

Способы описания алгоритма и блок-схем по ГОСТ

Откройте для себя мощь алгоритмов и стандартов, которые делают их понятными и эффективными. В этой презентации мы рассмотрим ключевые методы описания алгоритмов и углубимся в требования ГОСТ по блок-схемам.

Четыре способа описания алгоритма

1

Словесно-формульное

Подробное текстовое описание с включением математических формул для точной спецификации действий. Идеально для концептуального изложения.

2

Графическое

Визуальное представление логики алгоритма с помощью блок-схем. Обеспечивает наглядность потока управления и данных.

3

Псевдокод

Упрощенный, неформальный язык, имитирующий структуру кода. Легко читаем человеком и является связующим звеном перед программированием.

4

Программное

Непосредственная реализация алгоритма на выбранном языке программирования, готовая к исполнению компьютером.

Почему важно использовать разные способы?

Идея и логика

Словесное описание помогает глубоко понять основную концепцию и логику алгоритма, его предназначение и ожидаемый результат.

Мост к коду

Псевдокод служит эффективным мостом между абстрактной идеей и конкретным программным кодом, помогая в переходе от замысла к реализации.

Восприятие

Графика значительно упрощает восприятие сложных структур, таких как ветвления и циклы, делая их интуитивно понятными.

Конечный результат

Программный код является конечным продуктом, который может быть выполнен компьютером, воплощая алгоритм в жизнь.

Основные символы блок-схем по ГОСТ

Терминатор

Овал: указывает на начало (Старт) и конец (Конец) алгоритма.

Ввод/Вывод

Параллелограмм: используется для обозначения операций ввода (получения) и вывода (отображения) данных.

Операция

Прямоугольник: обозначает выполнение какой-либо операции, обработки данных или присваивания значения.

Ветвление

Ромб: символ принятия решения (условия). Из него выходят две или более линии с подписями («да»/«нет», «истина»/«ложь»).

Подпрограмма

Прямоугольник с двойными вертикальными линиями: указывает на вызов predetermined или внешней процедуры/подпрограммы.

Соединитель

Круг: используется для соединения различных частей блок-схемы на одной или разных страницах.

Правила построения линий и потоков управления

Направление потока

Поток управления обозначается **сплошными линиями со стрелками**. Эти стрелки обязательно показывают направление выполнения алгоритма.

Расположение линий

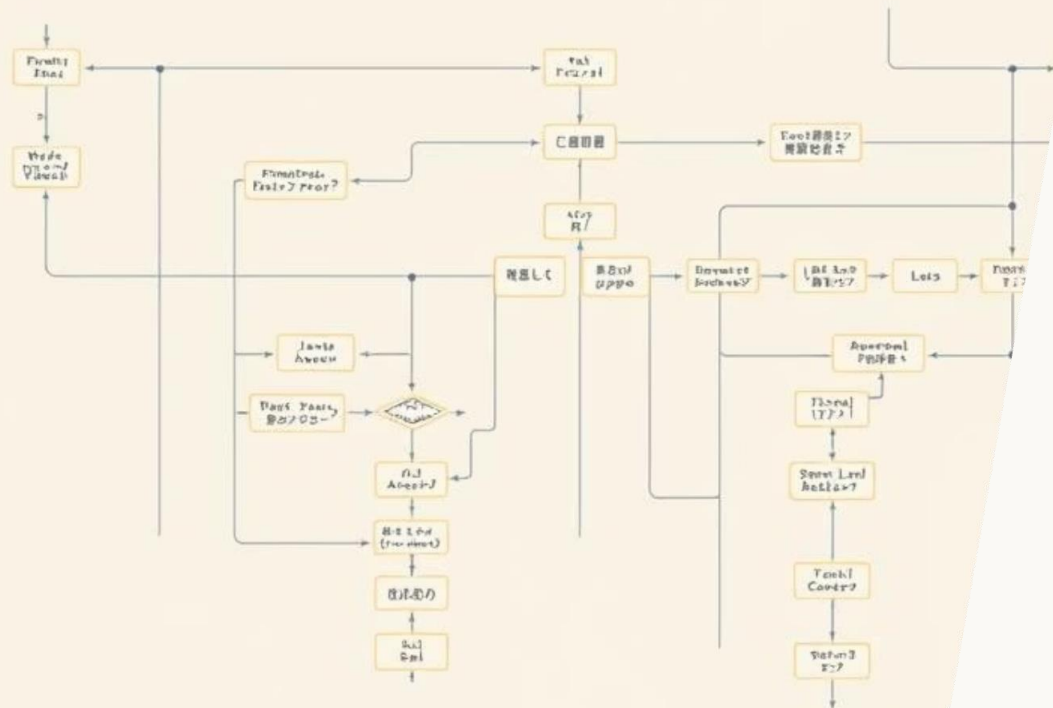
Линии потока управления обычно **подходят к символу слева или сверху, а исходят справа или снизу**. Это обеспечивает читаемость и логичность схемы.

