

Определение и свойства алгоритма

Разберём, что такое алгоритм, как он устроен и почему его свойства важны.



Зачем изучать алгоритмы в информатике?

Алгоритмы лежат в основе работы поисковиков, навигаторов, мессенджеров, игр, банковских сервисов и робототехники. В 9 классе это ключ к пониманию ветвлений, циклов и поиска ошибок в программах.

3

Алгоритм включает входные данные, обработку и получение результата.

Учебный курс информатики 9 класса

У алгоритма всегда есть входные данные, действия по обработке и конечный результат выполнения; это помогает точно описать решение задачи.



Исполнитель алгоритма и система команд

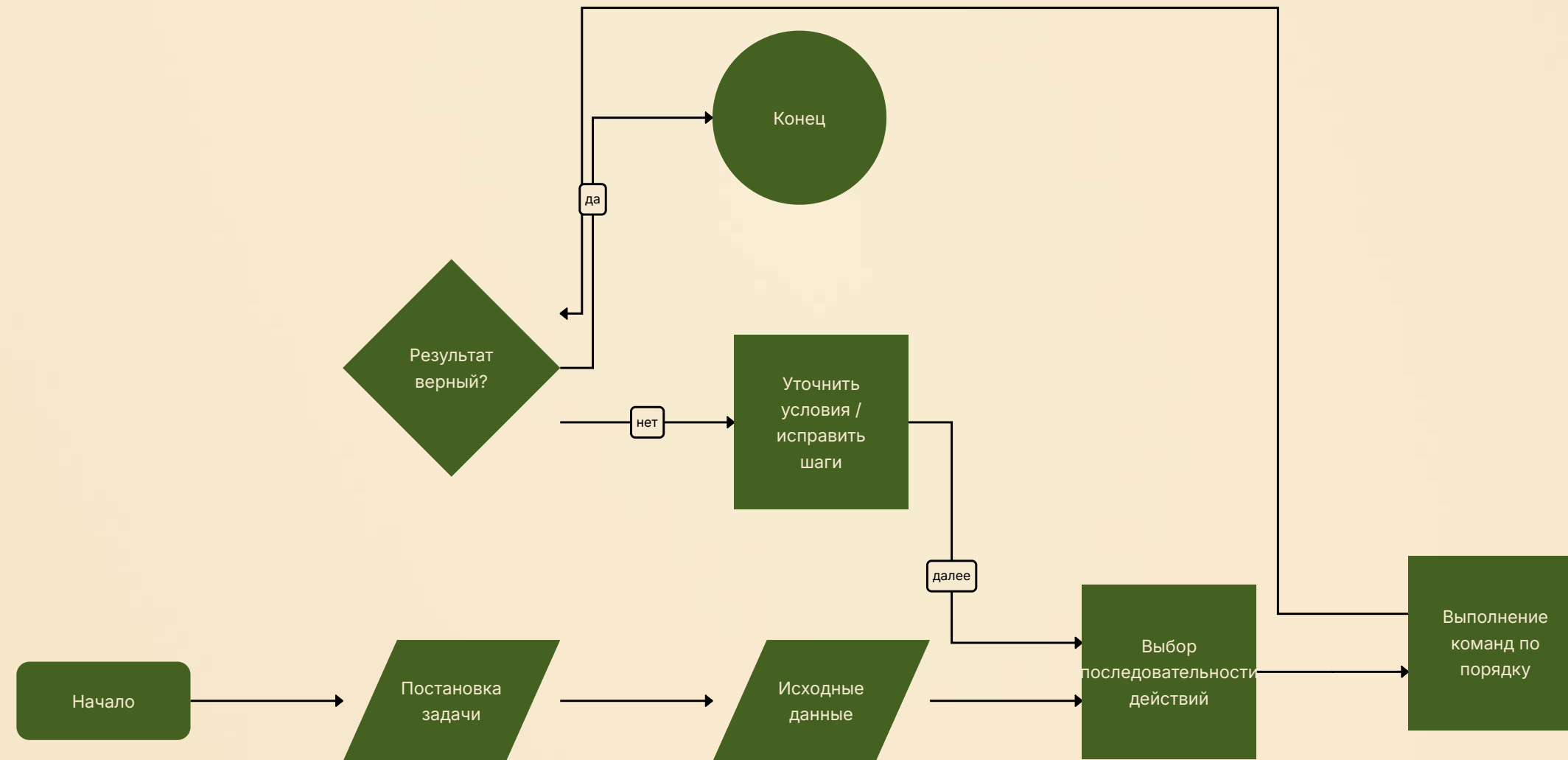
2 Система команд задаёт набор допустимых действий. Если алгоритм включает команду вне системы команд, исполнитель не сможет её выполнить, а значит, алгоритм перестаёт быть точной инструкцией.

