Компьютерные инструменты в образовании, 2025
№ 2: 81–95
УДК: 378.147:004.43
http://cte.eltech.ru
doi:10.32603/2071-2340-2025-2-81-95

практика программирования

СОРЕВНОВАНИЯ ПО ПРОГРАММИРОВАНИЮ НА ПЛАТФОРМЕ ЯНДЕКС.КОНТЕСТ: КОМУ НУЖНЫ И КАК СОЗДАВАТЬ

Половикова О. Н.[1], канд. физ.-мат. наук, доцент, ✉ polovikovaol@yandex.ru,
orcid.org/0000-0002-9403-1195
Смолякова Л. Л.[1], старший преподаватель, knaus.larisa@gmail.com,
orcid.org/0000-0001-8547-0324

[1]Алтайский государственный университет, пр. Ленина, д. 61, 656049, Барнаул, Россия

Аннотация

В данном исследовании акцентируется внимание на необходимости получения студентами IT-направлений подготовки опыта решения практических задач. Активное участие студентов в спортивном программировании вне зависимости от начальных навыков способствует получению такого опыта. Представлен подход к созданию состязаний с использованием библиотеки *testlib* на бесплатной российской платформе Яндекс.Контест. Программные модули платформы обеспечивают основные этапы жизненного цикла соревнования от генерации тестов к задачам до подведения итогов. Предоставлена возможность реализовать логику работы своей системы оценивания, учитывая набранные студентами баллы и штрафы. Используя настройки заданий можно ограничивать ресурсы памяти и время выполнения программного кода участников. Специальный модуль также формирует отдельный набор требований к способу решения задач, что является важным условием для проведения соревнований в рамках учебного процесса. Создавая банк заданий для соревнования с учётом пройденного материала и изучаемых курсов, можно «вшить» проведение соревнований в учебный процесс, чередуя индивидуальные и командные этапы. Предлагаемый подход апробирован на практике: два раза в семестр студенты младших курсов одного из институтов Алтайского госуниверситета участвуют в состязаниях, тренируя свои навыки быстрого и безошибочного написания программного кода. Опыт проведения подобных турниров показал готовность студентов и преподавателей осваивать новые формы учебных занятий.

Ключевые слова: *соревнование по программированию, автоматизированная проверка кода, чекер, интерактор, постпроцессор, генерация тестов, штрафные очки, баллы за решение, монитор участников, банк заданий, тесты, программный код.*

Цитирование: Половикова О. Н., Смолякова Л. Л. Соревнования по программированию на платформе Яндекс.Контест: кому нужны и как создавать // Компьютерные инструменты в образовании. 2025. № 2. С. 81-95. doi: 10.32603/2071-2340-2025-2-81-95

1. ВВЕДЕНИЕ

Современные тенденции кадрового рынка требуют от выпускников вузов опыта решения практических задач в их профессиональной деятельности. Эти требования предъ-

являют работодатели, а формируются они средой, в которую попадают выпускники после трудоустройства. «Главная и наиболее часто звучащая претензия работодателей к выпускникам вузов сегодня — оторванность полученных знаний от практики» [1]. Федеральные государственные образовательные стандарты высшего образования по направлению «Информатика и вычислительная техника» определяют компетенции по программам бакалавриата, которые можно обеспечить в том случае, если студенты получили достаточный опыт самостоятельной и коллективной разработки программ [2]. Поэтому для вузов с IT-направлениями подготовки в ближайшей перспективе должна быть решена задача по созданию специальных условий для развития навыков программной разработки [3, 4].

Именно от опыта исполнителя зависит результативность всех этапов жизненного цикла прикладного проекта: от построения концепции до внедрения. Прикладные навыки формируются не только за счёт выполненного объёма подобных работ, но и спецификой и разнородностью решаемых задач. Нетривиальные (глубокие по сложности) решения, основанные на эффективных алгоритмах, невозможно построить, если использовать только заранее известные шаблоны. Для программной реализации таких задач нужны универсальные инструменты: системное мышление и навыки программной разработки с учётом некоторых ограничений (время на разработку, используемые программой ресурсы, методология и др.).

Одним из очевидных способов получения практического опыта разработки является участие в соревнованиях по программированию: вовлечение в сам процесс олимпиадного движения [5, с. 52]. Анализ исследований в данной предметной области [5–8] показал высокую ценность таких состязаний. От участников требуется максимальное погружение в предметную область, концентрация всех внутренних сил для достижения результата. В ходе работы формируется команда и выстраиваются взаимоотношения, оцениваются личностные качества: стрессоустойчивость, коммуникабельность, целеустремленность.

Олимпиадное программирование предполагает решение нетривиальных задач с учётом ограничений. «Основным принципом таких соревнований является решение заданных алгоритмических задач с ограничениями на объем потребляемой памяти и на время выполнения программы. При этом учитывается правильность решения задания, время написания этого решения и количество неудачных попыток сдачи решения» [8, с. 7].

Конечно, существует множество федеральных и международных соревнований подобного рода. Но следует учитывать тот факт, что эти соревнования требуют от участников серьёзной подготовки и слаженной командной работы, так как рассчитаны на «профессионалов», которые уже обладают уникальными навыками. Сложно найти и актуализировать состязания, которые рассчитаны на «новичков», делающих первые шаги как разработчики (программисты). В рамках учебного курса для нескольких групп (учебный поток) востребованы состязания «локального» уровня, которые будут соответствовать программе обучения: полученной базе теоретических знаний, рассмотренным языкам и средам программирования, освоенным инструментам для проектирования решений. Подобные мероприятия можно «вшить» в образовательную траекторию профильного курса, для того чтобы каждый студент мог приобрести ценный опыт участника, оценить свои возможности и получить дополнительные стимулы для своего развития.

В данной работе представлен подход к созданию соревнований по программированию с использованием готового решения - платформы Яндекс.Контест [9]. Соревнование рассматривается как система-конструктор, основные элементы которой формируются

через web-интерфейс, не требуя дополнительного программного обеспечения. На основе подхода построены состязания для студентов первого и второго курсов направления подготовки 09.04.03: Прикладная информатика Института математики и информационных технологий Алтайского госуниверситета. Периодичность их проведения — один раз в семестр за несколько недель до начала сессии. В течение двух недель предлагается решить несколько десятков задач для различных предметных областей. При этом необходимо использовать конкретные парадигмы программирования и определенные структуры данных. Вес каждой задачи в итоговой сумме баллов зависит от её сложности, от числа неудачных попыток решения и от количества положительных вердиктов, полученных ранее другими участниками. Личные вклады участников (набранные баллы) учитываются при выставлении зачётов по предмету.

