VI Поволжская олимпиада по информационным технологиям среди студентов и аспирантов «Волга ИТ – 2013»

Номинация «Прикладное программирование»

Задание финального этапа

Первая часть

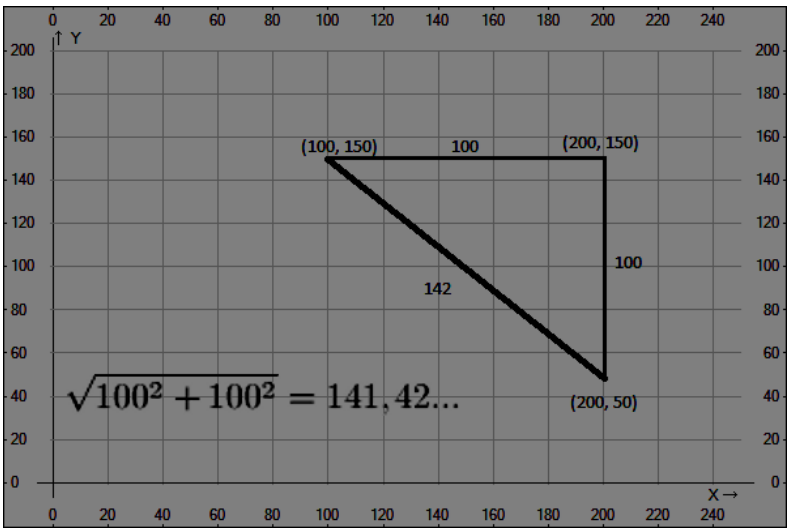
Введение

Грузоперевозочная компания, расположенная по адресу г. Ульяновск, Набережная реки Свияги, офис 3/118, столкнулась с проблемой. Оказывается, многие водители при перевозке груза выбирают не самые оптимальные маршруты. На коротком совещании руководства было принято решение по написанию программного продукта, способного оптимизировать передвижение грузового транспорта и показывать кратчайшие пути от одного города к другому.

Постановка задачи

На плоской поверхности расположены города, соединенные дорогами. По дорогам едут грузовики. Необходимо для каждого грузовика найти кратчайший путь от его начальной точки до конечной. Каждый грузовик движется с постоянной скоростью и начинает с некоторым количеством бензина. На каждую единицу пройденного пути тратится одна единица бензина. Гарантируется, что будет существовать по меньшей мере один маршрут и на его преодоление хватит топлива.

Расстояние между городами не может быть дробным числом, поэтому оно вычисляется как наименьшее целое, которое не меньше чем расчетное по теореме Пифагора расстояние между двумя точками (см. рисунок ниже). Например, расстояние между городами с координатами (100, 150) и (200, 50) будет:
round(ceil(sqrt((200-100)^2 + (150-50)^2))) = round(ceil(141,4...)) = 142

