**Задача АB1. GCD**

Радо и Емил играят една игра. Радо си намисля едно число между и , а Емил трябва да го познае, като многократно пита въпроси от вида: „Какъв е най-големият общ делител на и , където и ?“ Целта на Емил е да познае числото с възможно най-малък среден брой заявки (двамата ще играят играта много пъти). Знае се, че Радо си избра по напълно произволен начин, като всяко число между и е еднакво вероятно да бъде избрано, т.е .

По принцип Емил лесно би се справил с такава игра, но напоследък е много зает с работата си и няма време за задачи. Помогнете му като вместо него напишете програма gcd.cpp, която да играе тази игра. Тази програма ще се компилира с програма на журито, която ще играе от страната на Радо.

**Детайли по имплементацията**

Вие трябва да имплементирате функция play със следния прототип:

ull play(ull MAX\_X);

Тя ще бъде извикана много пъти (винаги с еднакво ). Функцията трябва да върне стойността на , която Радо си е намислил. За тази цел, тя може многократно да вика функцията query на журито. Тя има следния протип:

ull query(ull a, ull b);

Тази функция ще отговаря на гореописаните заявки, т.е. ще връща .

Вашата програма не трябва да съдържа функция main, да чете от стандартния вход или да печата на стандартния изход. Програмата Ви също така трябва да включва хедър файла gcd.h (чрез указание към препроцесора #include "gcd.h"), където ull е дефинирано като unsigned long long. Стига да спазва тези условия, програмата Ви може да съдържа каквито и да е помощни функции, променливи, константи и прочие.

**Ограничения**

**Оценяване**

Всеки тест се оценява поотделно. За да получите точки за даден тест, програмата Ви трябва правилно да познае при всички викания на функцията play без да подава невалидни стойности на и към query. Функцията play ще бъде викната . Резултатът Ви за теста зависи от средния брой викания на query, които програмата Ви прави. Нека тази бройка е и нека и са константи за теста. Тогава частта от максималния брой точки за теста, която ще получите, е:

* , ако
* , ако

**Тестове**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тест | Точки |  |  |  |
| 1 | 10 |  |  |  |
| 2 | 90 |  |  |  |

**Локално тестване**

Предоставени са Ви файловете gcd.h и Lgrader.cpp, които можете да компилирате заедно с програма си, за да я тествате. При стартиране на програмата трябва да се въведе . След това решението Ви ще се изпълни и ще се изведе средната бройка заявки, която прави, или описание на грешка, ако възникне такава. Грейдърът на оценяващата система ще се държи по същия начин като предоставения Ви локален грейдър.

**Примерна комуникция**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Действия на play | Действия на грейдъра | Обяснение |
| 1. |  | play(10) |  |
| 2. | query(0, 15) | return 3 |  |
| 3. | query(2, 8) | return 8 |  |
| 4. | return 6 |  |  |
| 6. |  | play(10) |  |
| 7. | query(1, 7) | return 1 |  |
| 8. | query(12, 10) | return 2 |  |
| 9. | query(0, 10) | return 10 |  |
| 10. | return 0 |  |  |