Авторы исследования не ставят перед собой цели доказать уникальность выбранной платформы перед другими площадками с подобным функционалом. В данной статье сформированы практические рекомендации проведения состязаний по программированию с учётом ресурсов самой платформы и вспомогательных модулей, которые автоматизируют часть процессов по подготовке и проверке заданий. Материалы исследования позволят сделать выводы относительно применимости такого подхода к конструированию состязаний в конкретном вузе. Комментарии по настройке соревнования помогут организаторам избежать ошибок и сократить временные затраты на создание таких турниров также и на других платформах.

2. ПОЧЕМУ ВЫБРАНА ПЛАТФОРМА ЯНДЕКС.КОНТЕСТ?

Основываясь на выводах компании Rubrain.com [10], можно выделить следующие популярные платформы для проведения онлайн-состязаний по направлению «спортивное программирование»: CodeForces, Яндекс.Контест, AllCups, HackerRank. Каждая из этих систем обладает необходимым функционалом для тестирования кода и для поддержки своего сообщества (тренировки, курсы, тренажеры, образовательные раунды и т. д.). Например, AllCups — это площадка от компании Mail.ru Group для проведения крупных соревнований, таких как «Цифровой прорыв». Для сторонних организаций могут быть предоставлены ресурсы платформы, но для этого нужно обсудить условия сотрудничества (в том числе и финансовые) с компанией Mail.ru, согласовать концепцию соревнования, задачи и другие детали. Платформа HackerRank также позволяет создавать свои соревнования сторонним организациям и компаниям по платной подписке, предоставляя возможность использовать базу заданий платформы и/или самостоятельно формировать базу заданий.

CodeForces [11] — российская платформа для создания соревнований (мэшапов). Задачи для мэшапа можно загрузить свои (должны быть предварительно подготовлены в системе Poligon) или выбрать из архива Codeforces с использованием API. CodeForces развивается как тренировочная и учебная платформа, так как прошедшие соревнования можно использовать для самостоятельной подготовки. Эту возможность студенты могут использовать для развития навыков олимпиадного программирования и для налаживания эффективного взаимодействия внутри команды. Но существенным ограничением платформы является требование к пользователям для создания мэшапов: данный интерфейс становится доступен, если зафиксировано участие не менее чем в трех официальных рейтинговых раундах CodeForces [12].

Яндекс.Контест — платформа с полной поддержкой жизненного цикла соревнования (контеста). Правила для состязаний и задания готовят и размещают организаторы. Так

же как в системе Poligon [13], подготовка и настройка заданий основываются на библиотеке *testlib* [14]. В Яндекс.Контест есть открытые для зарегистрированных пользователей соревнования (без ограничения по рейтингу). По оценке Rubrain.com [10] большинство участников — начинающие программисты (джуны) или просто учащиеся.

Опыт организаторов олимпиад по программированию [15, 16] показывает, что платформа Яндекс.Контест обладает полноценным набором возможностей для подготовки, создания и проведения соревнований (контеста) с перспективой многократного использования, что важно для учебных заведений. Функционалом для построения контеста может бесплатно воспользоваться любой зарегистрированный пользователь, при этом возможна коллективная работа по его созданию и настройке. Используемая система прав доступа позволяет управлять возможностями участников, оперируя учебными подгруппами, группами и потоками. Служба поддержки по запросу предоставляет готовые наборы задач разного уровня сложности, что бесспорно ускоряет построение соревнования и расширяет банк заданий.

3. ОСНОВНЫЕ ЭЛЕМЕНТЫ КОНТЕСТА. КАК ОНИ ВЗАИМОДЕЙСТВУЮТ?

Яндекс.Контест позволяет конструировать системы, элементы которых по необходимости можно корректировать и настраивать непосредственно во время проведения соревнований. У преподавателей есть возможности: вносить изменения банк заданий, изменять подсистему проверки, настраивать подсистему оценивания, анализировать и комментировать программный код участников, отслеживать активности.

Как и на большинстве платформ с поддержкой библиотеки *testlib*, работа с кодом в Яндекс.Контест основана на использовании вспомогательных подпрограмм: генератор, валидатор, чекер и постпроцессор. Аналогично платформе CodeForces Яндекс.Контест позволяет экспортировать задачи из системы Polygon. В данный сервис уже встроены средства автоматизации и самопроверки контеста, что позволяет сократить ресурсозатраты на подготовку соревнований. На рисунке 1 представлена схема взаимодействия подпрограмм (модулей), отвечающих за ввод заданий и проверку решений участников.

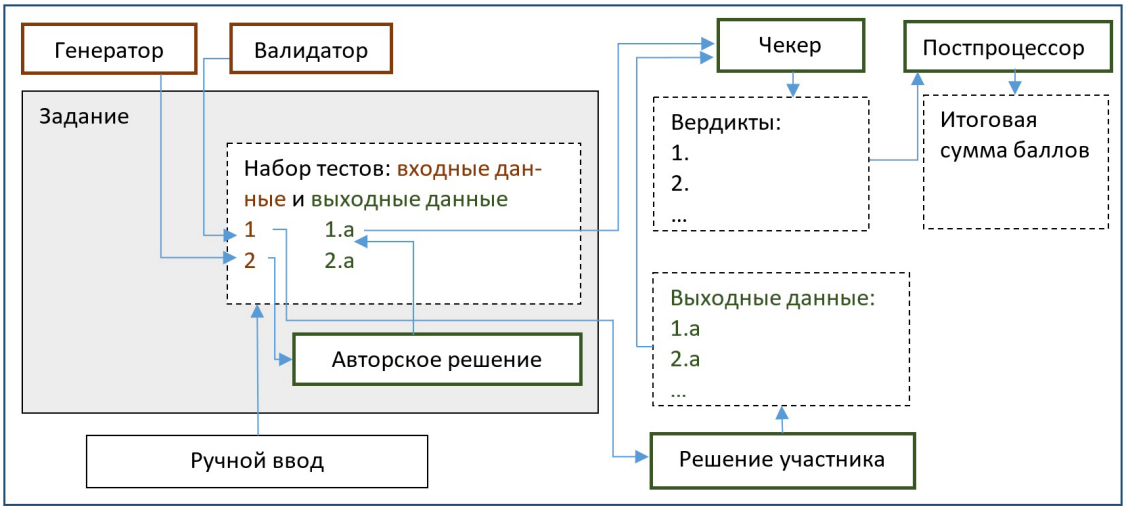


Рис. 1. Схема взаимодействия вспомогательных подпрограмм на платформе

Для автоматической проверки решений для каждого задания следует построить наборы тестов: варианты входных и выходных данных. Тесты можно создать *вручную* или воспользоваться генератором. Основная задача генератора — получить на выходе псевдослучайный набор данных, который и будет использоваться в качестве входных данных тестов. Авторское решение формирует выходные данные, которые система использует во время проверки пользовательских решений в качестве образцов.

