1 этап

Методика проведения испытания

Вступительное испытание 1-го этапа проводится очно в Университетском Лицее №1511 или №1523 в соответствии со списками распределения (становятся известны накануне даты соответствующего вступительного испытания).

В течение 90 минут абитуриент решает задачи с **кратким ответом** в системе автоматизированной проверки решений Яндекс.Контест. Каждое задание оценивается либо в 0 баллов, либо в полный балл, указанный в непосредственно условии задания. Максимальный балл за выполнение испытания 1-го этапа — 100 баллов, и вычисляется как сумма полученных баллов за все оценённые собеседующим задания. При решении заданий первого этапа можно пользоваться черновиком, который выдают в аудитории.

Во время написания экзамена нельзя пользоваться любыми справочными материалами, гаджетами, смартчасами, нейросетями, средставми аудио и видеосвязи, переговариваться с остальными абитуриентами. В случае нарушения данных правил вы можете быть аннулированы и удалены с прохождения испытания.

Перечень разделов и тем для подготовки

Системы счисления.

Позиционные системы счисления. Основание системы, разряды и вес разряда. Представление чисел в двоичной, восьмеричной, шестнадцатеричной и десятичной системах. Перевод чисел между системами счисления прямыми и обратными методами. Использование степеней двойки для анализа числовых выражений. Определение количества цифр, количества единиц в записи числа, работа со степенными разложениями. Решение задач на смешанные выражения, содержащие степени, суммы и разности чисел в различных системах.

Логика.

Высказывания и логические значения. Операции «и», «или», «не», импликация, эквивалентность. Истинностные таблицы для логических выражений. Приоритет логических операций. Преобразование составных логических условий. Поиск набора значений, при которых выражение истинно или ложно. Решение задач на логику, включая анализ логических схем и рассуждений.

Выполнение и анализ простых алгоритмов.

Линейные, разветвляющиеся и циклические алгоритмы. Порядок выполнения команд, изменение значений переменных, пошаговая трассировка алгоритма. Анализ арифметических выражений, условий, вложенных и последовательных команд. Определение результата работы алгоритма по его тексту, анализ граничных случаев. Типичные задачи: вычисление значений после последовательности присваиваний, проверка условий, работа циклов.

Кодирование графической информации.

Пиксельная модель изображения. Представление изображения как прямоугольной таблицы пикселей. Градации яркости и глубина цвета. Объём графического файла, вычисление размера изображения в битах и байтах. Кодирование цветных изображений: модели RGB и простейшие способы кодирования

цвета. Работа с квадратами, прямоугольниками и фрагментами изображения. Задачи на определение объёма, изменение разрешения и глубины цвета.

Кодирование данных, комбинаторика, системы счисления.

Комбинаторные принципы: правило произведения, правило суммы. Подсчёт числа возможных сообщений фиксированной длины. Мощность алфавита и длина сообщения. Связь между количеством комбинаций, длиной кода и числом информационных бит. Применение систем счисления в комбинаторике: минимальное число разрядов, достаточное для кодирования. Решение задач на перебор вариантов, поиск однозначной или неоднозначной расшифровки, построение и анализ кодов.

Вычисление информационного объема сообщения.

Понятие информационного объёма. Алфавит и количество символов. Определение минимального количества бит для кодирования одного символа. Объём сообщения в битах и байтах. Связь между глубиной кодирования, длиной сообщения и объёмом. Задачи на вычисление информации в случаях: фиксированная длина кода, переменная длина кода, составные сообщения.

Выполнение алгоритмов для исполнителя.

Модель исполнителя на клетчатой плоскости. Команды перемещения, условия свободности соседних клеток. Циклы и условия в программах для исполнителя. Анализ поведения исполнителя в лабиринте, определение допустимых и недопустимых маршрутов. Построение модели движения, учёт стен и ограничений. Определение конечного положения исполнителя после выполнения алгоритма.

Составление запросов для поисковых систем с использованием логических выражений.

Принцип работы поисковых запросов. Использование логических операторов в поисковых системах: `AND, OR, NOT`. Составление точных запросов с логическими выражениями. Уточнение результатов поиска через ключевые слова, исключения, уточнения и комбинации условий. Анализ корректности запросов и их влияния на результаты поиска.

Поиск алгоритма минимальной длины для исполнителя.

Оптимизация алгоритмов движения. Понятие минимальной длины алгоритма. Сравнение разных вариантов программ, определение лишних шагов. Сокращение последовательностей команд, удаление повторов. Принципы оптимизации траектории: кратчайший путь, уменьшение числа шагов, выбор направления. Решение задач на поиск кратчайшей программы, приводящей к нужному результату.

Использование информационных моделей.

Информационные модели: таблицы, графы, логические схемы, правила соответствия. Выбор типа модели в зависимости от задачи. Описание объектов и связей, заполнение таблиц, интерпретация данных. Использование графов для моделирования маршрутов, зависимостей и переходов. Анализ табличных и графовых моделей: поиск пути, подсчёт вариантов, проверка условий, извлечение информации. Применение информационных моделей в задачах классификации, перебора, анализа структурированных данных.