Комментарии

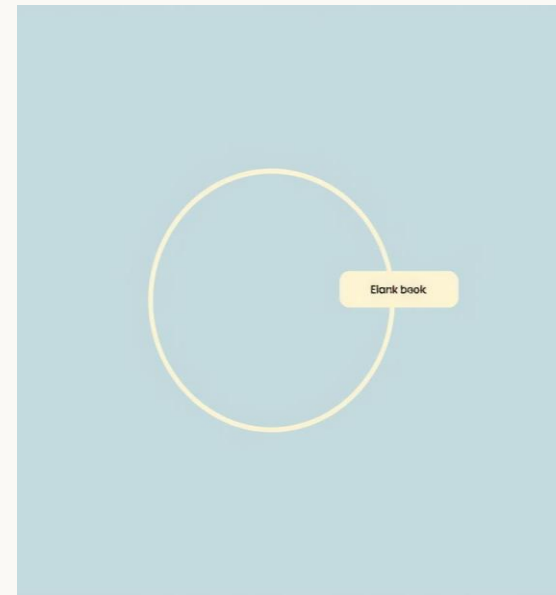
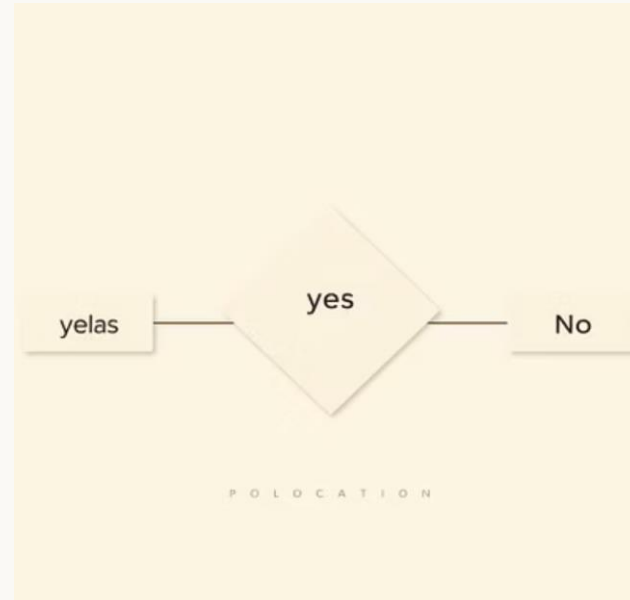
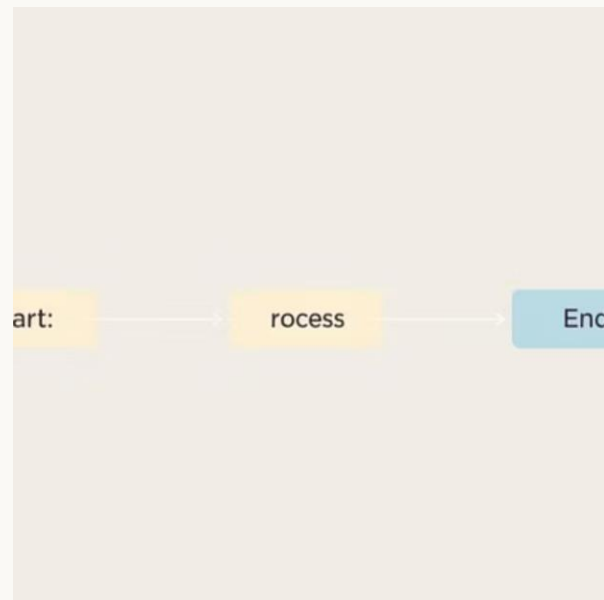
Пунктирные линии используются для связи блоков с **дополнительными комментариями**, не влияющими на ход выполнения алгоритма.

Выходы ветвлений

Ветвления могут иметь **несколько выходов**, каждый из которых подписывается **соответствующим условием** для ясности логики принятия решений.



