Задачи 7-8

Комплект задач содержит три задачи. Тестирование решения участника предполагается на прилагающемся наборе тестов. Распределение баллов за пройденный тест смотри в следующей таблице.

Задача	Количество тестов	Баллы за 1 тест	Баллы за задачу
Соседние числа	10	10	100
Одномерная жизнь	50	2	100
Постфиксный калькулятор	50	2	100

Задача 1. Соседние числа

Эта простая задача, в которой для элемента массива $a[i]$ ответом являются $a[i\text{-}1]$ и $a[i+1]$. Кроме того, нужно учесть два пограничных случая: i=1 (выводим «L») и i=6 (выводим «R»).

Задача 2. Одномерная жизнь

Эта задача предполагает моделирование процесса по заданным правилам.

Заметим, что указанные в условии правила эволюции гарантируют, что колония клеток разрастается на одну клетку слева и справа в каждом поколении независимо от начальной конфигурации. Отсюда, учитывая, что новое значение клетки зависит только от состояния самой клетки и ближайших соседей, получаем, что состояние первых M клеток нового поколения полностью определяется первыми M клетками старого поколения.

Таким образом, эмуляцию процесса эволюции можно организовать в виде простого цикла по номеру поколения, а внутри цикла анализировать массив, описывающий текущее поколение, и формировать массив следующего поколения согласно правилам. При этом оба массива будут иметь фиксированную длину, в соответствии с указанием из предыдущего абзаца.

Замечание: интересные математические свойства данного клеточного автомата можно найти в статье на сайте Википедии: https://ru.wikipedia.org/wiki/Правило_30

Задача 3. Постфиксный калькулятор

Естественным подходом к решению данной задачи является представление промежуточных итогов работы калькулятора в виде двух объектов: текущего списка числовых аргументов (в некоторые моменты процесса вычислений он может быть пустым) и списка команд – ещё не использовавшихся элементов исходного выражения.

Муниципальный этап Всероссийской олимпиады школьников по информатике 2014-2015 учебного года

Нужно обратить внимание, что число также является командой, состоящей в добавлении данного числа в конец списка числовых аргументов.

Основной цикл алгоритма состоит в последовательном проходе по элементам исходного выражения с обновлением списка числовых аргументов. Если очередная команда – число, добавляем его в конец списка. Если очередная команда – бинарная операция, то извлекаем два числа из конца списка и применяем к ним операцию, а результат возвращаем в конец списка. Это принцип работы структуры данных стек. Поскольку гарантируется корректность входных выражений, по завершении процесса вычислений, список будет содержать только одно число – ответ задачи.

Задачи 9-11 класс

Задача 1. Перестановки.

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта
Максимальная оценка:	100 баллов

Буратино изучает тему «Перестановки» под руководством Мальвины. Она пишет на листе шесть чисел и указания как он должен их переставить.

Например, Мальвина пишет числа 56, 9, 20, 34, 15, 82. Указания по перестановке записаны следующим образом: 5, 1, 4, 6, 2, 3. Последняя запись означает, что на 1-ом месте в новой последовательности должно стоять 5-ое число исходной последовательности, на 2-ом в новой – 1-ое из исходной и так далее. Таким образом, правильный ответ, который должен записать Буратино должен быть 15, 56, 34, 82, 9, 20.

Требуется написать программу, которая поможет Буратино решить задачу Мальвины и переставить числа правильным образом.

Формат входных данных

Входные данные состоят из двух строк.

Первая строка содержит шесть натуральных чисел не превосходящих 100. Числа разделены одиночными пробелами. Эти числа представляют собой последовательность записанную Мальвиной.

Вторая строка содержит шесть натуральных чисел не превосходящих 6. Числа разделены одиночными пробелами и каждое встречается в этой строке только один раз. Эта строка описывает указания для перестановки чисел.

Формат выходных данных

Выходной файл должен содержать только одну строку, в которой шесть исходных чисел переставлены в требуемом порядке.

Пример входных и выходных данных

input.txt	output.txt
56 9 20 34 15 82 5 1 4 6 2 3	15 56 34 82 9 20
4 3 5 1 2 6 4 5 2 1 3 6	1 2 3 4 5 6

Задача 2. Машинки.