Есть задачи, для которых сгенерировать возможные варианты входных данных крайне сложно, либо написание программы генерации является более затратным, чем *ручной ввод*. Для подобных задач нужен модуль — валидатор, который проверяет данные на соответствие некоторому шаблону, то есть отвечает за корректность самих тестов, на основе которых будет оцениваться программа участника. При ручном вводе тестов, валидатор позволяет избежать ошибок или неточностей, связанных, например, со знаками препинания или с форматами вещественных чисел.

Основная проверка решений осуществляется программой — чекером. Для обычных заданий чекер выполняет код участника на наборе тестов, сравнивает вывод программы участника с правильными ответами и сообщает системе результат сравнения в виде кодов возврата. Оценивать результат программы можно не только по символьному совпадению с шаблоном-ответа, но и руководствуясь специальными критериями, например, по принадлежности числового результата заданному диапазону, по включению полученного ответа в список, по соответствию сформированного текстового результата регулярному выражению.

Для специальных заданий, когда предъявляются, в том числе, требования к способу решения (например, использовать классы, рекурсию, функции и т. д.), к проверке подключается интерактор (см. рис. 2).

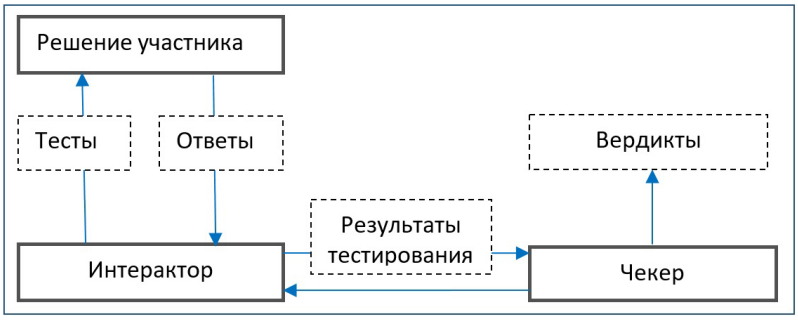


Рис. 2. Проверка решения с использованием интерактора

В этом случае для оценки программного кода выполняются следующие этапы: запускается интерактор, код динамически формирует тестовые данные для решения участника, проверяемая на корректность программа выполняется и полученный ответ анализируется интерактором. Таким образом, обеспечивается взаимодействие вида: *интерактор — проверяемый код — интерактор*. Количество необходимых запросов к коду может зависеть, например, от набора проверяемых функций программы или методов класса. Шаблоны (примеры) кода для создания интерактора можно загрузить с платформы Яндекс.Контест или из открытых репозиториев библиотеки *testlib*.

Для индивидуального оценивания каждой посылки участника нужно к заданию прописать логику работы постпроцессора. Чекер проверяет правильность работы программы на каждом тесте. Постпроцессор определяет итоговый балл за решение

и может управлять процессом оценивания. «Стоимость» решения в баллах может корректироваться, например в зависимости от количества пройденных тестов. Даже если проверяемая программа ошибается на некотором наборе тестов, постпроцессор может оценить это решение ненулевым баллом. «Штрафовать» участников можно за количество посылок с решением для каждой задачи, тем самым требуя от участников качественной разработки и тщательной предварительной проверки своего кода.

Также есть возможность открыть участникам доступ к отчетам чекера на проверяемых решениях. Студенты могут анализировать текстовые выводы (найденные ошибки) автоматической системы проверки решений (см. рис. 3), повторно отправлять исправленные варианты на проверку, отслеживать через специальный интерфейс (по web-монитору) активность других участников, планировать и распределять свои усилия. Следует заметить, что данная платформа обеспечивает достаточный для учебного процесса выбор языков программирования, который отвечает современным тенденциям процесса разработки.

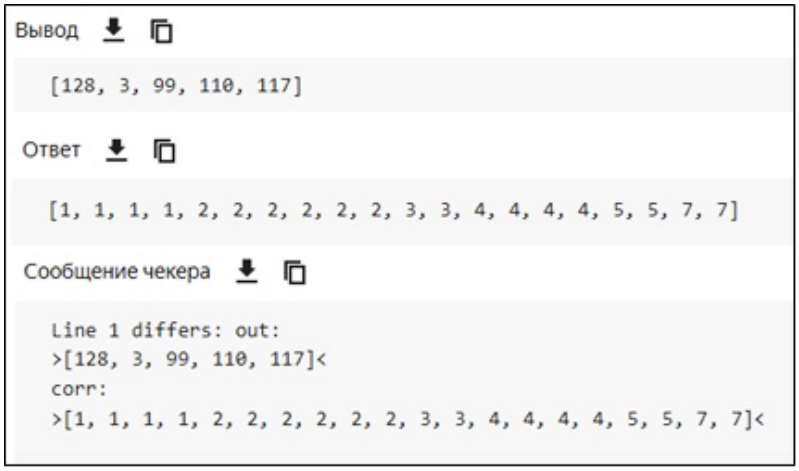
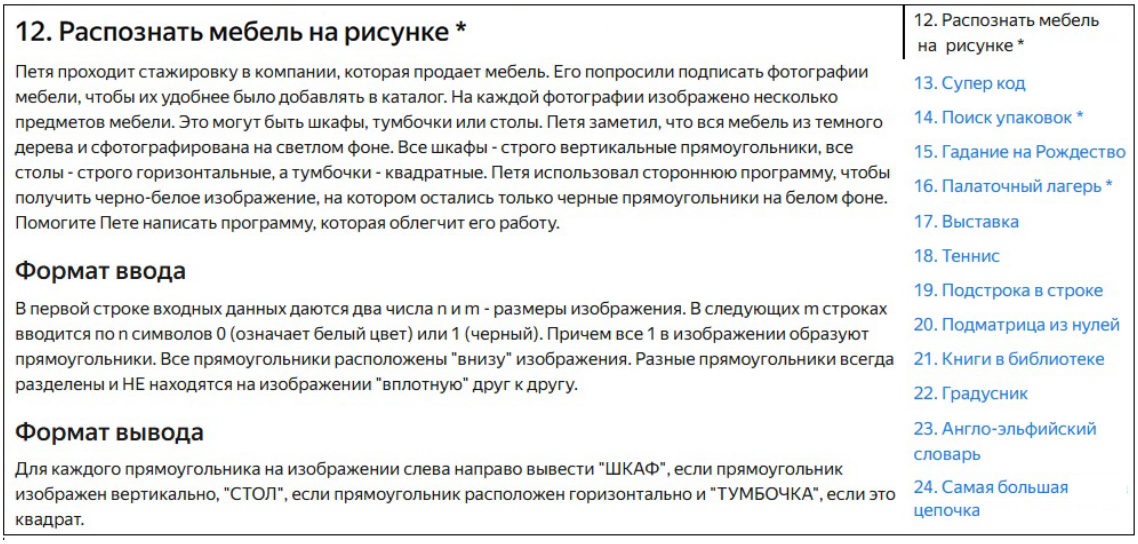


Рис. 3. Сообщение чекера по результатам проверки программного кода на тесте

Гибкость системы для проведения соревнования по программированию формируется не сложными многоступенчатыми настройками всех её параметров, а открытостью процессов работы с платформой. Можно запрограммировать свою логику работы всех подсистем соревнования, в рамках определенного набора правил и принципов.