1 Исполнитель алгоритма — человек, компьютер, робот или программа, которые умеют выполнять команды. Важно, что он действует по своим правилам и не может выполнять то, чего «не знает» в рамках обучения.

3 Команда должна быть однозначной и выполнимой без догадок. Один и тот же замысел можно записать по-разному: для человека — понятными словами, для компьютера — формальными командами.

Как алгоритм приводит к решению задачи

Постановка задачи, данные, выполнение шагов, проверка и завершение

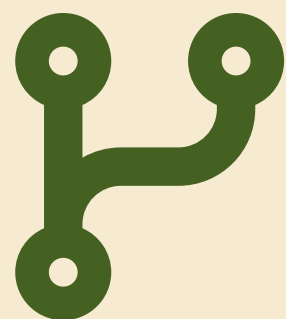


Свойство дискретности



Шаги вместо «одного действия»

Алгоритм состоит из отдельных, логически завершённых шагов. Это позволяет описывать решение поэтапно и точно проверять, что именно происходит на каждом шаге в программе или инструкции.



Шаг можно развивать деталями

Дискретность даёт возможность уточнять отдельные шаги: например, что именно значит «вычислить» или «сохранить». Так алгоритм становится пригодным для машинного исполнения.



Каждый шаг имеет своё место

Шаги идут в понятном порядке: открыть файл, прочитать данные, выполнить вычисление, сохранить ответ. Такое деление помогает избежать пропусков при написании алгоритма.



Удобно для учебной отладки

Когда шаги выделены, проще найти ошибку: какой командой нарушен ход решения? Это свойство помогает учиться отладке и корректно составлять инструкции.

Свойство определённости

1

Каждая команда алгоритма должна иметь только один смысл. Формулировки вроде «сделай правильно» слишком расплывчаты: исполнитель не может понять, какие именно действия нужны в конкретной ситуации.