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта
Максимальная оценка:	100 баллов

Чебурашка любит играть в машинки. У него есть N треков для его машинок. На каждом треке он может запустить ровно одну машинку. Каждый трек состоит из нескольких этапов, между которыми стоит стыковочная платформа. Таким образом, машинка, проехав первый этап трека, попадает на стыковочную платформу, тратит некоторое время на переход на следующий этап трека, а затем мчится по следующему этапу.

Чебурашка может настраивать свои машинки, а именно, задавать постоянную скорость движения по треку. И поэтому у него возникла идея: настроить скорости машинок так, чтобы все они приехали на финиш за 100 секунд, стартовав одновременно. Это будет красиво!

Требуется написать программу, которая поможет Чебурашке рассчитать скорости машинок так, чтобы все машинки были в пути (с учетом переходов между треками) одинаковое время.

Формат входных данных

Первая строка входного файла содержит число N ($2 \le N \le 100$) – количество треков и, соответственно, машинок.

Каждая следующая строка содержит описание i-го трека. Описание трека состоит из T_i ($0 \le T_i \le 10$) – время задержки на любой из платформ трека в секундах, числа K_i ($1 \le K_i \le 10$) – количество этапов трека. Дальше перечисляются длины всех этапов i-го трека в сантиметрах. Длины являются целыми числами в диапазоне от 1 до 100. Все числа в строке описания трека разделены одиночными пробелами.

Формат выходных данных

Выходной файл должен содержать одну строку, содержащую N вещественных чисел. Число с номером i представляет скорость, которую должен задать Чебурашка i-ой машинке на i-ом треке. Все вещественные числа будут сравниваться с эталонами с точностью до 0.01.

Пример входных и выходных данных

input.txt	output.txt
2 1 1 10 10 2 4 5	0.1 0.1
2 10 3 25 25 50 5 3 10 14 6	1.25 0.33

Задача 3. Одномерная жизнь

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта
Максимальная оценка:	100 баллов

Рассмотрим эволюцию колонии закрашенных клеток бесконечной в обе стороны ленты. Судьба каждой клетки в следующем поколении зависит от наличия «организма» в данной клетке и соседних с ней (слева и справа) клеток в текущем поколении.

Правила выживания указаны в следующей таблице:

Текущее состояние трёх соседних клеток	111	110	101	100	011	010	001	000
Новое состояние центральной клетки	0	0	0	1	1	1	1	0

Единицей отмечены «организмы», а нулём – пустая ячейка. Таким образом, правила следует читать так: если, например, в текущем поколении в некоторой клетке стоит 1, слева от неё тоже стоит 1, а справа – 0, то в следующем поколении, в соответствии со вторым столбцом таблицы, в этой центральной клетке будет стоять 0. Далее приведён пример эволюции по этим правилам небольшой колонии.

Поколение 0	0	0	0	0	0	1	1	0	1	0	0	0	0	0
Поколение 1	0	0	0	0	1	1	0	0	1	1	0	0	0	0
Поколение 2	0	0	0	1	1	0	1	1	1	0	1	0	0	0
Поколение 3	0	0	1	1	0	0	1	0	0	0	1	1	0	0

Требуется написать программу, которая по заданной начальной конфигурации и номеру поколения выведет состояние M первых клеток в данном поколении (начиная с самого левого «организма»).

Формат входных данных

Первая строка входного файла содержит состояние нулевого поколения: последовательность цифр из нулей и единиц длиной от 2-х до 100 символов, начинающуюся и заканчивающуюся единицей.

Вторая строка содержит число N ($1 \le N \le 500$) – номер итогового поколения.

Третья строка содержит число M ($1 \le M \le 250$) – количество клеток итогового поколения, которые необходимо вывести.

Формат выходных данных

Одна строка: последовательность из M знаков из нулей и единиц, соответствующая состояниям первых M клеток итогового поколения (начиная с первой «живой» клетки).

Пример входных и выходных данных

input.txt	output.txt
1101 3 4	1100
111 5 3	110

Примечание: первый пример проиллюстрирован в тексте задачи.

Задача 4. Длинный код

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта
Максимальная оценка:	100 баллов

Клиент получил от банка некоторый длинный секретный номер, состоящий из цифр. В процессе телефонного разговора с клиентом, чтобы косвенно удостовериться, что клиент владеет секретным номером, банковский служащий просит его назвать 3 цифры этого номера, стоящие в указанных позициях. При этом номера позиций идут в порядке возрастания, и клиент в ответ называет цифры в указанном порядке. Например, при

Муниципальный этап Всероссийской олимпиады школьников по информатике 2014-2015 учебного года

секретном номере 0257385332 сотрудник может попросить назвать 3-ю, 4-ю и 8-ю цифры, а клиент должен ответить «573». Далее будем называть ответы клиентов «мини-кодами».