Литература и интернет-источники для подготовки

1. **Босова, Л. Л. Информатика. 9–11 классы : учебник для общеобразоват. организаций** / Л. Л. Босова, А. Ю. Босова. — М. : БИНОМ. Лаборатория знаний, 2022. — 352 с.
2. **Поляков, К. Ю. Информатика. Базовый курс. 9–11 классы : учебник для общеобразоват. школ** / К. Ю. Поляков, Е. А. Еремин. — СПб. : Питер, 2023. — 400 с.
3. **Семакин, И. Г. Информатика. 9–11 классы : базовый уровень : учебник** / И. Г. Семакин, Л. А. Залогова. — М. : БИНОМ. Лаборатория знаний, 2021. — 352 с.
4. **Фоксфорд. Учебник. Информатика : теория, логика, алгоритмы** [Электронный ресурс].
5. **Решу ЕГЭ. Информатика : задачи по теории информации, кодированию, количеству информации, логике, моделированию** [Электронный ресурс].

2 этап

Методика проведения испытания

Вступительное испытание 2-го этапа также проводится очно в Университетском Лицее №1511 или №1523 в соответствии со списками распределения (становятся известны накануне даты соответствующего вступительного испытания).

В течение 90 минут абитуриент решает задачи с проверкой написанного кода на наборе тестов в системе автоматизированной проверки решений Яндекс.Контест. Разрешённые языки для вступительного испытания — **Python, C/C++, Java.** Гарантируется, что задачи варианта возможно решить на полный балл на любом из этих языков. Для решения разных задач и для разных попыток сдачи конкретной задачи можно отдельно выбирать язык программирования из данного списка.

Количество баллов за каждую задачу будет описано в условии каждой задачи в разделе "Система оценивания". В случае полностью правильного решения задачи (положительного результата проверки на всём наборе тестов) выставляется максимальный балл за задачу и вердикт **OK**; в случае частично верного решения (результат проверки положительный только на некоторых тестов из всего набора) выставляется пропорциональный балл по отношению к количеству пройденных тестов; в случае полностью неверного решения выставляется 0 баллов.

Во время написания экзамена нельзя пользоваться любыми справочными материалами, гаджетами, смартчасами, нейросетями, средствми аудио и видеосвязи, переговариваться с остальными абитуриентами. В случае нарушения данных правил вы можете быть аннулированы и удалены с прохождения испытания.

Описание некоторых вердиктов тестирующей системы после отправки решения участника:

OK **Тест пройден корректно;**

WA **Неверный ответ на тест** — ответ на тест от программы участника не совпадает с верным ответом на тест;

TL **Превышение времени работы программы** — на конкретном тесте программа выполняется дольше указанного в задаче времени на выполнение;

ML **Превышение используемой памяти в программе** — на конкретном тесте программа занимает большую память, чем указано в ограничениях к памяти в задаче;

RE **Ошибка исполнения программы** — во время исполнения кода на данных из теста возникла программная ошибка;

CE Ошибка компиляции программы — во время компиляции программы произошла ошибка, на тестирование решение отправлено быть не может;

MI — Решение аннулировано в связи с нарушением правил проведения экзамена.

Максимальный балл за выполнение испытания 1-го этапа — 100 баллов, и вычисляется как сумма полученных баллов за все решённые в тестирующей системе задания.

Перечень разделов и тем для подготовки

Арифметические операции.

Арифметические выражения и их вычисление. Приоритет операторов, порядок вычислений. Операции сложения, вычитания, умножения, деления (в том числе целочисленного), остатка от деления. Типы данных в языках Python, C++, Java. Преобразование типов. Ошибки целочисленного деления и особенности работы оператора `//` в Python, оператора `/` и `%` в C++ и Java. Решение задач на вычисления, составление арифметических выражений, анализ значения переменных.

Условный оператор.

Понятие условия. Булевы выражения: сравнение чисел, логические операции, составные условия. Конструкции ветвления: `if`, `if-else`, `else-if` (Python: `elif`). Особенности синтаксиса условий в Python, C++, Java. Вложенные условия, каскад условий, корректность логики программы. Решение задач с разветвляющимся алгоритмом, производных от следующих базовых алгоритмов: выбор максимума, проверка свойств числа, ветвление по диапазонам.

Циклы с условием и счётчиком.

Назначение циклов. Циклы с известным числом повторений (`for`) и циклы с условием (`while`). Переменные-счётчики и управляющие переменные. Порядок изменения значений. Организация циклов в Python (`for ... in range`, `while`), C++ (`for`, `while`), Java (`for`, `while`). Бесконечные циклы и условия остановки. Решение задач, производных от следующих базовых алгоритмов: подсчёт суммы и количества элементов, поиск минимума/максимума, обработка цифр числа, перебор вариантов.

