

## **Цикл «Статьи для начинающих»**

### **Вместо предисловия.**

Под начинающим авторы понимают учащегося, который умеет написать хотя бы простую работающую программу на каком-либо языке программирования и умеет обращаться со средой программирования, поддерживающей этот язык. Мы не ставим своей задачей научить программированию с нуля. Цель этого цикла статей — рассказать о соревнованиях по программированию.

### **Как работает проверяющая система и какие не-алгоритмические ошибки можно сделать в решении (статья 1)**

Соревнования по программированию очень популярны в мире, и существует много проверяющих систем. Конечно, в чем-то они похожи друг на друга, но имеют и отличия. Поэтому перед использованием новой для Вас системы стоит ознакомиться с ее особенностями и с примерами решений, принимаемых этой системой. Как правило, такие примеры можно найти на сайте, где размещен интерфейс для отправки решений.

В настоящей статье будет рассказано о том, как работает проверяющая система на <http://contest.uni-smr.ac.ru>.

### **Как начисляются баллы за задачи.**

В системе содержатся задачи как школьных, так и студенческих соревнований. Как правило, индивидуальные соревнования проводятся по «школьным правилам» — с начислением баллов за каждый пройденный тест. Командные соревнования проводятся по «студенческим правилам» — с начислением одного балла за задачу только в случае прохождения всех тестов.

### **Зачем нужно регистрироваться**

Чтобы сдавать задачи, необходимо зарегистрироваться. Условия задач доступны и без регистрации, но возможность сдать решение появляется только тогда, когда Вы войдете в систему под своими логином и паролем. Регистр символов имеет значение, поэтому пользователь Test и пользователь test с точки зрения системы — разные пользователи. Рекомендуем использовать в логинах латинские буквы, цифры, символы подчеркивания — тогда вне зависимости от используемого браузера Вы наверняка сможете успешно просмотреть свой профиль.

В профиле есть возможность просматривать отправленные на проверку решения. Когда Вы перейдете по ссылке «Мои решения», то увидите список отправленных Вами решений на проверку. Каждый элемент списка содержит числовой идентификатор решения (по этой ссылке Вы можете увидеть текст отправленного Вами решения), название задачи, выбранный при сдаче решения компилятор, вердикт проверяющей системы и количество набранных баллов. Конечно, если Вы не сдали еще ни одной задачи, то список будет пустым.

### **Как выглядит решение задачи**

Решением задачи с точки зрения проверяющей системы является текстовый файл с исходным текстом программы, записанным на одном из допустимых языков программирования. Вы можете скопировать этот исходный текст в поле для ввода решения из среды программирования, в которой он был разработан, или же выбрать файл с решением с помощью кнопки «Обзор». Делать одновременно и то, и другое не нужно.

Решение должно в точности соблюдать формат входных и выходных данных, описанный в условии задачи. В частности, это означает, что если в задаче указано, что данные прочитываются из входного файла с именем input.txt, а выводятся в файл output.txt (а именно это, как правило, и указано), то не нужно разрабатывать многооконное приложение

со множеством форм ввода, кнопок и меню: подумайте о том, что проверяться Ваше решение будет автоматизированной системой, которая не умеет читать надписи на кнопках, вводить данные в формы ввода и вряд ли придет в восторг от того, что полученный ответ Вы вывели по центру экрана красным цветом шрифтом Arial кеглем 24. Существуют конкурсы, на которых необходимо разрабатывать приложения, ориентированные на пользователя. Но в олимпиадном программировании вся красота заключена в алгоритме решения, а интерфейс программы весьма аскетичен.

### **Проверка решения и сообщения системы**

Перед тем как отправить решение задачи на проверку, убедитесь, что выбран нужный компилятор. Если Вы используете язык Pascal, настоятельно рекомендуем при разработке пользоваться средами Free Pascal или Delphi, поскольку далеко не все задачи могут быть решены средствами Borland Pascal.

Если все (с Вашей точки зрения) хорошо, можете нажимать кнопку «Отправить на проверку». Проверяющая система спустя некоторое (небольшое) время сообщит Вам вердикт. Вердиктом может быть:

Accepted — решение принято, прошло все тесты.

Wrong answer on test NN (где NN — некоторый номер теста) — решение не прошло некоторые тесты, и NN — первый по порядку номер среди номеров этих тестов. Не нужно удивляться, если после доработки решения Вы получите большее количество баллов, а номер первого неверного теста сохранится. Это лишь означает, что Вы исправили другие ошибки, что позволило Вашему решению пройти некоторые другие тесты с большими порядковыми номерами.