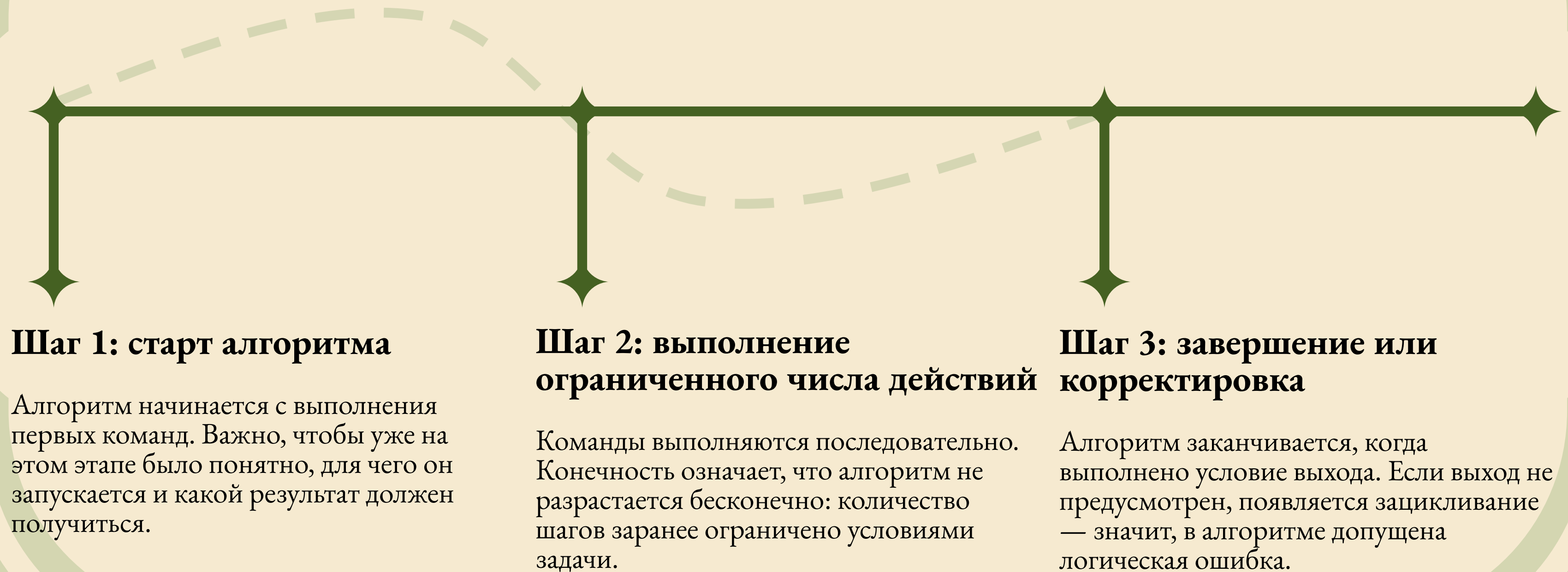
2

Уточняйте действие и объект. Например, «сложи 7 и 5» задаёт конкретную операцию и числа; «перейди к следующей строке» определяет перемещение по шагам; «сравни А и В» задаёт критерий.

3

Если команда допускает несколько трактовок, решения могут получаться разными. Поэтому определённость делает алгоритм надёжным: при одинаковых входных данных исполнитель получает один и тот же результат.

Свойство конечности



Основные свойства алгоритма

■ Дискретность ■ Определённость ■ Конечность
■ Результативность ■ Массовость



Доли на диаграмме условные: они лишь показывают, какие свойства особенно критичны для «точной инструкции».

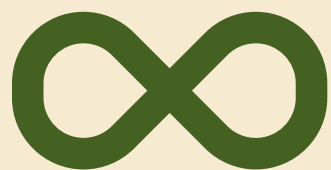
Без определённости и конечности алгоритм не будет надёжно исполняемым. Остальные свойства дополняют картину и помогают проверять корректность решения.

Учебный курс информатики 9 класса

Свойство результативности

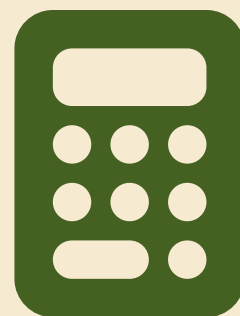
- 1 Результатом алгоритма является полезный итог: найденное число, готовый файл, выполненное построение или ответ к задаче. После выполнения человек должен увидеть завершённый смысловой outcome, а не набор команд.
- 2 Алгоритм считают результативным, если он доводит выполнение до конца с нужным итогом. Важно отличать «делал шаги» от «получил решение»: цепочка сама по себе не является результатом.
- 3 В школьных заданиях результативность проверяют по тому, решена ли исходная задача. Если ответ неверный или не получен, значит, алгоритм не приводит к нужному итогу.
- 4 Например, алгоритм вычисления должен завершаться числом по условию; алгоритм поиска — сообщением о найденном элементе. Тогда свойство результативности легко подтвердить на практике.

СВОЙСТВО МАССОВОСТИ



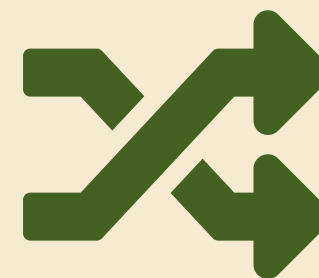
Алгоритм для целого класса задач

Массовость означает, что алгоритм решает не один конкретный случай, а группу похожих задач. Один и тот же порядок действий подходит многим наборам входных данных, а не «только для примера».



Сумма двух чисел — общий подход

Алгоритм нахождения суммы двух чисел работает для любых значений: целых, рациональных, положительных и отрицательных. Исполнитель просто выполняет команды и получает корректный результат для каждого набора.



Сравнение для объектов одного типа

Алгоритм сравнения применим ко множеству пар объектов одного вида: например, двух чисел или двух строк по заданному правилу сравнения. Главное — сохраняется одинаковая логика обработки входных данных.

Свойства алгоритма и их смысл

Свойство	Краткий смысл	Школьный пример
Дискретность; определённость	Шаги отделены; каждая команда имеет один смысл	Открыть файл → прочитать данные; «сложи А и В» вместо «сделай как надо»
Конечность; результативность	Выполнение завершается; появляется полезный итог	Проверить условие → вывести ответ «верно/неверно»; завершить после вычисления
Массовость	Подходит для многих похожих задач	Алгоритм вычисления площади прямоугольника для разных пар а и b

Свойства алгоритма описывают, насколько инструкция будет понятной и пригодной для решения задач. Они работают вместе: без одного свойства алгоритм может стать неполным, неточным или непроверяемым.