Одномерные массивы.

Представление массива в памяти. Индексация элементов: нумерация с нуля (Python, C, C++, Java). Обращение к элементам массива через индекс.

Создание и инициализация массивов: списки в Python, статические и динамические массивы в C и C++, массивы в Java. Правила задания длины массива, доступ к элементам, попытка выхода за границы массива как типовая ошибка.

Последовательный обход массива: цикл по индексам и цикл по элементам (Python). Получение значений элементов, их изменение, обработка каждого элемента массива в цикле. Типовые алгоритмы: подсчёт количества элементов, удовлетворяющих условию; вычисление суммы элементов; поиск максимального и минимального значений; поиск индекса нужного элемента; проверка наличия элемента.

Использование массивов для решения задач перебора: анализ числовых данных, обработка последовательностей, формирование новых массивов по заданному правилу. Типичные ошибки при работе с массивами: неправильная инициализация, неверные границы циклов, изменение массива во время перебора.

Двумерные массивы.

Понятие двумерного массива (матрицы, таблицы) как структуры данных, имеющей два индексных измерения: строку и столбец. Представление двумерного массива как массива массивов (Python, C, C++, Java). Индексация: порядок указания индексов (строка, затем столбец), от 0 или от 1 — в зависимости от языка.

Перебор двумерного массива: вложенные циклы. Прямой обход по строкам, обход по столбцам, обход по выбранным областям таблицы. Алгоритмы обработки таблиц: подсчёт количества элементов, удовлетворяющих условию; сумма элементов строки или столбца; поиск максимального/минимального элемента; подсчёт значений в прямоугольной области; сравнение элементов внутри строки или столбца.

Функции.

Понятие функции в программировании как способа структурирования программы и повторного использования кода. Определение функции и вызов функции. Передача параметров: позиционные аргументы, именованные аргументы (Python), передача по значению и по ссылке (C/C++), объектные параметры и примитивные типы (Java). Возвращаемые значения, использование оператора `return`. Область видимости переменных: локальные и глобальные переменные, особенности скрытия переменных. Функции как инструмент разбиения алгоритма на подзадачи. Решение задач на построение функций: вычисление значений по формуле, проверка условий, операции с массивами. Рекурсивные функции: идея рекурсии, базовый случай, рекурсивный шаг. Примеры типовых рекурсивных алгоритмов: вычисление факториала, вычисление чисел Фибоначчи, обход дерева или таблицы. Анализ корректности рекурсивных алгоритмов, риски переполнения стека.

Динамическое программирование.

Понятие динамического программирования (ДП) как способа оптимизации перебора. Разбиение задачи на подзадачи, которые перекрываются. Хранение результатов в таблицах или массивах для предотвращения повторных вычислений.

Типовые задачи: вычисление чисел Фибоначчи, поиск количества путей в таблице, задача оптимизации маршрута, простые задачи на последовательности.

Структура решения задач ДП: формулировка состояния, определение переходов, инициализация базовых значений, порядок заполнения массива/таблицы, вычисление ответа.

Анализ сложности алгоритмов динамического программирования.

Бинарный поиск.

Идея бинарного поиска: поиск элемента в отсортированном массиве путём деления диапазона пополам. Условия применения бинарного поиска: массив должен быть отсортирован. Понятие границ поиска: левая и правая границы. Алгоритм проверки середины и сдвига границ. Циклическая и рекурсивная формы алгоритма. Частые ошибки: неверный пересчёт середины, бесконечный цикл, выход за границы массива.

Применение бинарного поиска: поиск элемента в массиве, поиск первого или последнего вхождения, поиск границы (аналог функций `lower_bound` и `upper_bound` в C++), бинарный поиск ответа (поиск минимального подходящего значения параметра или решения задачи монотонного характера).

Теория чисел.

Основные числовые свойства: делимость, простые и составные числа, признаки делимости. Нахождение всех делителей числа, количество делителей, сумма делителей. Разложение числа на простые множи-

тели. Наибольший общий делитель (НОД) и наименьшее общее кратное (НОК). Алгоритм Евклида и расширенный алгоритм Евклида. Быстрое возведение в степень и его применение.

Применение теории чисел в задачах программирования: проверка числа на простоту, генерация последовательностей по числовым свойствам, анализ делимости элементов массива, задачи с использованием НОД и НОК, задачи на свойства чисел и разложение, использование числовых свойств в оптимизационных и поисковых алгоритмах.

Решение задач на свойства чисел, оптимизацию перебора и эффективные вычисления.

Литература и интернет-источники для подготовки

1. **Яндекс.Лицей. Самостоятельные онлайн-курсы по Python и Алгоритмам** [Электронный ресурс].
2. **Образовательный центр «Сириус». Онлайн-платформа программирования для школьников** [Электронный ресурс].
3. **Решу ЕГЭ. Информатика : задания по программированию для подготовки школьников** [Электронный ресурс].