Time limit exceeded on test NN (где NN — некоторый номер теста) — решение превысило допустимое время выполнения на некоторых тестах, и NN — первый по порядку номер среди номеров этих тестов. Замечание, сделанное в предыдущем абзаце сохраняет свою силу.

Runtime error on test NN (где NN — некоторый номер теста) — решение написано синтаксически верно, но при прохождении некоторых тестов Ваша программа пытается выполнить какие-то недопустимые действия.

Compilation error — ошибка компиляции. Ваше решение содержит синтаксические ошибки.

Иногда (по разным причинам) проверяющая система может быть отключена. В этом случае Вы будете видеть сообщение «Waiting...» (такое же сообщение появляется при сильной загрузке системы, но это случается в дни соревнований, о которых известно заранее). Беспокоиться не стоит — Ваше решение обязательно будет проверено системой, как только она будет запущена.

### **Почему Вы могли не получить Accepted**

Если вердикт, полученный Вашим решением, отличается от Accepted, то Вам придется искать ошибки. Вполне возможно, что алгоритм, придуманный Вами, не решает предложенную задачу. Однако не стоит забывать, что проверка решений ведется автоматизированно, и они должны соответствовать определенному шаблону, так что проблема может состоять именно в несоответствии. Поэтому обсудим некоторые возможные причины возникновения «ошибок», не связанных с неверным алгоритмом.

Итак, на какие вопросы стоит постараться ответить, если Вы получили вердикт...

#### ***Compilation error.***

Действительно ли Вы выбрали соответствующий компилятор при сдаче решения (на странице Мои решения)?

Не случилось ли так, что при копировании исходного текста программы в поле ввода Вы скопировали только лишь его часть?

Внимательно ли Вы изучили пример оформления задачи на выбранном Вами языке программирования? Некоторые из этих примеров доступны в разделе «Для подготовки»

(например, для решения на Java в нашей системе обязательно указывать пакет).

### ***Runtime error on test NN***

Проверяющая система компилирует исходный текст Вашего решения, после чего запускает полученный исполняемый файл последовательно на всех имеющихся в системе тестах. Ваша программа должна прочитывать файл input.txt из текущего (по отношению к размещению созданного проверяющей системой исполняемого файла) каталога и записывать ответ в файл output.txt, который также формируется в текущем каталоге. Возможно, что при написании решения Вы обращались к файлам на Вашем жестком диске, используя полные пути (например C:\MyProgram\input.txt). В решении, которое Вы сдаете на проверку, должны использоваться имена файлов input.txt и output.txt без указания каких-либо каталогов. Проверьте, так ли это? (Заметим, что эта ошибка проявляется при NN = 1).

Не пытается ли Ваша программа создать какой-либо «вспомогательный» файл? Например, для отладки Вы выводили некие промежуточные результаты вычислений и забыли отключить эту возможность в окончательной версии программы.

Обратите внимание на ограничение по памяти в условии задачи (если оно отсутствует, можете считать, что в распоряжении Вашей программы будет 64 Мб). Не пытается ли Ваша программа, например, создать массив из пары сотен миллиардов вещественных чисел?

Внимательно прочитайте формат входных данных и посмотрите на пример в условии задачи. Вполне возможно, например, что там указано «N строк, в каждой из которых содержится M целых чисел через пробел», а Ваша программа пытается прочитать M строк по N целых чисел. Или же «в строке содержится K символов 0 и 1 подряд», а Ваше решение написано так, как если бы между этими символами были пробелы. Или, быть может, Вы перепутали порядок ввода данных и (опять же к примеру) пытаетесь прочитать из входного файла целое число, в то время как там на этом месте записано вещественное?

Проверьте также, сдаете ли Вы именно ту задачу, решение которой отправили? У другой задачи другой формат входных данных, и, хотя Ваше решение осуществляет чтение правильно, оно осуществляет его не из надлежащего источника.

Чаще всего первый тест в задаче совпадает с тестом из условия. В задачах официальных соревнований последних лет это правило практически всегда соблюдается. В остальных случаях этот вопрос остается на усмотрение автора. В качестве простого теста Вы можете попробовать написать программу, прочитывающую данные из входного файла и, если они совпадают с данными первого теста, выводящую в качестве ответа результат первого теста. Конечно, баллы за это Вы не получите (хотя авторы иногда проявляют неожиданную щедрость и выставляют ненулевые баллы за пробный тест), но зато будете уверены, что чтение из входного файла происходит так, как надо (конечно, авторы бывают разные, и некоторые предпочитают приводить в качестве теста в условии совсем простые и малоинформативные тесты).

