

Génebesz

Kutatók DNS szekvenciákat vizsgálnak, amelyeket különböző műveletekkel manipulálnak. Minden DNS szekvencia olyan karaktersorozattal adott, amely csak az A, C, G vagy T karaktereket tartalmazza.

Az ilyen műveletek elvégzését kell megoldanod a következő könyvtári műveletek megvalósításával. Az aktuális DNS szekvencia karaktersorozat elemeit az $1, \dots, n$ pozíciókkal azonosítjuk, ha a sorozat n elemet tartalmaz. Csak olyan művelet elvégzését kéri, amely az adott paraméterekkel elvégezhető.

Könyvtár

1. `void Kezd(string s)` : az értékelő főprogram egyszer hívja a program elején, az s paraméter adja meg a kezdeti DNS szekvenciát.
2. `void Beszur(int i, char x)` : az x karaktert beszúrja az i . pozíció után ($0 \leq i \leq n$).
3. `void Mutal(int i, char x)` : az i . ($1 \leq i \leq n$) pozícióban lévő karaktert x -re cseréli.
4. `void Kivag(int i, int j)` : az aktuális DNS szekvenciából törli az $i, \dots, j$ pozícióban lévő elemeket ($1 \leq i \leq j \leq n$).
5. `int Szamlal(int i, int j, char x)` : az i, j ($1 \leq i \leq j \leq n$) pozíciókkal meghatározott részsorozat x -el egyező elemeinek számát adja.
6. `string Eredmeny()` : az aktuális, tehát az eddigi műveletek által eredményezett DNS szekvenciát adja. Ezt a függvényt az értékelő többször is hívhatja.

A programod **nem írhat és nem olvashat** semmilyen fájlt!

A programodnak tartalmaznia kell az `#include "grader.h"` importáló sort és nem tartalmazhat `main` függvényt! Egyik művelet végrehajtása sem eredményezheti a program terminálását!

Gyakorlás

A letölthető `minta.zip` egy minta `grader` programot tartalmaz és egy `feladat.cpp` fájlt, amiben a fenti könyvtári műveleteket implementációját kell megvalósítanod. A fejlesztői környezetben amit használsz, add a projekthez a `grader.h`, `grader.cpp` és `feladat.cpp` állományokat! (Más állományt ne tartalmazzon a projekt!)

A program a standard bemenetről olvassa az elvégzendő műveleteket, soronként egyet. Minden sorban az első szám az elvégzendő művelet sorszáma, ezt követik a művelet paraméterei (egy-egy szóközzel elválasztva, a fent megadott sorrend szerint). A 0 sorszámú művelet a program befejezését eredményezi. A minta `grader` program nem ellenőrzi a megoldás helyességét.

A `ki1.txt` és a `ki2.txt` a `be1.txt`, illetve a `be2.txt` műveletsor hatására keletkező helyes `Eredmeny` és `Szamlal` értékeket tartalmazza.

Csak a `feladat.cpp` állományt kell beadnod.

Korlátok

$1 \leq N \leq 100\,000$ a kezdeti DNS szekvencia hossza.

$0 \leq K \leq 100\,000$ az elvégzendő műveletek száma.

Az **Eredmeny** műveletet legfeljebb 100 alkalommal hívják.

Időlimit: 0.2 s

Memória limit: 64 MiB

Pontozás

Részfeladat	Pontok	Korlátok
1	0	a példa
2	30	nincs sem Beszur sem Kivag művelet.
3	30	nincs Kivag művelet.
4	20	a kezdeti DNS szekvencia hossza $\leq 50\,000$, a műveletek száma $\leq 50\,000$
5	20	nincs egyéb korlát.