XXXV НАЦИОНАЛНА ОЛИМПИАДА ПО ИНФОРМАТИКА
Национален кръг
Стара Загора, 15 – 17 март 2019 г.
Група АВ, 9 – 12 клас, Ден 2

Задача АВ6. СЪОБЩЕНИЕ

Марсоходът Опъртюнити иска да изпрати съобщение до Земята. Проблемът е, че уредите му са повредени и части от съобщението няма успешно да се изпратят. По-конкретно, информацията, която той желае да достигне Земята, се състои от N бита. Марсоходът може да изпрати съобщение с избрана от него дължина M бита. Още преди изпращането на съобщението е известно, че точно D бита от него ще бъдат изгубени. Марсоходът иска да изпрати такова съобщение, че на Земята да могат да възстановят максимално точно полезната информация от N бита, която той желае да предаде.

Един от ръководителите на мисията Ви е помолил за помощ – да измислите и имплементирате протокола за комуникация. Трябва да напишете функция *transmit*, която Опъртюнити ще използва. Тя получава низ от N бита и стойността на D и трябва да генерира друг низ с дължина M – това са данните, които марсоходът ще опита да излъчи. Също така трябва да напишете и отделна функция *receive*, която NASA ще използва на Земята. Тя получава низ от $M - D$ бита – съобщението генерирано от *transmit* след загубата на някои от битовете, както и стойностите на N и на D. Тя трябва да реконструира оригиналните данни от N бита максимално точно.

Целта е двойката функции да максимизират отношението на успешно реконструирана смислена информация (а не просто шум) към общото количество успешно изпратена информация. Това ще се оценява по следната формула ($correct$ е броят правилно познати битове):

$$\frac{2 \times correct - N}{\max(M - D, N)}$$

Детайли по имплементацията

Функция *transmit* трябва да има следния прототип:

```
std::vector<bool> transmit(const std::vector<bool>& data, int d);
```

Тя ще бъде извикана точно веднъж и ще получи като аргументи данните за изпращане и бройката битове, които ще бъдат изтрити, като трябва да върне съобщението за изпращане.

Функцията *receive* трябва да има следния прототип:

```
std::vector<bool> receive(const std::vector<bool>& message, int n, int d);
```

Тя също ще бъде извикана точно веднъж и ще получи като аргументи полученото на Земята съобщение и стойностите на N и D, като трябва да върне вектор с дължина точно N – реконструираното с някаква точност съобщение.

Двете функции задължително трябва да бъдат имплементирани в отделни файлове **transmit.cpp** и **receive.cpp**, които трябва да изпратите към системата (за тази задача системата предоставя възможност да се изпращат два файла). В тях може да имате каквито искате помощни функции, структури, променливи и т.н. Силно препоръчително е всички такива (т.е. всичко без самите функции *transmit* и *receive*) да бъдат декларирани като *static* с цел да се предотвратят конфликти в имената между двата ваши файла). Те само не трябва да съдържат функция *main* и трябва да включват хедър файла **transmission.h** чрез указание към препроцесора #include “transmission.h” в началото.

XXXV НАЦИОНАЛНА ОЛИМПИАДА ПО ИНФОРМАТИКА
Национален кръг
Стара Загора, 15 – 17 март 2019 г.
Група АВ, 9 – 12 клас, Ден 2

ВАЖНО: Двете функции ще бъдат изпълнявани в различни процеси на системата и НЕ могат да обменят информация пряко помежду си.

Ограничения

$N = 10\,000$ във всички тестове
$1 \leq D \leq N/10$
$M \leq 100\,000$

Оценяване

Всеки тест се оценява отделно. За даден тест Вашето решение ще получи точки, различни от 0, ако двете функции успешно приключат изпълнение, функцията *transmit* върне вектор с дължина не по-голяма от 100 000 и функцията *receive* върне вектор с вярна дължина и е познала над ½ от битовете. Точките, които ще получите на теста са равни на максималният брой точки за теста, умножени по $\min(yourScore/authorScore, 1)$, където $yourScore$ е резултатът Ви от формулата, дадена по-горе, а $authorScore$ е резултатът на авторовото решение от същата формула. Части от тестовете имат допълнителни ограничения.

Тестове

Низовете от данни за изпращане са произволно генерирани на всички тестове бит по бит, където 1 и 0 имат равен шанс.

Кои битове ще бъдат изтрити не зависи по смислен начин от съдържанието на съобщението, което изпращате, а само от размера му, но може да се мени при различни съобщения. Позициите на триене са произволно генерирани на по групи от съседни битове с дължина K.

В 15% от тестовете: $D = 1$ и $K = 1$	**минимален резултат:** 0.971
В 25% от тестовете: $D \leq 1.1 \times \sqrt{N}$ и $K = 1$	**минимален резултат:** 0.771
В 25% от тестовете: $K = 1$	**минимален резултат:** 0.473
В 35% от тестовете: без доп. ограничения	**минимален резултат:** 0.432

За всяка група Ви е даден минималният резултат, който авторовото решение е получило на някой тест от нея, с точност до третия символ след десетичната запетая.

Локално тестване

Предоставени Ви са файловете **transmission.h** и **Lgrader.cpp**, които можете да компилирате заедно с Вашите програми, за да ги тествате, както и две примерни глупави имплементации на функциите. За да компилирате файловете заедно използвайте следната команда в конзолата/терминала (рънната в папката, в която се намират файловете):

```
g++ -O2 -std=c++11 -o transmission.exe receive.cpp transmit.cpp Lgrader.cpp
```

При стартиране програмата ще Ви пита за N, D и K и от там всичко друго ще се генерира произволно. Ако искате да я конфигурирате по друг начин, може да правите каквито си промени искате по предоставените Ви файлове.

XXXV НАЦИОНАЛНА ОЛИМПИАДА ПО ИНФОРМАТИКА
Национален кръг
Стара Загора, 15 – 17 март 2019 г.
Група АВ, 9 – 12 клас, Ден 2

Примерна комуникация

№	Викания на функции от журито	Техният return
1.	transmit({0,0,1,0,1,0},2)	{0,0,1,0,1,0}
2.	receive({0,1,1,0},6,2)	{0,0,0,0,0,0}

Обяснение на примерната комуникация

Тази примерна комуникция се е извършила с двете примерни наивни имплементации на функциите, които са Ви предоставени.

Данните за изпращане са 001010 и ще бъдат изтрити два бита. Функцията transmit не прави никакви модификации по съобщението и се опитва да изпрати просто това. Не са се изпратили успешно вторият и четвъртият бит и receive получава 0110. Тази наивна имплементация обаче изобщо не опитва да реконструира данните, ами просто връща низ от нули.

Резултатът, който двойката функции получават на теста, е $(2 \times 4 - 6)/6 = 1/3$. Точките, които ще се получат, зависят от резултата на авторовото решение.