Если алгоритм дискретен, определён и конечен, его можно правильно выполнить. Результативность гарантирует полезный итог, а массовость превращает задание в общий способ решения.

Учебный курс информатики 9 класса

Алгоритмы в жизни и в учебных задачах



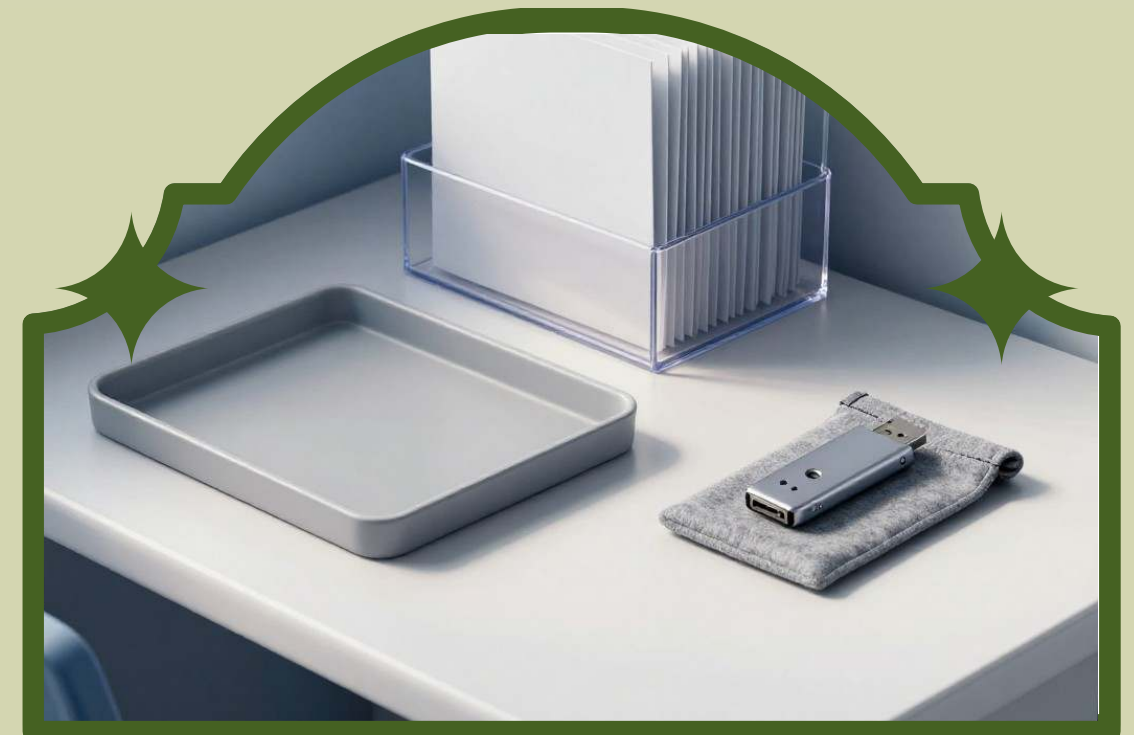
Маршрут до школы: всегда по шагам

Ученик идёт по привычному маршруту: выйти из дома, перейти дорогу по светофору, найти нужный поворот и проверить ориентир. Если один шаг пропустить, путь окажется другим. Поэтому маршрут работает как алгоритм: последовательность помогает получить предсказуемый результат.



Кулинарная инструкция: последовательность превращает идеи в блюдо

Приготовление простого блюда обычно начинается с подготовки: измерить ингредиенты, выполнить нагрев по времени и остановиться на нужной степени готовности. Рецепт — это алгоритм, потому что он задаёт точные действия и завершение. Благодаря этому блюдо получается одинаково у разных людей при сходных условиях.