4. С ЧЕГО НАЧАТЬ СОЗДАНИЕ СОРЕВНОВАНИЯ?

Для организации и проведения соревнований необходимо подготовить *Правила проведения* и *Положение о соревновании*, сформировать банк заданий, создать и настроить соревнование на специальной платформе, задать доступ для пользователей и определить их роли. *Правила* определяют цели, задачи и общие принципы в рамках которых будет строиться соревновательный процесс. *Положение* формируется с учётом *Правил*, определяет порядок и конкретные условия проведения соревнования по программированию: задания, сроки, этапы, участники, платформа и т. д. *Положение* формируется в процессе конструирования самого соревнования, потому что конкретные условия и ход выполнения его этапов зависят от тонкостей *программной настройки* контеста. Таким образом, в процессе работы будут получены конкретные правила проверки и оценивания реше-



12. Распознать мебель на рисунке *

Петя проходит стажировку в компании, которая продает мебель. Его попросили подписать фотографии мебели, чтобы их удобнее было добавлять в каталог. На каждой фотографии изображено несколько предметов мебели. Это могут быть шкафы, тумбочки или столы. Петя заметил, что вся мебель из темного дерева и сфотографирована на светлом фоне. Все шкафы - строго вертикальные прямоугольники, все столы - строго горизонтальные, а тумбочки - квадратные. Петя использовал стороннюю программу, чтобы получить черно-белое изображение, на котором остались только черные прямоугольники на белом фоне. Помогите Пете написать программу, которая облегчит его работу.

Формат ввода

В первой строке входных данных даются два числа n и m - размеры изображения. В следующих m строках вводится по n символов 0 (означает белый цвет) или 1 (черный). Причем все 1 в изображении образуют прямоугольники. Все прямоугольники расположены "внизу" изображения. Разные прямоугольники всегда разделены и НЕ находятся на изображении "вплотную" друг к другу.

Формат вывода

Для каждого прямоугольника на изображении слева направо вывести "ШКАФ", если прямоугольник изображен вертикально, "СТОЛ", если прямоугольник расположен горизонтально и "ТУМБОЧКА", если это квадрат.

Рис. 4. Пример задания (текст)

ний, процедуры формирования рейтинга участников и разрешения спорных ситуаций. В *Положении* следует обозначить и мотивационную составляющую соревнования, так как она во многом определяет его успешность. Это могут быть варианты возможных поощрений для студентов, которые преодолеют заданный в *Положении* порог баллов.

Так как соревнование должно соответствовать образовательному процессу, это в первую очередь накладывает ограничения на предлагаемый банк заданий: сложность заданий и стиль их текстового описания, ограничения на языки программирования и набор дополнительных требований по реализации. Примеры заданий и оценка их сложности представлены в работе Н. О. Котелина, Н. К. Попова [15]. Пример задания из соревнования по программированию для студентов Алтайского госуниверситета представлен на рисунке 4. Всего на данный момент в банке более сотни задач, решение которых предусматривает использование различных структур данных и алгоритмов поиска.

Количество тестов к заданиям определяется условиями конкретной задачи, но, как правило, для разносторонней проверки следует предусмотреть порядка 15–20 тестов. Один или несколько тестов (входные и выходные данные) можно добавить в качестве образца в примечание к задаче. Например, для задачи «Распознать мебель на рисунке» генератор формирует входные данные: два натуральных числа для размера изображения и бинарный массив, который и выступает изображением. Выходные данные (ответ) позволяют получить авторское решение. На рисунке 5 представлен пример такого примечания к задаче.

Для каждой задачи настраиваются индивидуальные требования к решению, включая ограничения на запуск, компиляцию и работу чекера. Требования контролируются автоматически в процессе проверки решения участника с формированием соответствующего вердикта. Закладываемые ограничения позволяют учитывать не только корректную работу алгоритма на некотором наборе тестов, но и время, ресурсы компиляции и выполнения кода, объём выводимых данных. Задачу можно включать в несколько соревнований, при этом совокупность требований и условий её выполнения может существенно отличаться. Тем самым можно значительно сэкономить ресурсы на создание новых за-

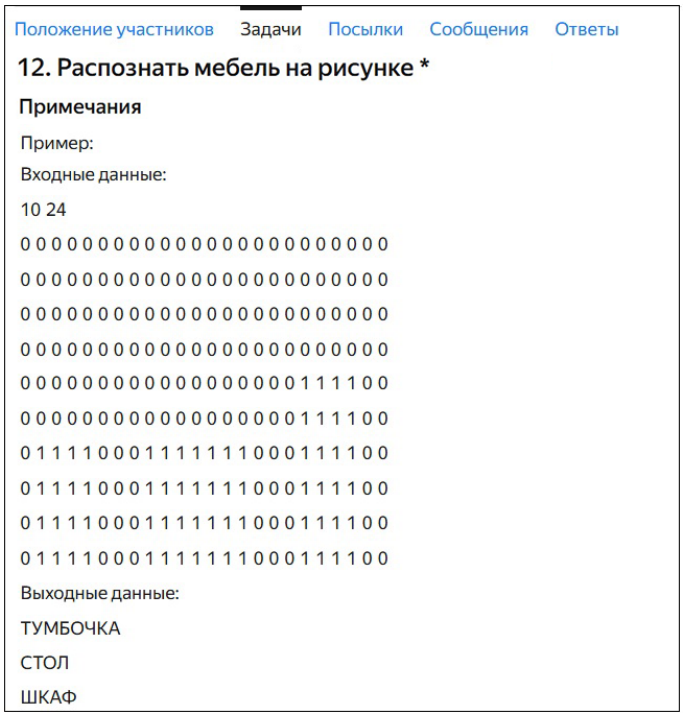


Рис. 5. Вариант теста к задаче в качестве примечания

дач. На едином банке заданий можно формировать соревнования различной сложности, ориентированные на подготовленность студентов.

