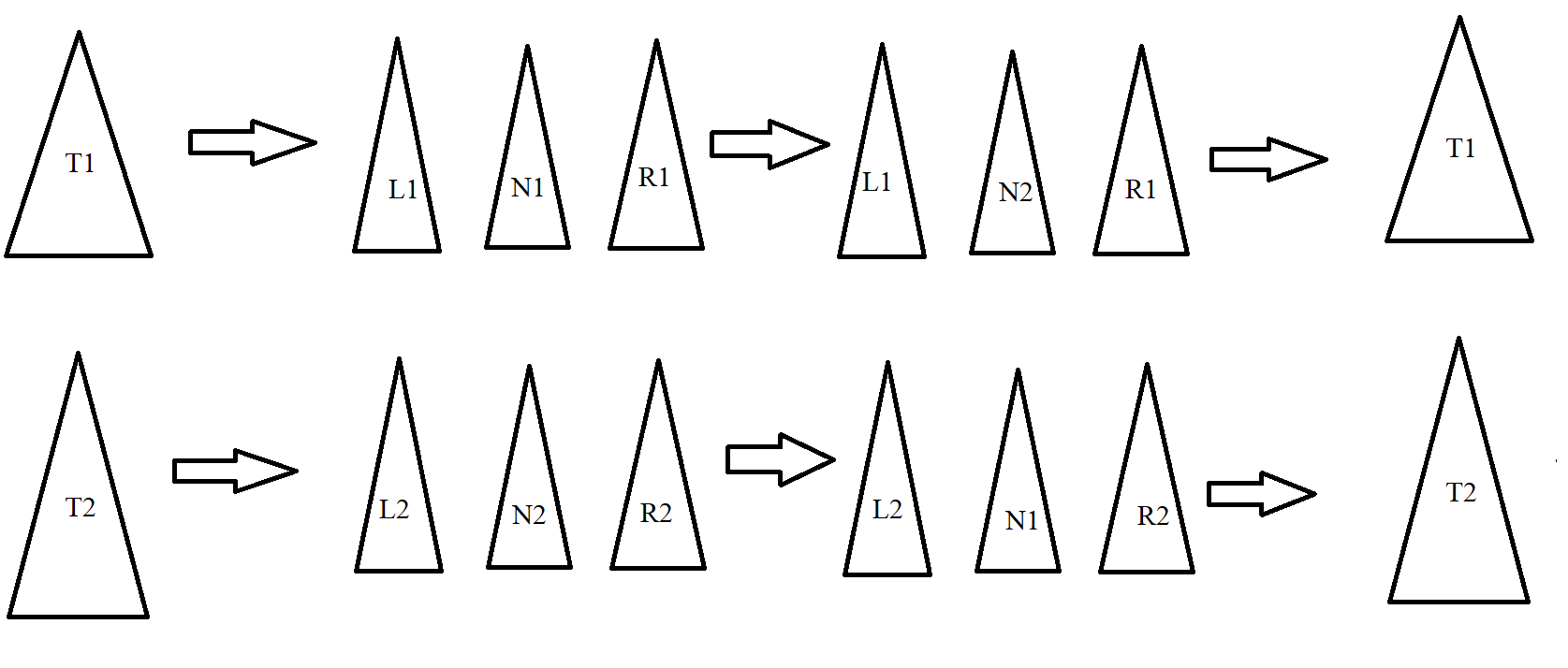
**}**

**}**

**};**

Как решаваме заявки от втори тип:

За да можем ефикасно да бъдат изпълнявани трябва да пазим масива не в един трийп, а в два такива: един за нечетните и един за четните позиции. По този начин за интервала от [l; r] отцепваме нужните ни две части от двата трийпа (всяка за O(logN)) и ги сменяме.



Заявките от втори тип ще се правят по същия начин (с две дървета), като им сумираме отговорите.