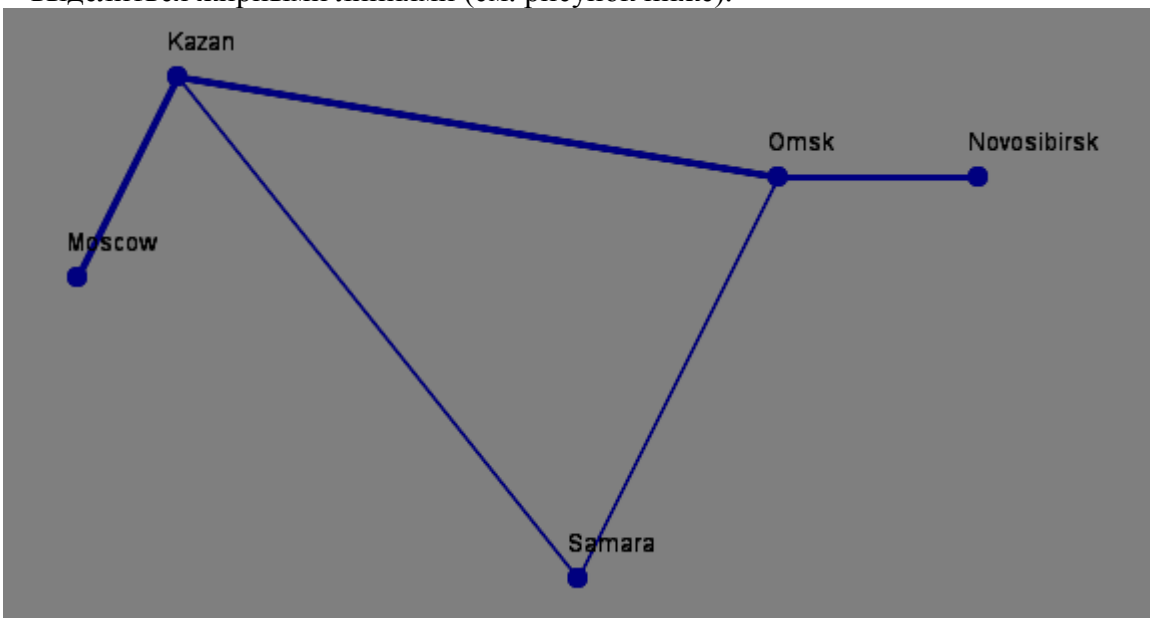


Проект предоставляется в виде двух исполняемых файлов. Приложения должны загрузить данные из файла input.txt, рассчитать и вывести маршруты для каждой грузовой машины

- в файл output.txt не позднее чем через 3 секунды после запуска (первый исполняемый файл)
- на экран (второй исполняемый файл)

Требованию к графическому интерфейсу:

- Карта области, размером 1024x512 пикселей. Масштаб отображения должен подобраться автоматически, что бы подходить под требуемые размеры с сохранением соотношения сторон.
- Города должны отображаться точками. Дороги — линиями. Каждый город должен быть подписан своим названием.
- Слева от карты находится список водителей.
- При выборе грузовика из списка все дороги оптимального маршрута должны выделиться жирными линиями (см. рисунок ниже).



На программу наложено ограничение по количеству потребляемой памяти в 1 ГБ.

Входные данные

Файл input.txt имеет кодировку ASCII и следующий вид:

W H	Ширина и высота карты $100 \le W, H \le 10`000$
N	Количество городов $1 \le N \le 100$
City-i Xi Yi	Координаты (x, y) и название каждого города City-i – одно слово, состоящее исключительно из цифр и английских букв обоих регистров; каждое слово уникально. $1 \le Xi \le W$ $1 \le Yi \le H$ $1 \le i \le N$
M	Количество дорог $1 \le M \le 1`000$
A-i B-i	Порядковые номера городов Ai и Bi, соединенных i-ой дорогой $1 \le Ai, Bi \le N$ $1 \le i \le M$
K	Количество грузовиков $1 \le K \le 10$
Name-i A1-i B2-i Speed-i Fuel-i	i-ый грузовик, управляемый водителем с именем Name-i, следует из пункта с номером A1-i в пункт B2-i со скоростью Speed-i и начальным количеством бензина Fuel-i Name-i – одно слово, состоящее исключительно из цифр и английских букв обоих регистров; каждое слово уникально. $1 \le Ai, Bi \le N$ $1 \le Speed\text{-}i \le 40$ $1 \le Fuel\text{-}i \le 10`000$ $1 \le i \le M$

Пример:

```
1024 512
5
Moscow 50 200
Kazan 100 100
Samara 300 350
Omsk 400 150
Novosibirsk 500 150
5
1 2
2 3
3 4
4 5
2 4
2
Petrov 1 5 5 1000
Sidorov 5 3 3 1000
```

Выходные данные

Файл output.txt должен иметь кодировку ASCII и принимать следующий вид:

Fi Ti Ai-1 Ai-2 …	Количество бензина, которое останется у i-ого грузовика на конец поездки, время прибытия и список номеров городов, которые он проехал для каждого i на новой строке. $1 \le i \le N$

В случае, если лимит времени исчерпан, алгоритм должен прекратить свое выполнение и вывести в выходной файл все маршруты (не обязательно полные), которые были найдены за отведенное время, начиная с первого и заканчивая последним найденным.

Разрешается ставить лишние пробелы в конце строк и пустые строки в конце файла.

Пример:
483 104 1 2 4 5
676 109 5 4 3

Дополнительные условия

Программа предоставляется в виде двух исполняемых файлов, исходных текстов, файлов проекта для среды разработки и readme.txt.