5. КАК НАСТРОИТЬ ОЦЕНИВАНИЕ РЕШЕНИЙ?

В рамках рассматриваемого подхода, организаторы могут задать свою оценочную подсистему, которая каждому ответу участника (посылке) назначает баллы, таким образом рейтинг участника (его место в рейтинговой таблице) будет учитывать не только количество решённых задач. Расчет баллов и определение рейтинга также могут зависеть от других параметров, например, от количества неудачных посылок или потраченного времени [17]. В исследовании Г. О. Евстропова [18, с. 65] представлены базовые положения, на которые следует обратить внимание при построении оценочной подсистемы. На данный момент платформа Яндекс.Контест поддерживает 4 основных подхода к оцениванию участников, каждый из которых дополнительно настраивается в рамках соревнования:

SHAD — баллы за успешное решение рассчитываются как разность между «номинальной стоимостью» задачи и суммой штрафных баллов. Штрафные баллы можно назначить за неудачные посылки. Места в турнирной таблице определяются по убыванию итоговой суммы баллов.

ACM — за каждую решённую задачу начисляется один балл. При равной сумме итоговых баллов при ранжировании участников учитываются минуты, потраченные на решение задач.

IOI — баллы за решение назначаются исходя из «номинальной стоимости» задачи, но могут быть переопределены логикой постпроцессора, исходя из количества пройденных

тестов. В дополнительных настройках данного подхода нет возможности настроить систему штрафов, но можно реализовать ее в постпроцессоре за «неудачные» тесты или в зависимости от потраченных минут. Ранжирование участников выполняется по убыванию итоговой суммы баллов.

SCORING — баллы за задачу выставляются чекером или постпроцессором. Ранжирование участников выполняется по убыванию итоговой суммы баллов, но при одинаковой сумме баллов можно учитывать штрафные очки (задаётся настройками).

Следует обратить внимание на подходы: IOI и SCORING, которые учитывают результаты работы постпроцессоров и позволяют настраивать правила штрафов за неудачные попытки. Гибкие настройки позволяют учесть уровень подготовки участников соревнований. Так, например, можно оценивать ненулевыми баллами решения, даже если не все тесты были пройдены (см. рис. 6). При этом можно поощрять дополнительными баллами решения, которые прошли проверку на всех тестах. Более опытных студентов можно штрафовать за потраченное время и за посылки с ошибками.

Следует стремиться к максимальной прозрачности при оценивании решений, заранее подготовить и публично представить правила проверки ответов. Студенты должны видеть работу оценочной подсистемы, которая начисляет или снимает баллы с «номинальной стоимости», опираясь на критерии. Так как соревнования по программированию встраиваются в учебный процесс подготовки выпускников IT-направлений, на занятиях учебных курсов можно разобрать работу дополнительных модулей (генератор,

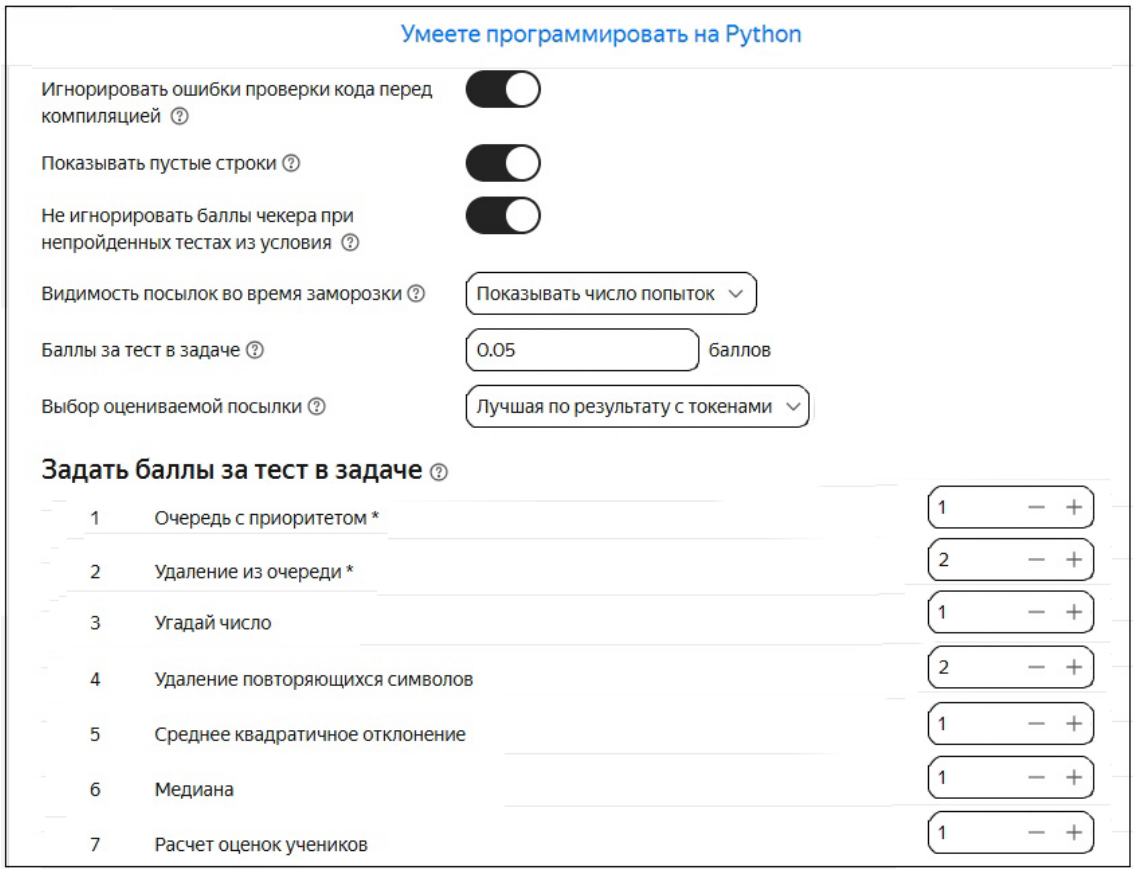


Рис. 6. Настройки контеста для оценивания решений участников

валидатор, постпроцессор, чекер, интерактор) и предложить студентам самостоятельно настроить их работу для конкретных задач. Шаблоны и примеры таких программ можно загрузить с официального сайта платформы.

Созданные студентами задачи можно использовать в качестве источника при обновлении существующего банка заданий. Знакомство с принципами работы дополнительных модулей в рамках создания контеста помогает студентам быстро интерпретировать и исправлять свои ошибки в коде, а также выстраивать тактику решения задач для получения максимального балла в соревнованиях.