### ***Time limit exceeded on test NN***

Когда сформированный системой исполняемый файл последовательно запускается на каждом тесте, то на обработку входных данных теста ему отводится определенное время. В системе можно назначить разное время на выполнение разных тестов. Однако чаще всего (в официальных соревнованиях как правило) на каждый тест отводится одинаковое время. В условии указывается максимальное время на тест. Если оно не указано, это значит, что на прохождение каждого теста отводится 1 секунда. Решение, которое не уложилось в отведенное время, считается не прошедшим тест и не получает баллы на данном тесте. Такой подход направлен на то, чтобы исключить из рассмотрения неэффективные по времени решения. Но причина может быть не только в этом.

Проверьте, не ожидает ли Ваша программа каких-либо действий от пользователя? Быть может, Ваша программа печатает на экране что-то вроде “Vvedite N ” или же предлагает указать путь к файлу со входными данными? (такая ошибка чаще всего проявляется уже при

NN = 1).

Не выводит ли Ваша программа на экран что-либо, например, большие объемы промежуточных или выходных данных? Вполне возможно, Вы использовали такой вывод в целях отладки и забыли его закомментировать в отправленной на проверку версии программы. Вывод на экран — достаточно медленная операция (заметим, что такой вывод, вообще говоря, может быть расценен как вмешательство в работу проверяющей системы) и даже при эффективном алгоритме может привести к превышению ограничения по времени..

Хотя мы обещали оставить в стороне «алгоритмические проблемы», стоит проверить, нет ли у в Вашем решении бесконечно работающих циклов.

### ***Wrong answer on test NN***

Выходной файл, сформированный Вашей программой, должен в точности соответствовать формату, описанному в условии задачи. В большинстве случаев производится побайтовое сравнение правильного ответа с тем, что выдала Ваша программа (реже используются «чекеры», сравнивающие ответ с правильным не побайтово, а «по содержанию»). Это означает, что никаких лишних знаков (в том числе пробелов) в файле output.txt быть не должно. Например, если в строке выводится несколько чисел через пробел, то перед первым и после последнего числа не должно быть пробелов (перевод строки после последнего числа допустим). Остальные же числа должны разделяться ровно одним пробелом. Красивое форматирование, которое может порадовать глаз, будет оценено проверяющей системой как неверный результат.

Если в формате выходного файла указано, что в первой строчке содержится слово YES или NO, а во второй — два целых числа через пробел, не стоит писать в первой строчке “Zadacha imeet dva resheniya”, во второй “Kolichestvo = 5”, а в третьей “Summa = 15”.

Если в качестве ответа требуется вывести какое-либо слово, нужно выводить именно его, без кавычек, в том регистре символов, в каком оно указано в условии. И даже с теми же ошибками (если они, на Ваш взгляд, есть).

Если в качестве ответа требуется вывести вещественное число с точностью 3 знака после десятичной точки (в условии может быть написано «запятой», Ваши сомнения должен разрешить пример в условии), то цифр после десятичной точки должно быть ровно 3, даже если все они нули. Если дробная часть числа содержит больше цифр, округление производится по обычным математическим правилам (обычно в языках программирования стандартные функции справляются с этой задачей).

Обратите также внимание на порядок выводимых данных. Например, в условии задачи просили указать в ответе, сколько шагов придется совершить герою на запад, и сколько на север, а Ваше решение выводит сначала количество шагов на север, а потом на запад. Или, быть может, необходимо вывести три числа по одному на строку, а Вы выводите их в одну строку через пробел.

Проверить, ту ли задачу Вы сдаете, также будет не лишним. В ряде несложных задач формально входные данные могут совпадать (мало ли задач можно придумать, когда на вход подаются несколько целых чисел), поэтому чтение входных данных происходит без проблем, но на выходе получается совсем не то, что ожидает проверяющая система.

### **Какие задачи можно порешать «для начала»?**

Чтобы научиться работать с проверяющей системой, не стоит выбирать сразу сложные задачи. Начните с самых простых, чтобы можно было сосредоточиться на оформлении задачи, а не на алгоритме решения (в простых он очевиден).