В рамках исследования безопасности данного способа подтверждения, требуется написать программу, которая по известному набору правильных мини-кодов найдёт пример самого короткого секретного номера.

Формат входных данных

Первая строка файла содержит целое число N ($2 \le N \le 12$) – количество мини-кодов. Следующие N строк содержат запись мини-кодов, каждый из которых представляет собой последовательность из трёх цифр. Известно, что каждый мини-код представляет собой некоторую подпоследовательность цифр секретного номера.

Формат выходных данных

Выходной файл должен содержать одну строку – последовательность цифр варианта секретного номера минимальной длины.

Пример входных и выходных данных

input.txt	output.txt
2 307 971	93071
3 870 131 374	8131704

9-11 класс

Комплект задач содержит четыре задачи. Тестирование решения участника предполагается на прилагающемся наборе тестов. Распределение баллов за пройденный тест смотри в следующей таблице.

Задача	Количество тестов	Баллы за 1 тест	Баллы за задачу
Перестановки	10	10	100
Машинки	20	5	100
Одномерная жизнь	50	2	100
Длинный код	100	1	100

Задача 1. «Перестановки»

В задаче требуется произвести перестановку чисел в последовательности. Если $a[]$ – это входной массив из шести чисел, а $r[]$ – это указанная перестановка, то ответом задачи будет массив чисел $b[i] = a[r[i]]$.

Задача 2. «Машинки»

Из школьного курса математики, а потом и физики, участники знают формулу вычисления скорости по расстоянию и времени: $v = s/t$. Остается определить s и t. Выберем первый трек и машинку на нем. Расстояние s, которое должна проехать машинка, равно сумме длин этапов трека. Время t, которое может потратить машинка, равно 100 сек (общее время) отнять суммарное время задержек на переходы с этапа на этап. Аналогичные вычисления надо произвести для каждой машинки (терка).

Задача 3. Одномерная жизнь

См. аналогичную задачу для возрастной категории 7-8 класс.

Задача 4. Длинный код

Алгоритм 1. Основная идея предлагаемого решения состоит в последовательном переборе вариантов очередной цифры секретного номера. Алгоритм представляет собой перебор с возвратом и использование отсечений. На каждом шаге перебора имеются следующие элементы: выбранное начало секретного номера и набор условий (мини-кодов) на оставшуюся часть номера. Условия представляют собой просто последовательность цифр, которая должна встретиться в виде подпоследовательности в этой оставшейся части секретного номера.

Оптимизация перебора достигается с помощью следующих наблюдений:

Муниципальный этап Всероссийской олимпиады школьников по информатике 2014-2015 учебного года

1. Очередная цифра секретного номера минимальной длины должна быть одной из множества первых цифр имеющихся мини-кодов (легко доказывается от противного);
2. Если некоторая цифра встречается только в начале мини-кодов, то её можно сразу добавить в префикс, а набор мини-кодов преобразовать, отбросив первую цифру у тех, которые с неё начинаются;
3. Базовое условие отсечения состоит в том, что если длина текущего префикса плюс количество различных букв, встречающихся в текущем наборе мини-кодов больше или равно длине уже найденного примера секретного номера, то данную ветку перебора можно прерывать;
4. Условие отсечения вариантов можно усилить путём учёта цифр, которые встречаются в некотором мини-коде дважды (значит и в оставшейся части номера они должны встретиться не менее двух раз): вместо количества различных букв можно использовать количество различных букв плюс количество "удвоенных" цифр;
5. Поскольку оставшаяся часть секретного номера полностью определяется набором мини-кодов и не зависит от префикса, можно хранить множество просмотренных вариантов, чтобы не повторять их перебор. Элементами этого множества может выступать пара (<длина префикса>, <отсортированный текущий набор мини-кодов>).

Решение, основанное на таком переборе, проходит все тесты и должно оцениваться из 100 баллов.

Алгоритм 2. В качестве простого частичного решения можно организовать перебор по всем вариантам номера, начиная с самых коротких, пока не будут выполнены все условия из набора. Решение, основанное на таком переборе, пройдет не все тесты из-за ограничения по времени.