6. УЧАСТНИКИ. ГДЕ И КАК СЛЕДИТЬ ЗА ХОДОМ СОРЕВНОВАНИЯ?

Участниками состязаний могут быть как зарегистрированные пользователи платформы, так и внешние пользователи. Расширенные возможности платформы по управлению ролями позволяют координировать работу пользовательских групп, которые можно создать и «связать» с учебными группами вуза. Следует также отметить проблему, с которой авторы статьи «столкнулись» при настройке групповой политики соревнования, — это ограничения по использованию кириллицы в названиях групп. Но несмотря на это неудобство, возможность оперировать учебными группами существенно сокращает затраты на работу с участниками соревнований. Платформа также обеспечивает решение всех технических вопросов по созданию и проведению командных соревнований. В этом случае итоговые баллы (и штрафы) суммируются, исходя из результатов всех участников команды.

Активности участников (набранные баллы, штрафные очки, количество посылок по задачам и т. д.) отображаются на специальном мониторе (см. рис. 7). Платформа в реальном времени генерирует данный интерфейс с актуальными результатами состязания. Тип монитора определяется выбранным подходом для системы оценивания. Интерфейс распределяет участников в порядке их рейтинга, учитывая итоговую сумму баллов и штрафы. Доступ к монитору настраивает администратор соревнования в зависимости

Соревнование по программированию на python для группы 4.305-1

Положение участников | Задачи | Посылки | Сообщения | Ответы

Последний правильный ответ: Задача 13 — lva0412 4м. 9д.

Последнее отправленное решение: Задача 12 — tok.a.78 5м. 8д.

Показать реальный монитор

№	Участник	1 9/70	2 14/56	3 6/24	4 14/20	5 13/29	6 11/26	7 11/29	8 5/50	9 10/80	10 0/34	11 4/33	12 5/42	13 3/8	14 6/12	15 3/11	16 4/16	17 4/7	Очки	Штраф
1	student4206m3	+4 9д. 21ч.	+ 10д. 4ч.	+10 10д. 20ч.	+ 10д. 23ч.	+1 11д. 1ч.	+9 11д. 3ч.	+ 11д. 4ч.	+6 11д. 8ч.	+ 11д. 9ч.	−11 13д. 8ч.	+2 11д. 22ч.	+ 12д. 21ч.	+1 13д. 6ч.	+ 13д. 6ч.	+ 13д. 8ч.	+3 13д. 9ч.	+ 13д. 19ч.	40	952125
2	Convert.ToInt32	+3 00:16	+2 00:26	+ 00:31	+ 00:33	+2 00:48	–	+ 01:05	+2 01:10	+3 01:35	−5 02:07	+1 02:20	+15 03:32	+ 5д. 1ч.	+ 5д. 1ч.	+ 5д. 1ч.	+1 5д. 1ч.	+ 5д. 1ч.	20	74334
3	lva0412	+3 11д. 11ч.	+2 10д. 12ч.	–	+ 7д. 5ч.	+ 7д. 7ч.	+ 13:05	+4 7д. 1ч.	+3 12д. 9ч.	+8 8д. 6ч.	−11 4м. 7д.	+3 4м. 6д.	+2 9д. 13ч.	+1 4м. 9д.	+2 7д. 5ч.	–	+4 4м. 7д.	+1 4м. 7д.	16	1037522
4	dimasika	+8 2м. 1д.	+3 6д. 9ч.	+1 18д. 4ч.	+ 7д. 18ч.	+3 10д. 22ч.	+1 13д. 9ч.	+ 18д. 9ч.	+2 30д. 21ч.	+11 1м. 8д.	–	–	–	–	–	–	–	–	11	456755
5	sachacok	−2 1д. 2ч.	+2 01:07	–	+ 01:34	+1 1д. 1ч.	+ 1д. 1ч.	+ 2д. 19ч.	−10 2д. 19ч.	–	−1 2д. 19ч.	–	–	−3 1д.	–	–	–	–	9	26883
6	tok.a.78	–	+ 00:00	+ 00:00	+ 00:06	–	−1 03:38	–	–	+ 00:00	–	–	−1 5м. 8д.	–	–	–	–	–	8	46827
7	Vilktaki	+3 1д.	+7 02:59	–	+ 04:03	+1 03:05	–	–	–	–	–	+22 3д. 21ч.	–	–	–	–	–	–	7	24035

Рис. 7. Монитор соревнования с промежуточным рейтингом участников

от роли пользователя. Опыт проведения соревнований показал: если актуальный монитор с посылками решений доступен не только для судей, но и для участников, можно наблюдать резкое увеличение («всплески») количества посылок к задачам, по которым кем-то из участников был получен успешный вердикт.

Такие «быстрые» решения, как правило, сырые и не проходят проверку тестированием. При этом участники тратят на эти решения время и силы. Чтобы избежать излишней эмоциональной нагрузки участников и переключить их внимание на продуктивную работу, в начале и несколько раз по ходу состязания необходимо «замораживать» монитор. Временная остановка монитора позволит участникам сосредоточиться на выборе подходов к решению и на проверке. Время и продолжительность заморозки можно регулировать в ходе проведения соревнования, опираясь на динамику изменения количества посылок к задачам.

Оперируя настройками соревнования, можно разрешить участникам продолжить выполнять задания, просматривать и сравнивать свои результаты после установленного времени завершения. Рейтинговая таблица фиксируется в момент завершения состязания (чтобы объявить результаты), но участники могут «дорешивать» предложенный набор задач, оставлять комментарии к решениям, задавать вопросы организаторам.

7. ЗАКЛЮЧЕНИЕ

С 2022 года спортивное программирование включено во Всероссийский реестр видов спорта. Данным исследованием авторы предприняли попытку обратить внимание на необходимость вовлечения студентов в такие состязания, которые тренируют навыки быстрого выбора алгоритма и скоростного безошибочного кодирования. Для успешного участия в соревнованиях по программированию участники должны иметь глубокие знания в различных научных дисциплинах, таких как топология, геометрия, дискретная математика, криптография и т. д. Это подталкивает студентов к самообразованию и к совершенствованию навыков командной работы с распределением ролей согласно сфере их интересов.

Анализ исследований [19, 20] показал прямую зависимость развития системного мышления у учащихся (студентов) от приобретения опыта решения практических задач. Этот вывод подтверждается тем, что, как отмечено в работе [21, с. 79], для перехода на уровень «описание системы» от учащихся требуется владение техниками системного мышления для глубокого анализа, либо соответствующий практический опыт. Именно такая практика развивает системное мышление, которое позволяет в будущем ставить и решать задачи из любой предметной области. Быстро разрабатывать эффективные по вычислительным ресурсам программы могут выпускники, которые на практике оценили разнообразие структур данных и базовых алгоритмов и приобрели опыт их адаптации под конкретную специфику предметной области.