В файл readme.txt включите

- Краткое описание алгоритма
- Инструкцию по сборке программы из исходных текстов (дополнительные условия, настройки среды - постарайтесь свести их к минимуму)
- Требования для запуска исполняемого файла (если не запустится, мы попробуем собрать заново из исходных текстов, но это минус)

Принимаются частичные решения: GUI реализован частично или отсутствует, отсутствие ограничения на время работы алгоритма, и пр. В таком случае в файле readme.txt необходимо указать какие ограничения наложены на программу.

Оценка

Итоговая оценка будет складываться из результатов прогона на тестовых задачах (оцениваться будет степень близости ответа программы к ожидаемому ответу), результатов ручного тестирования GUI и, возможно, анализа исходных текстов программы.

Вторая часть

Введение

Через некоторое время состоялось второе совещание руководства. На нем было принято решение по расширению разрабатываемого продукта путем введения правил для каждого грузовика. Так же теперь ведется учет перевозимого товара.

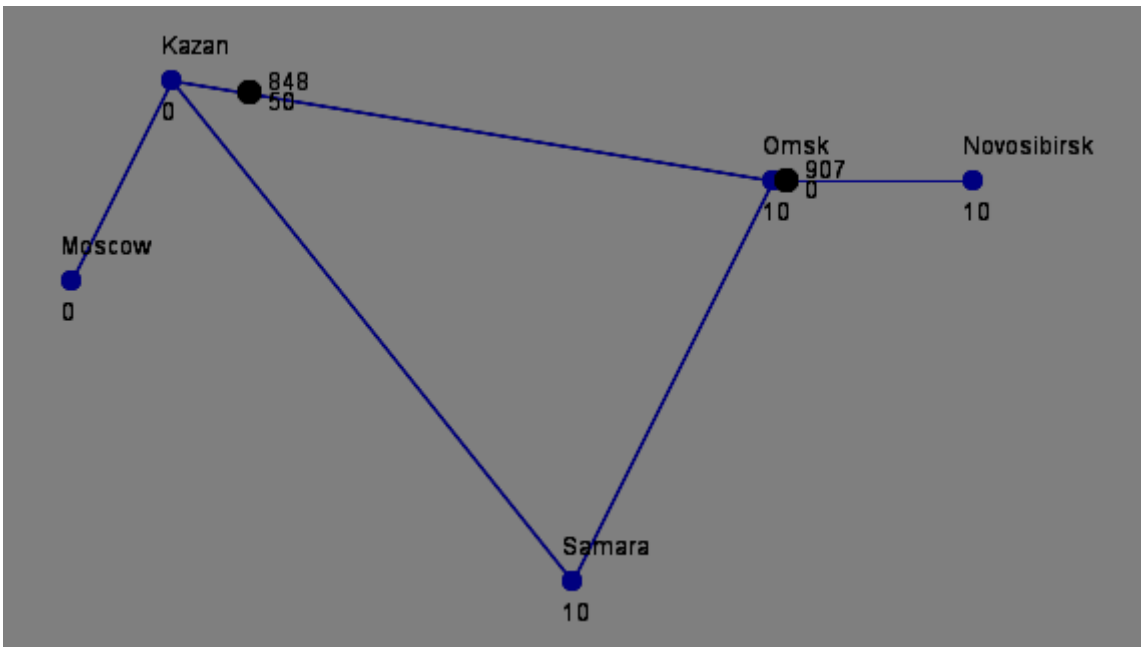
Постановка задачи

О правилах:
У каждого водителя есть программа поведения, заданная правилами в файле driver-name.txt (name – имя водителя). Маршрут из начального в конечный город теперь задается только правилами. Каждое правило содержит одно или несколько условий и одно или несколько действий, которые выполняются при удовлетворении всех условий. Правила проверяются только в городе при отсутствии дальнейшего плана движения (например, если машина движется из A в C через B, то следующая проверка будет выполнена в городе C, но не в B). Выполняется только одно правило, удовлетворяющее всем условиям, находящееся первым в общем списке. Если ни одно из правил не выполнилось, то ход пропускается. Выполнение правил и действий не занимает ходов. При достижении конечного города проверка правил заканчивается.

О ресурсах:
В каждом городе есть товарная станция с некоторым заранее заданным количеством товара. Машины могут как загружать товар, так и выгружать его. Если грузовику требуется загрузить товар, но его на складе недостаточно, то водитель ждет его привоза и тронется на следующих ход. Если в городе ожидает несколько машин, то их ожидание закончится в момент, когда товара будет достаточно всем машинам сразу. При достижении конечного города все товары разгружаются автоматически.

