Как я перестал бояться и полюбил LP-солверы

В некоторых задачах вам потребуется написать решения, использующие сведение к задаче линейного или смешанного программирования. Вам доступны следующие опции: вы можете либо сами написать любой доступный вам алгоритм, либо воспользоваться выбранной нами библиотекой для языков C/C++ под названием GLPK.

GLPK

GLPK — промышленная open-source библиотека, предоставляющая API для языков C и C++, и содержащая ряд процедур для численного решения задач различных общих форм, в том числе, задач линейного программирования.

Чтобы воспользоваться данной библиотекой в своём решении на языке C или C++, нужно подключить заголовочный файл `"glpk.h"` (именно так, в двойных кавычках), и послать решение в тестирующую систему с использованием компилятора Make2 и расширением файла `.c` или .cpp.

Полная документация по данной библиотеке доступна в виде pdf-документа. Рекомендуем внимательно её проглядеть, чтобы не упустить никакую из мощных фичей данной библиотеки.

Чтобы скомпилировать решение с использованием данной библиотеки локально под Linux, надо откуда-нибудь раздобыть заголовочный файл `glpk.h` и библиотеку `libglpk.so`/`libglpk.a`. Самый простой способ сделать это в debian-подобных системах — поставить пакет `libglpk-dev` командой `sudo apt-get install libglpk-dev`. Можно также собрать пакет из исходников, которые можно скачать здесь, командой `./configure && make`, и унести из него файлы `src/glpk.h` и `src/.libs/libglpk.a` в директорию со своей программой.

Чтобы подключить библиотеку в своём решении, используйте директиву `#include "glpk.h"`. Пример компилирующегося решения доступен в архиве с домашним заданием в файле `stub.cpp`.

Для того, чтобы скомпилировать своё решение, можно воспользоваться командой `g++ <name>.cpp -o <name> -lglpk [-L.]` (последний ключ нужен, если вы положили библиотеку `libglpk.a` локально).

Пара замечаний по пользованию библиотекой. Во-первых, обратите внимание, что все массивы, которые вы будете передать в процедуры, предоставляемые библиотекой, индексируются с единицы (в частности, нулевой элемент не имеет значения). Во-вторых, по умолчанию библиотека выводит много отладочной информации в стандартный вывод; если решение с её использованием отправить на проверку в тестирующую систему, то вы, скорее всего, получите вердикт «Неправильный формат вывода». Отключите отладочный вывод путём передачи в процедуру симплекс-алгоритма параметра `msg_lev`, равного `GLP_MSG_OFF`.

Отчёт

Помимо сдачи задач в тестирующую систему, необходимо предоставиить отчёт, в котором **обязательно** должны быть указаны ваш логин и ссылки на страницы ваших решений (которые имеют вид `https://contest.yandex.ru/contest/1234/run-report/42424242`). В обоих задачах требуется развёрнутый отчёт с описанием вашего подхода, удачных и неудачных идей и всего, что пожелаете нужным. В отчёте должны быть формально описаны задачи LP/IP/MIP, которые вы решаете с помощью solver'а.

Система оценки

Задача «Capacitated Warehouse Location» стоит 2 балла, а задача «Min Weight Perfect Matching» — 4 баллов. Отчёт оценивается из 25% стоимости задачи.

Задача A. Capacitated Warehouse Location

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
Ограничение по памяти:	512 мегабайт

Межгалактическая торговая корпорация Bozon проводит конкурс по оптимизации расходов. Для обслуживания своих клиентов компания содержит n складов на разных планетах. Каждый склад характеризуется своей *вместимостью* $capacity_w$, а также *стоимостью содержания* $openCost_w$.

Кроме того, у компании есть m постоянных клиентов, которым надо поставлять товары со складов. Каждый клиент характеризуется величиной своего *потребления* $demand_c$.

Каждый клиент может потенциально обслуживаться несколькими складами одновременно. В такой ситуации потребление клиентом ресурсов складов делится в какой-то пропорции между несколькими складами. Формально говоря, назовём планом потребления X такой набор неотрицательных вещественных чисел $x_{w,c}$ по всем складам w и клиентам c, что $\sum\limits_w x_{w,c} = 1$, а также, что $\sum\limits_c x_{w,c} demand_c \leqslant capacity_w$ для любого склада w (потребление ресурсов склада не должно превосходить вместимости склада). Для каждой пары из клиента c и склада w также известна величина $useCost_{w,c}$, задающая стоимость полного обслуживания клиента c складом w; в случае, если клиент c потребляет ресурсы склада w только частично, затраты уменьшаются пропорционально степени потребления.

С целью минимизации расходов, компания решила оставить только некоторое подмножество W из имеющихся складов, тем самым сократив расходы на их содержание. Разумеется, закрытые склады не могут больше обслуживать клиентов. Более конкретно, компания каждый месяц тратит деньги на два вида расходов:

1. $\sum\limits_{w \in W} openCost_w$ — затраты на поддержание открытых складов;
2. $\sum\limits_{w,c} x_{w,c} useCost_{w,c}$ — затраты на использование ресурсов складов клиентами.

Вы участвуете в конкурсе по оптимизации расходов, проводимом компанией. Результатом вашей работы должна быть пара (W, X), состоящая из множества оставленных открытыми складов и плана потребления ресурсов клиентами. Ваша задача — найти любую оптимальную пару (W, X).