Таким образом, можно сделать вывод, что опыт участия в подобных состязаниях является важным звеном всего образовательного процесса. Студенты учатся не только программировать в ограничениях, но и принимать решения в экстремальных условиях, анализировать работу модулей системы проверки, планировать порядок решения задач.

Представленный в статье подход к конструированию соревнований не требует дополнительных финансовых вложений и позволяет создавать состязания для студентов с разной начальной подготовкой. Перспективой развития данного подхода является подключение к подсистеме автоматизированной проверки решений дополнительного модуля —

детектора на «антиплагиат». Детектор, используя алгоритм «просеивания» метода «отпечатков», оценивает схожесть программных блоков для каждой новой посылки с уже принятыми решениями этой задачи [22]. Модуль не участвует в вынесении вердикта, но выявляет схожесть программ и помечает схожие фрагменты кода, информируя организаторов о возможной проблеме.

Список литературы

1. Раздел «Карьера» Санкт-Петербуржского политехнического университета. Требования и претензии работодателей к выпускникам вуза [Электронный ресурс]. URL: https://www.spbstu.ru/students/employment/demands-claims-employers-graduates/ (дата обращения: 10.07.2025).
2. Приказ Министерства образования и науки РФ от 19 сентября 2017 г. № 929 «Об утверждении федерального государственного образовательного стандарта высшего образования — бакалавриат по направлению подготовки 09.03.01 Информатика и вычислительная техника» (с изменениями и дополнениями) [Электронный ресурс]. URL: http://publication.pravo.gov.ru/document/0001201710110018 (дата обращения: 10.07.2025).
3. *Галахов Д. В.* Формирование навыков командной работы и коммуникации у будущих специалистов в области информатики и вычислительной техники с использованием проектной методологии // Образование. Наука. Научные кадры. 2024. № 1. С. 333–342. doi:10.24412/2073-3305-2024-1-333-342
4. *Селиванова И. В., Ураева Е. Е.* Технологии и методы алгоритмического программирования в образовательном процессе при подготовке будущих it-специалистов Ученые записки // Электронный научный журнал Курского государственного университета. 2024. № 1. С. 267–273.
5. *Крайванова В. А., Крючкова Е. Н.* Олимпиадное программирование как эффективный инструмент подготовки профессиональных программистов // Вестник Новосибирского государственного университета: Информационные технологии. 2012. № 4. С. 51–56.
6. *Горчаков Л. В., Стась А. Н., Карташов Д. В.* Обучение программированию с использованием системы Ejudge // Вестник Томского государственного педагогического университета. 2017. № 9. С. 109–112.
7. *Андреева Н. Л., Федорова А. Г.* Саратовская школа программистов. Региональный опыт // Компьютерные инструменты в образовании. 2002. № 6. С. 51–55.
8. *Алексеев А. В, Карелин В. А., Синицын С. В.* Студенческие соревнования по программированию в Югорском государственном университете // Вестник Югорского государственного университета. 2011. № 3. С. 7–9.
9. Платформа Яндекс.Контест [Электронный ресурс]. URL: https://contest.yandex.ru/edu (дата обращения: 10.07.2025).
10. Обзор платформ по проведению онлайн-чемпионатов. URL: https://temofeev.ru/info/articles/obzor-platform-po-provedeniyu-onlayn-chempionatov/ (дата обращения: 10.07.2025).
11. Платформа CodeForces [Электронный ресурс]. URL: https://codeforces.com/ (дата обращения: 10.07.2025).
12. Как создать мэшап на Codeforces [Электронный ресурс]. URL: https://mksegment.ru/a/kak-sozdat-mehshap-na-codeforces (дата обращения: 10.07.2025).
13. Платформа Poligon [Электронный ресурс]. URL: https://polygon.codeforces.com/ (дата обращения: 10.07.2025).
14. Документация и исходные коды библиотеки testlib [Электронный ресурс]. URL: https://github.com/MikeMirzayanov/testlib (дата обращения: 10.07.2025).
15. *Котелина Н. О., Попова Н. К.* Подготовка интернет-тура чемпионата по программированию на Yandex.Contest // Вестник Сыктывкарского университета. 2018. № 26. С. 73–79.
16. *Юрьев И. А., Гостева И. Н.* Механизм и технология настройки онлайн-сервиса автоматизированной проверки заданий по информатике // Электронный научный журнал Курского государственного университета. 2023. № 2. С. 47–54.
17. *Станкевич А. С.* Общий подход к подведению итогов соревнований по программированию

при использовании различных систем оценки // Компьютерные инструменты в образовании. 2011. № 2. С. 27–38.
18. *Евстропов Г. О.* Системы оценивания в задачах с автоматической проверкой на олимпиадах по программированию // Информатика и образование. 2016. № 3. С. 65–67.
19. *Каримова Б. С.* Развитие системного мышления в опережающем обучении // Austrian Journal of Humanities and Social Sciences. 2015. № 9–10. С. 64–67.
20. *Галиев Т. Т., Ускенбаева Д. А.* Формирование системного мышления учащихся в процессе обучения // International scientific review. 2016. № 18. С. 92–93.
21. *Ковалев Г. О.* Системное мышление как компетенция // Вестник науки и образования. 2017. № 9. С. 72–79.
22. *Половикова О. Н., Иванова В. Е.* Разработка детектора автоматической проверки на плагиат блоков программного кода для образовательной среды // Высокопроизводительные вычислительные системы и технологии. 2020. № 1. С. 173–178.

Поступила в редакцию 05.03.2025, окончательный вариант — 10.07.2025.