Самые «проверенные» — это задачи с официальных соревнований. Перейдите по ссылке Соревнования на соответствующую страницу и выбирайте то, которое понравится. Конечно, понятно, что «самая простая» задача с командного студенческого чемпионата может оказаться заметно сложнее, чем многие задачи окружного (в разные годы — городского, муниципального) тура олимпиады школьников по информатике. Оценить сложность задачи,

можно открыв ссылку «Монитор» на странице выбранного соревнования. Количество участников, решивших задачу, обычно весьма красноречиво говорит о том, достаточно ли она простая. Конечно, бывают случаи, когда действительно простую задачу почему-либо не решают, но обратные ситуации намного реже.

### **Ресурсы**

В заключение этой статьи приведем список простых задач из официальных соревнований, которые можно попробовать решить начинающему. Разумеется, список не претендует на полноту.

Задача С. Не двенадцать стульев. XI командный чемпионат по программированию среди высших учебных заведений Самарской области (<http://contest.uni-smr.ac.ru/ru/problemset/5208/>)

Задача I. USB-окрестность. XI командный чемпионат по информатике и математике среди школьников Самарской области. (<http://contest.uni-smr.ac.ru/ru/problemset/5180/>)

Задача Н. Секундное дело. XI командный чемпионат по информатике и математике среди школьников Самарской области. (<http://contest.uni-smr.ac.ru/ru/problemset/5179/>)

Задача А. "Едет с грамотой гонец..." Окружной этап предметной олимпиады по информатике 2009 / 2010. (<http://contest.uni-smr.ac.ru/ru/problemset/891/>)

Задача С. Битва. Окружной этап предметной олимпиады по информатике 2009 / 2010. (<http://contest.uni-smr.ac.ru/ru/problemset/893/>)

Задача D. Городские стены. Окружной этап предметной олимпиады по информатике 2009 / 2010. (<http://contest.uni-smr.ac.ru/ru/problemset/894/>)

Задача К. Свиная отбивная. X Командный чемпионат по программированию среди высших учебных заведений Самарской области (<http://contest.uni-smr.ac.ru/ru/problemset/794/>)

Этапы большого пути. X командный чемпионат по информатике и математике среди школьников Самарской области (<http://contest.uni-smr.ac.ru/ru/problemset/728/>)

Свежая струя. X командный чемпионат по информатике и математике среди школьников Самарской области (<http://contest.uni-smr.ac.ru/ru/problemset/729/>)

Счастливые билетки. Окружной этап предметной олимпиады по информатике 2008 / 2009. (<http://contest.uni-smr.ac.ru/ru/problemset/663/>)

Папки. Окружной этап предметной олимпиады по информатике 2008 / 2009. (<http://contest.uni-smr.ac.ru/ru/problemset/666/>)

Перерыв на обед. Окружной этап предметной олимпиады по информатике 2008 / 2009. (<http://contest.uni-smr.ac.ru/ru/problemset/672/>)

Вечеринка. Окружной этап предметной олимпиады по информатике 2008 / 2009. (<http://contest.uni-smr.ac.ru/ru/problemset/674/>)

SP. Окружной этап предметной олимпиады по информатике 2008 / 2009. (<http://contest.uni-smr.ac.ru/ru/problemset/676/>)

Задача I. Светофоры. IX командный чемпионат среди вузов Самарской области. (<http://contest.uni-smr.ac.ru/ru/problemset/638/>)

Также подборка несложных задач из более ранних соревнований содержится в соревновании Simple contest (<http://contest.uni-smr.ac.ru/ru/contests/97/>)

Кроме этого, можно попробовать решить задачи из следующего списка. От предыдущих они отличаются лишь тем, что требуют весьма внимательного прочтения условия.

Задача А. Отбросить нельзя округлить. I тур окружного этапа предметной олимпиады по информатике среди школьников в 2010 / 2011 гг. (<http://contest.uni-smr.ac.ru/ru/problemset/5294/>)

Задача В. Информационное письмо. XI командный чемпионат по программированию среди высших учебных заведений Самарской области (<http://contest.uni-smr.ac.ru/ru/problemset/5294/>)

[smr.ac.ru/ru/problemset/5207/](http://smr.ac.ru/ru/problemset/5207/) )

И швец, и жнец... X командный чемпионат по информатике и математике среди школьников Самарской области (<http://contest.uni-smr.ac.ru/ru/problemset/721/> )

Задача F. Стремянка. IX командный чемпионат среди вузов Самарской области. (<http://contest.uni-smr.ac.ru/ru/problemset/635/> )