Для решения данной задачи следует воспользоваться методом вариацией метода ветвей и границ под названием «Branch & Cut». Постройте формулировку данной задачи в терминах смешанного программированя (линейного программирования с дополнительными условиями целочисленности некоторых переменных), после чего воспользуйтесь средствами библиотеки GLPK, описанными в разделе «Branch & Cut» документации. Обратите внимание на дополнительные настройки, влияющие на ход работы алгоритма.

Формат входных данных

В первой строке входных данных находятся два целых числа n и m ($1 \leqslant n, m \leqslant 150$) — количество складов и клиентов соответственно.

Далее будем считать, что w пробегает диапазон от 0 до $n - 1$, а c пробегает диапазон от 0 до $m - 1$.

В последующих n строках находятся пары чисел $capacity_w, openCost_w$ ($0 \leqslant capacity_w \leqslant 10^4$, $0 \leqslant openCost_w \leqslant 10^5$) — соответственно вместимость и стоимость содержания склада w.

В следующей строке находятся m чисел $demand_c$ ($0 \leqslant demand_c \leqslant 10^4$) — величины потреблений клиентов.

Далее следует матрица $n \times m$ из чисел $useCost_{w,c}$ ($0 \leqslant useCost_{w,c} \leqslant 10^5$) — стоимости полного обслуживания каждого из клиентов на каждом из складов.

Все величины вместимостей, потреблений и стоимостей — вещественные числа, заданные с не более, чем 5 знаками после запятой.

Формат выходных данных

В первой строке выведите целое число k ($1 \leqslant k \leqslant n$) — количество оставленных открытыми складов.

Во второй строке выведите k целых различных чисел от 0 до $n-1$ в возрастающем порядке — номера оставленных складов в 0-индексации в порядке следования во входных данных.

В последующих k строках выведите матрицу размера $k \times m$, задающую план использования (матрицу из чисел $x_{w,c}$ по всем оставленным складам и всем клиентам).

Выполнение условий $\sum\limits_{w} x_{w,c} = 1$ будет проверяться с абсолютной точностью 10^{-3}. Ваш ответ будет засчитан, если его относительная ошибка относительно ответа жюри не превосходит 10^{-3}.

Пример

стандартный ввод	стандартный вывод
3 4 12 3 50 100 5 4 6 7 2 1 6 14 20 42 0 5 0 0 600 14 2 41	2 1 3 1.000000 0.714286 0.000000 0.000000 0.000000 0.285714 1.000000 1.000000

Замечание

Вам будет доступно три открытых теста, которые можно будет скачать на странице курса, первый из которых приведён в условии. В первом тесте оптимальная стоимость составляет 70, во втором — 4816.70455, а в третьем — 96931.85081.

Задача B. Min Weight Perfect Matching

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	12 секунд
Ограничение по памяти:	512 мегабайт

Дан неориентированный граф из n вершин и m рёбер. Вершины пронумерованые целыми числами от 0 до $n-1$, рёбра пронумерованы целыми числами от 0 до $m-1$. У каждого ребра есть целый положительный вес. Гарантируется, что в графе есть совершенное паросочетание. Требуется найти совершенное паросочетание минимального суммарного веса рёбер.

Для решения данной задачи следует воспользоваться GLPK. MIP-solver в составе GLPK скорее не справляется с очевидной IP-формулировкой задачи о паросочетании минимального веса (во всяком случае, мы так считаем :) Напомним, что LP-задание для политопа паросочетаний в произвольном графе имеет экспоненциальный размер.

Для подобных ситуаций в MIP-solver'ах часто используется механизм «ленивого» задания дополнительных ограничений: стартовая программа содержит не все ограничения, а только их подмножество, а «не подходящие» промежуточные решения отсекаются путём предъявления конкретных ограничений, которым они не удовлетворяют, поле чего переборная часть solver'а продолжает свою работу.

Библиотека GLPK предоставляет такую возможность через user-defined callback, о котором стоит прочитать в документации. Подумайте, какой user-defined callback вам может пригодиться. Можно сделать «честный» callback, реализующий blossom inequalities (см. сответствующую лекцию), но существует более простой способ, если проявить фантазию и реализовать некую более слабую процедуру, которая иногда порождает очевидно не выполненное неравенство, а иногда ничего не порождает.

Формат входных данных

В первой строке заданы два целых числа n и m ($2 \leqslant n \leqslant 500$, n чётно, $1 \leqslant m \leqslant 1000$).

В последующих m строках заданы рёбра графа. Каждое ребро задаётся тройкой целых чисел a_i, b_i, w_i ($0 \leqslant a_i, b_i \leqslant n-1$, $a_i \neq b_i$, $1 \leqslant w_i \leqslant 1000$), где a_i и b_i — номера вершин, соединяемых ребром, а w_i — его вес.

Формат выходных данных

В первой строке выведите минимально возможный вес совершенного паросочетания.

Во второй строке выведите $n/2$ целых чисел — номера рёбер в совершенном паросочетании минимального веса. Рёбра пронумерованы от 0 до $m-1$ в порядке следования во входных данных.

Примеры

стандартный ввод	стандартный вывод
4 6 0 1 1 0 2 9 2 1 3 1 3 6 3 0 8 2 0 4	10 5 3
6 7 0 1 1 1 2 1 2 0 1 2 3 100 3 4 1 4 5 1 5 3 1	102 0 3 5

Замечание

Вам будет доступно три открытых теста, которые можно будет скачать на странице курса, первые два из которых приведены в условии задачи. Третий тест «тяжёлый», вес искомого паросочетания в нём составляет 116688.