Поиск и сохранение: техника помогает, когда есть правила

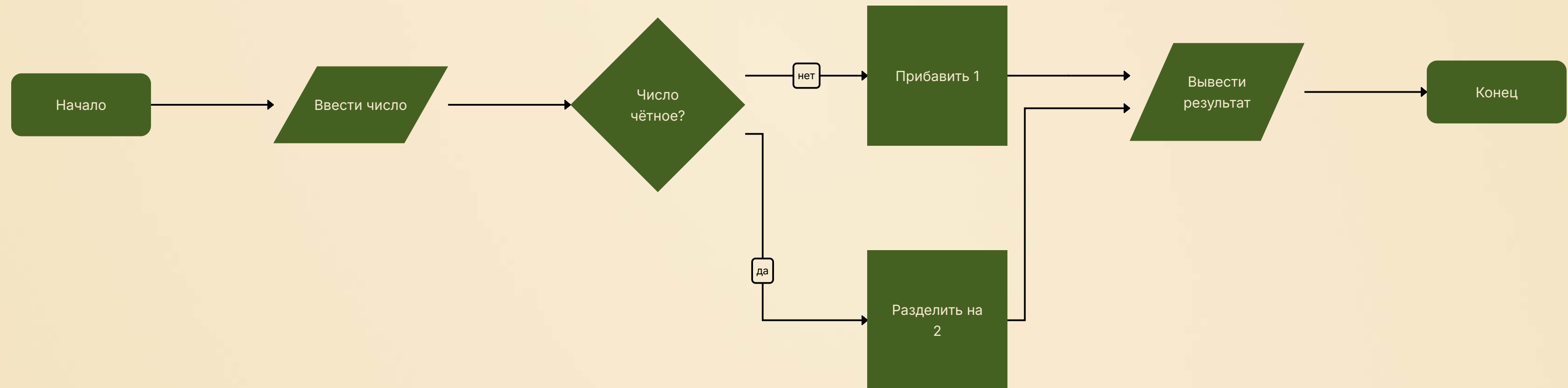
Чтобы найти слово в словаре или сохранить документ на компьютере, нужны понятные команды: открыть нужное приложение, ввести критерий поиска, выбрать нужную строку и подтвердить сохранение. Пошаговые действия уменьшают ошибки и экономят время. Так алгоритмы помогают в учебе и повседневной работе.

Способы записи алгоритмов

- 1 Алгоритмы записывают словами, в виде нумерованного списка, блок-схемой или программным кодом. Выбор формы зависит от того, кто будет выполнять алгоритм: человек или компьютер.
- 2 Для школьной информатики особенно полезны словесная запись и блок-схема: они показывают порядок действий и помогают понять логику решения. Кроме того, их легко проверять на ошибки.
- 3 Нумерованный список удобен, когда шаги следуют строго один за другим. Он позволяет ясно нумеровать действия и не пропускать важные операции при составлении и чтении алгоритма.
- 4 Программный код — это запись алгоритма на языке формальных команд. Компьютер выполняет такие команды однозначно, поэтому важно соблюдать правильность команд и структуру программы.

Блок-схема для ветвления: «чётное или нечётное?»

Логика зависит от условия: проверяем делимость числа на 2 и выбираем путь «да» или «нет».



Пример линейного алгоритма

1

Линейный алгоритм выполняется строго по порядку без ветвлений и повторений. Например: ввести два числа, затем выполнить вычисление суммы и сразу перейти к следующему шагу.

2

Далее алгоритм записывает полученную сумму в нужное место и формирует проверку результата. После этого остаётся завершить работу и сообщить, что действие выполнено.

3

Такие алгоритмы считают самыми простыми, потому что у них один путь выполнения. Именно поэтому их часто используют в первых упражнениях по информатике и математике.

Пример алгоритма с ветвлением

1

В алгоритме с ветвлением следующий шаг зависит от условия. Например: проверить, является ли число чётным, и выбрать действие в зависимости от ответа проверки.