Гарантируется корректное задание правил, отсутствие циклов и бесконечных ожиданий товара. Количество товара на складе в любой момент времени не будет превышать значения в 1`000`000 и быть меньше 0. Бензина хватит на поездку по оптимальному маршруту.

Дополнительные требования к графическому интерфейсу:
Реализовать кнопку «Демонстрация» при нажатии на которую происходит динамическое движении всех машин по дорогам. Скорость отображения анимации: 1 ход в 50 миллисекунд. Машина отображается в виде большого круга. Справа от маркера машины отображается текущее количество бензина (обновляется по мере движения) и количество перевозимого товара. Ниже маркера города показывается количество товара в нем. См. рисунок ниже.



Внимание!

Рекомендуем сохранить приложение с первого этапа в отдельном каталоге (как исходники, так и исполняемые файлы).

Входные данные

Файл stations.txt имеет кодировку ASCII и следующий вид:

P	Количество товарных станций с ненулевым количеством товара. Все остальные станции имеют начальное количество товара равное 0. $1 \le P \le N$
Ai Goods-i	i-ая товарная станция находится в городе с номером Ai с количеством товара Goods-i. $1 \le Ai \le N$ $0 \le Goods\text{-}i \le 1`000`000$ $1 \le i \le P$

Файлы driver-name.txt (name – имя водителя) имеют кодировку ASCII содержат программу поведения соответствующих водителей:

if	Ключевое слово
<условие> ...	Одно или несколько условий. Условия объединяется через логическое И.
then	Ключевое слово
<действие> ...	Одно или несколько действий
endif	Ключевое слово
if ...	Следующее правило общим счетом не более 20.

Список условий:

fuel < X	Количество бензина у машины строго меньше X
goods >= X	Количество товара на станции больше или равно X
time = X	Номер текущего хода равен X
city is X	Машина находится в городе X

Список действий:

refuel X	Заправится на X, но не больше, чем 10`000
load X	Загрузить X товара со станции в текущем городе
unload X	Выгрузить X товара на станцию в текущем городе
goto X	Отправится в город X по кратчайшему маршруту. Достаточность топлива гарантируется.

Пример stations.txt:
4
1 50
3 10
4 10
5 10

Пример driver-Petrov.txt:
if
city is Moscow
then
load 50
goto Omsk
endif
if
city is Omsk
then
unload 25
goto Novosibirsk
endif

Пример driver-Sidorov.txt:
if
city is Novosibirsk
then
goto Omsk
endif
if
city is Omsk
goods >= 20
then
load 20
goto Samara
endif

Выходные данные

В конец файла output.txt добавляется следующая информация

Goods-i	Количество товара в каждом городе из списка, данного в stations.txt Для каждого i на новой строке. $0 \leq i \leq P$

Пример:

483 104 1 2 4 5
676 159 5 4 3
0 30 15 35

Волга ИТ - 2013. Прикладное Программирование

Очный тур, Третий этап

Введение

Прошло еще некоторое время. Группой аналитики подготовлены списки правил всех водителей компании. Однако вот ведь дело — в правилах могут быть ошибки! Ручная проверка займет слишком много времени. Необходимо как можно быстрее отсеять неверные списки правил, пока программа не вошла в эксплуатацию и не привела компанию к огромным убыткам!

Постановка задачи

Необходимо обнаруживать следующие ситуации:

- Машина движется по кругу и не может из него выйти.
- Машина бесконечно ожидает товар на складе, который не может быть туда доставлен.

Разрешается давать неточную оценку ситуации и делать предположения. Задача заключается в наиболее точной диагностике перечисленных ситуаций за отведенное время.

Общее ограничение по времени по всем этапам: 3 секунды.

Выходные данные

В случае, если машина движется по кругу, то в выходной файл вместо информации о движении машины должна быть выведена строка
infinite loop
В случае бесконечного ожидания ресурса
infinite wait

Пример driver-Ivanov.txt:
if
city is Kazan
then
refuel 321
goto Samara
endif
if
city is Samara
then
refuel 321
goto Kazan
endif

Пример ответа:
infinite loop

Дополнительные условия

В файл readme.txt должно быть включено описание реализованного алгоритма.