Половикова Ольга Николаевна, кандидат физико-математических наук, доцент, доцент кафедры информатики, Алтайский государственный университет, ✉ polovikovaol@yandex.ru

Смолякова Лариса Ленгардовна, старший преподаватель кафедры информатики, Алтайский государственный университет, knaus.larisa@gmail.com

Computer tools in education, 2025
№ 2: 81–95
http://cte.eltech.ru
doi:10.32603/2071-2340-2025-2-81-95

Programming Competitions on the Yandex.Contest Platform: Who Needs Them and How to Create Them

Polovikova O. N.[1], Cand. Sc., Docent, ✉ polovikovaol@yandex.ru, orcid.org/0000-0002-9403-1195
Smolyakova L. L.[1], Senior Lecturer, knaus.larisa@gmail.com, orcid.org/0000-0001-8547-0324
[1]Altai State University, 61 Lenin Ave., 656049, Barnaul, Russia

Abstract

This study emphasizes the need for the IT students to gain experience in solving practical problems. Active participation in competitive programming, regardless of initial skills, helps students gain such experience. The article presents an approach to creating contests using Testlib library on the free domestic platform Yandex.Contest. The software modules cover the main stages of the competition life cycle, from generating tests for tasks to summarizing results. The use of a checker and postprocessor allows the implementation of the evaluation system logic, taking into account the points and penalties scored by the participants. The task configuration settings allow organizers to impose limits on memory usage and code execution time. The interactor program forms a separate set of requirements for the method of solving tasks, which is an important condition for holding competitions within the educational process. By creating a bank of tasks for competitions,

taking into account the material covered and curriculum, it is possible to «integrate» competitions into the educational process, alternating between individual and team stages. The proposed approach has been tested in practice: twice a semester junior students from one of the institutes of Altai State University participate in a competition, honing their skills in writing program code quickly and accurately. Experience with such competitions has shown that students and teachers are willing to embrace new forms of learning.

Keywords: *programming competition, automated code review, checker, interactor, postprocessor, test generation, penalty points, solution points, participant monitor, task bank, tests, program code.*

Citation: O. N. Polovikova and L. L. Smolyakova, “Programming Competitions on the Yandex.Contest Platform: Who Needs Them and How to Create Them,” Computer tools in education, no. 2, pp. 81–95, 2025 (in Russian); doi:10.32603/2071-2340-2025-2-81-95

References

1. “Requirements and claims of employers to university graduates,” in *www.spbstu.ru*, 2025. [Online] (in Russian). Available: https://www.spbstu.ru/students/employment/demands-claims-employers-graduates/
2. Ministry of Education and Science of the Russian Federation, “On Approval of the Federal State Educational Standard of Higher Education - Bachelor’s Degree in the Field of Study 09.03.01 Informatics and Computer Engineering (as amended and supplemented),” *Order No. 929, Sep. 19, 2017*, 2017. [Online] (in Russian). Available: http://publication.pravo.gov.ru/document/0001201710110018
3. D. V. Galakhov, “Formation of teamwork and communication skills among future specialists in the field of it using project methodology,” *Obrazovaniye. Nauka. Nauchnyye kadry = Education. Science. Scientific personnel*, no. 1, pp. 333–342, 2024 (in Russian); doi:10.24412/2073-3305-2024-1-333-342
4. I. V. Selivanova and E. E. Uraeva, “Technologies and methods of algorithmic programming in the educational process during preparation of future IT specialists,” *Uchenye zapiski: Elektronnyi nauchnyi zhurnal Kurskogo gosudarstvennogo universiteta*, no. 1, pp. 267–273, 2024 (in Russian).
5. V. A. Kraivanova and E. N. Kryuchkova, “Olympiad programming as an effective tool for the training of professional programmers,” *Vestnik NSU. Series: Information Technologies*, no. 4, pp. 51–56, 2012 (in Russian).
6. L. V. Gorchakov, A. N. Stas, and D. V. Kartashov, “Programming training using the ejudge system,” *Tomsk state pedagogical university bulletin*, no. 9, pp. 109–112, 2017 (in Russian); doi:10.23951/1609-624x-2017-9-109-112
7. N. L. Andreeva and A. G. Fedorova, “Saratov school of programmers. Regional experience,” *Computer tools in education*, no. 6, pp. 51–55, 2002 (in Russian).
8. A. V. Alekseev, V. A. Karelin, and S. V. Sinitsyn, “Student programming contest at Yugra State University,” *Yugra State University Bulletin*, no. 3, pp. 7–9, 2011 (in Russian).
9. Yandex, *Yandex.Contest platform*, 2025. [Online] (in Russian). Available: https://contest.yandex.ru/edu
10. Y. Temofeev, “Overview of platforms for conducting online championships,” *Temofeev.ru: IT, business, personal growth*, 2025. [Online] (in Russian). Available: https://temofeev.ru/info/articles/obzor-platform-po-provedeniyu-onlayn-chempionatov/
11. Codeforces, *CodeForces platform*, 2025. [Online] (in Russian). Available: https://codeforces.com/
12. Mksegment, “How to create a mashup on Codeforces,” *MK Segment: Programming and Development*, 2025. [Online] (in Russian). Available: https://mksegment.ru/a/kak-sozdat-mehshap-na-codeforces
13. Polygon, *Polygon platform*, 2025. [Online]. Available: https://polygon.codeforces.com/
14. M. Mirzayanov, *Documentation and source code of the testlib library*, 2025. [Online]. Available: https://github.com/MikeMirzayanov/testlib
15. N. O. Kotelina and N. K. Popova, “The preparation of the online round of the championship on programming on Yandex.Contest platform,” *Bulletin of Syktyvkar University Series 1: Mathematics. Mechanics. Informatics*, no. 26, pp. 73–79, 2018 (in Russian).
16. I. A. Yuriev and I. N. Gosteva, “Mechanism and technology of setting up an online service for automated verification of computer science assignments,” *Elektronnyi nauchnyi zhurnal Kurskogo gosudarstvennogo universiteta*, no. 2, pp. 47–54, 2023 (in Russian).

17. A. S. Stankevich, “General approach to summarizing the results of programming competitions when using different scoring systems,” *Computer tools in education*, no. 2, pp. 27–38, 2011 (in Russian).
18. G. O. Evstropov, “Scoring systems used for automatic evaluation in programming competitions in informatics,” *Informatics and education*, no. 3, pp. 65–67, 2016 (in Russian).
19. B. S. Karimova, “Development of systems thinking in advanced training,” *Austrian Journal of Humanities and Social Sciences*, no. 9-10, pp. 64–67, 2015.
20. T. T. Galiev and D. A. Uskenbaeva, “The formation of the system thinking of pupils in the learning process,” *International scientific review*, no. 18, pp. 92–93, 2016 (in Russian).
21. G. O. Kovalev, “Systems thinking as a competence,” *Vestnik nauki i obrazovaniia*, no. 9, pp. 72–79, 2017 (in Russian).
22. O. N. Polovikova and V. E. Ivanova, “Development of a detector of automatic check for plagiated blocks of program code for educational environment,” *Vysokoproizvoditelnye vychislitelnye sistemy i tekhnologii*, no. 1, pp. 173–178, 2020 (in Russian).

Received 05-03-2025, the final version — 10-07-2025.

Olga Polovikova, Candidate of Sciences (Phys.-Math.), Docent, Altai State University, ✉ **polovikovaol@yandex.ru**

Larisa Smolyakova, Senior Lecturer, Department of Informatics, Altai State University, Altai State University, knaus.larisa@gmail.com