Алгоритм 3. Хорошим эвристическим алгоритмом, способным удовлетворить части тестов, может выступать жадный алгоритм, состоящий в последовательном наращивании длины секретного номера путём вставки цифр так, чтобы он удовлетворял очередному мини-коду. При этом большое значение имеет выбор этого очередного мини-кода. Очевидным разумным критерием выбора является требование максимального пересечения текущего варианта секретного номера и мини-кода. Решение, основанное на таком переборе, пройдет не все тесты из-за ограничения по времени.

Памятка участнику муниципального этапа Всероссийской олимпиады школьников по информатике 2014/2015 г.

Олимпиада проводится в течение одного дня. На решение задач отводиться 3,5 часа. Во время тура все участники могут отправлять свои решения на автоматическую (!) проверку. Автоматическая проверка осуществляется путем подачи тестовых данных на вход вашей программы и сравнения результата с эталоном. Если тест пройдет, то участник получает некоторое количество баллов, в противном случае нет. Например, если задача оценена жюри в 100 баллов и для этой задачи предусмотрено 20 тестов, то участник, решение которого прошло 7 тестов, а 3 не прошло, получит 35 баллов. Если решение участника не прошло тесты, указанные в качестве примеров в условии задачи, то решение оценивается в 0 баллов.

Уважаемый участник олимпиады, внимательно прочтите следующие инструкции, которыми необходимо руководствоваться при решении и оформлении задач. В случае не выполнения одного из следующих требований Вы можете потерять баллы, как при автоматической проверке Ваших решений, так и в процессе собеседования.

1. Формат входных и выходных данных описан в формулировке каждой задачи. Читать входные и писать выходные данные можно как из (в) файлов, так и в стандартные потоки ввода/вывод При несоблюдении формата входных/выходных данных АСП будет считать соответствующие тесты не пройденными.
2. Ввод данных с клавиатуры и случайная генераци входных данных строго запрещены. Также запрещены программы, в которых от пользователя ожидаются какие-то действия (нажать кнопку, клавишу и т.п.). Последнее требование в особенности относиться к участникам использующим Visual C/C++, Visual Basic, Visual C#.
3. Размер файла с исходным текстом программы не должен превышать 256 килобайт. Время компиляции программы не должно превышать 1 минуты.
4. При компиляции решений участников допустим только следующие ключи компиляции:

Компилятор	Командная строка
Free Pascal 2.6.2	fpc <исходный файл>
Visual C 2010 (2013)	cl /02 /TC <исходный файл>
GNU C 4.4.0 (MinGW)	gcc -02 -x c –W1, --[illegible]
Visual C++ 2010 (2013)	cl /02 /EHs /TP <исходный [illegible]
GNU C++ 4.4.0 (MinGW)	g++ -02 -x c++ –W1, --[illegible]
Visual C# 2010 (2013)	mcs <исходный файл>
Visual Basic .NET 2010 (2013)	vbc <исходный файл>
PascalABC.NET 4.0	pascalabc <исходный файл>

5. АСП рассматривает только исходные тексты решений, размещенные в одном файле со стандартным расширением (.c, .cpp, .pas, vb, .cs). Названия входных/выходных файлов также определяются в формулировке задачи.
6. Участники могут задавать вопросы членам жюри. Прежде чем задать вопрос члену жюри участник должен попытаться найти ответ в тексте задачи. При этом члены жюри имеют право отказаться от комментариев по заданному вопросу. Например, на вопрос: «Правильно ли решена моя задача?» члены жюри будут отвечать: «Без комментариев».
7. Если на сервер жюри поступило несколько решений одной и той же задачи от одного и того же участника, то оцениваться будет только последнее.
8. Автоматическая система проверки находится на сайте ejudge.mmcs.sfedu.ru Перейдя на этот сайт необходимо нажать на гиперссылку с названием олимпиады «Городская олимпиада по информатике 2013» или «City 2013 contest». Дальше необходимо ввести логин и пароль (получите перед началом тура) и нажать кнопку «Log in». Далее необходимо нажать кнопку «Edit» и указать свои фамилию, имя и номер класса. Например, «Сергеев Иван 9». Если Вы видите пункты меню на английском языке, то рекомендуем изменить язык представления страниц олимпиады на русский, нажав кнопку «Settings», а затем выбрав язык «Change language». Далее необходимо нажать кнопку «Участвовать» («Participate»). После этих действий участник окажется на странице олимпиады, где ему необходимо выбирать задачу и отправлять решение (внизу страницы с задачей).