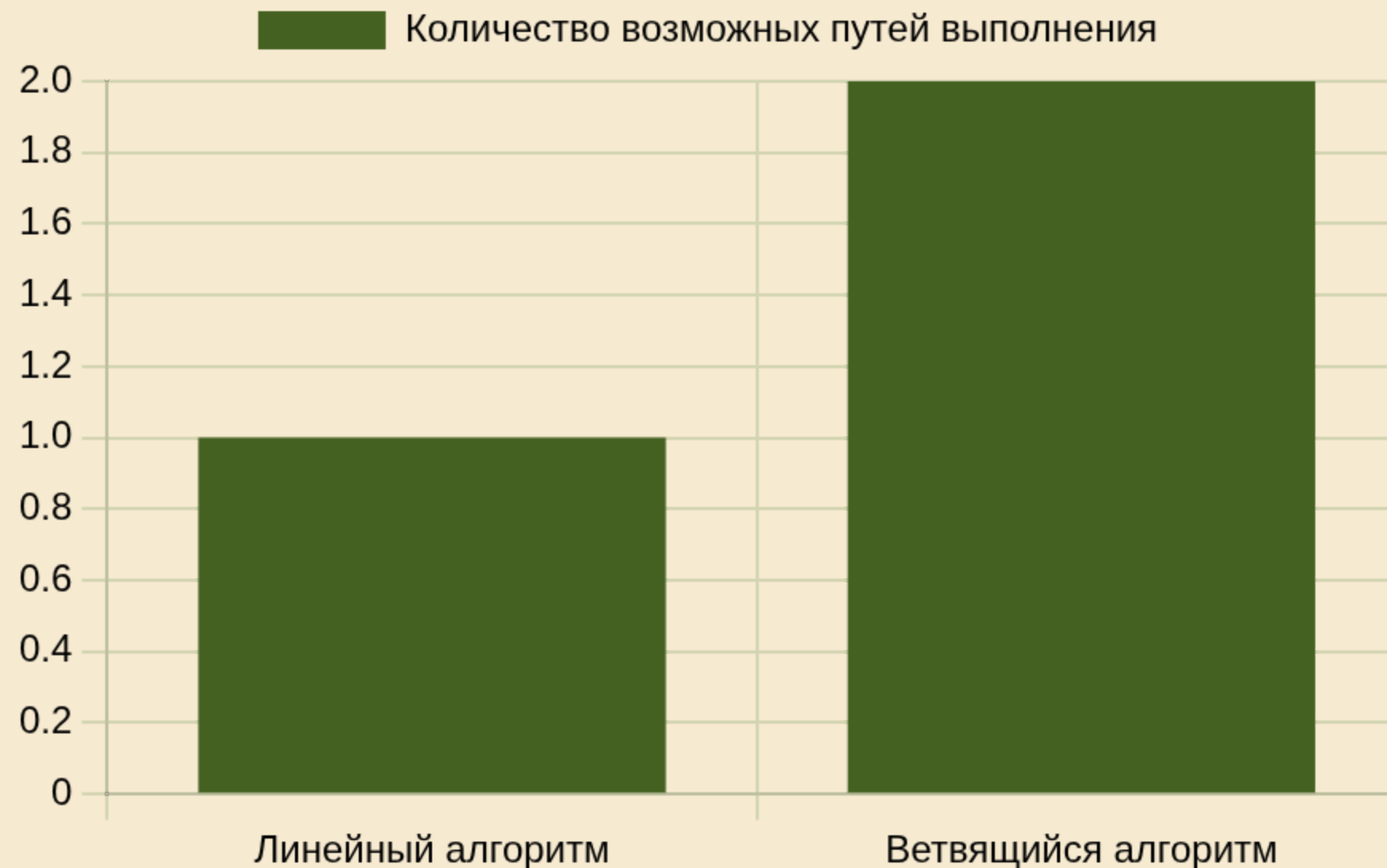
2

Если число чётное, нужно разделить его на 2. Если число нечётное, следует прибавить 1, а затем продолжить выполнение по общей схеме, где результат нужно вывести.

3

Другой школьный пример: если файл найден, его открывают, а если файл не найден — выводят сообщение об ошибке. Ветвление делает алгоритм гибким для разных ситуаций.

Сравнение линейного и ветвящегося алгоритмов



На практике увеличение числа путей означает больше проверок условий и ветвлений. Поэтому ветвящиеся алгоритмы обычно требуют внимательнее составлять команды и проверять логику.

Линейные алгоритмы проще и быстрее анализируются, а ветвящиеся лучше подходят задачам с выбором. Важно заранее предусмотреть каждую ветку, чтобы не потерять вариант условий.

Сравнение на основе структуры алгоритмов (учебные примеры).

Типичные ошибки при составлении алгоритмов

- 1 Пропуск важного шага делает алгоритм неполным: результат может быть неверным или не появиться. Например, если не указать ввод данных, исполнитель не сможет выполнить дальнейшие команды.
- 2 Неясная формулировка команды приводит к разным трактовкам. Фраза «сделай правильно» не годится: лучше писать конкретное действие и объект, чтобы исполнитель выполнял команды одинаково.
- 3 Отсутствие условия завершения часто вызывает зацикливание в программах. Если нет правила выхода, выполнение может продолжаться бесконечно, и задача фактически не решается.
- 4 Неправильный порядок действий и отсутствие проверки результата ухудшают качество решения. Исправляют это уточнением шагов, последовательной нумерацией и добавлением контроля итогового ответа.



Итоги темы: что важно запомнить

Алгоритм — это точная, конечная и понятная исполнителю последовательность действий для решения задачи. У алгоритма ключевые свойства: дискретность, определённость, конечность, результативность и массовость.