Виды блок-схем алгоритмов



Линейные

Действия выполняются строго **последовательно**, одно за другим, без условий и повторений.

Разветвляющиеся

Содержат **условия**, которые определяют дальнейший путь выполнения алгоритма.

Циклические

Предполагают **повторное выполнение** одного или нескольких действий до выполнения определённого условия.

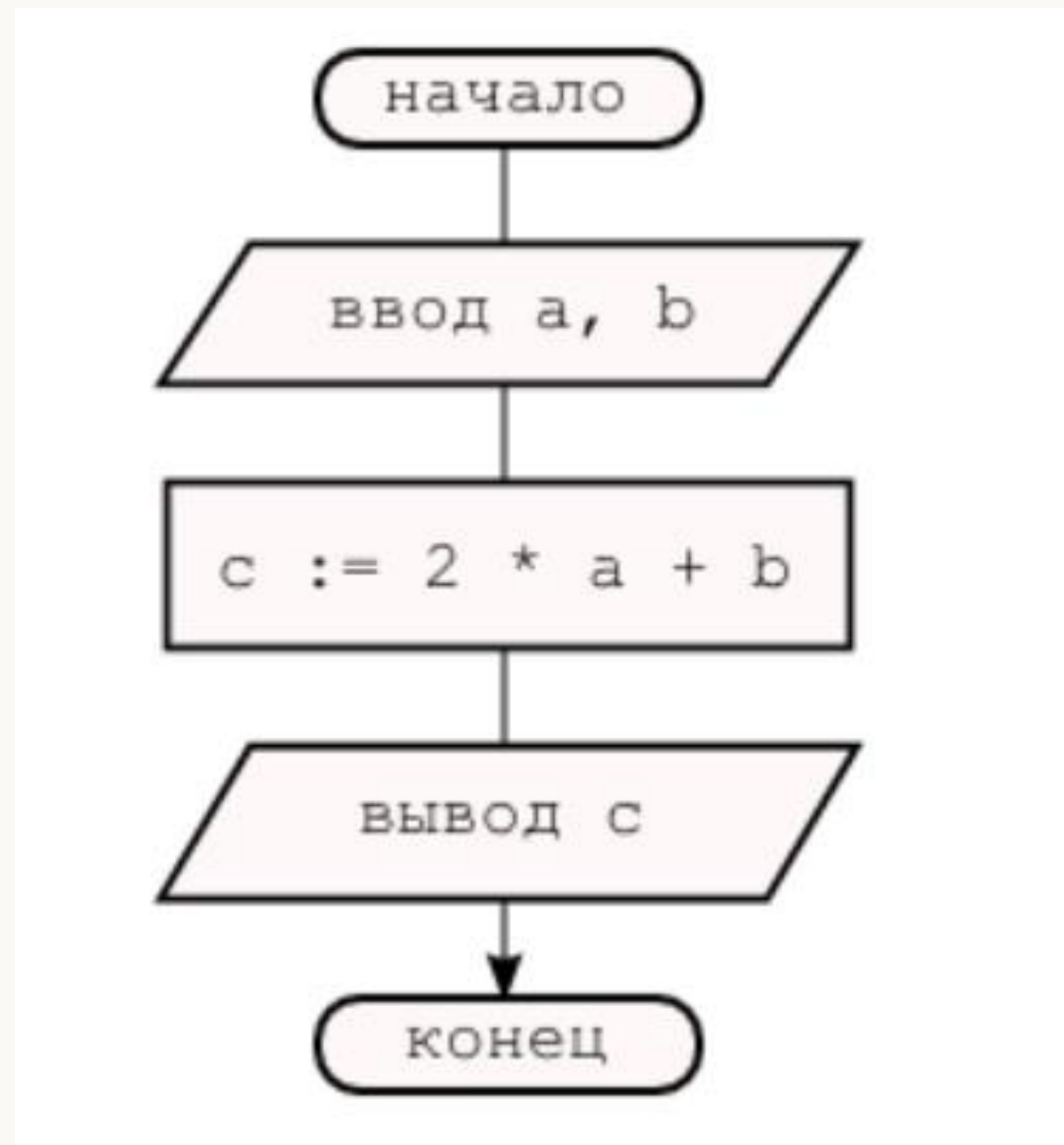
Сложные

Комбинируют **все предыдущие виды** и могут включать вызовы процедур и функций.

Виды блок-схем алгоритмов

Линейные

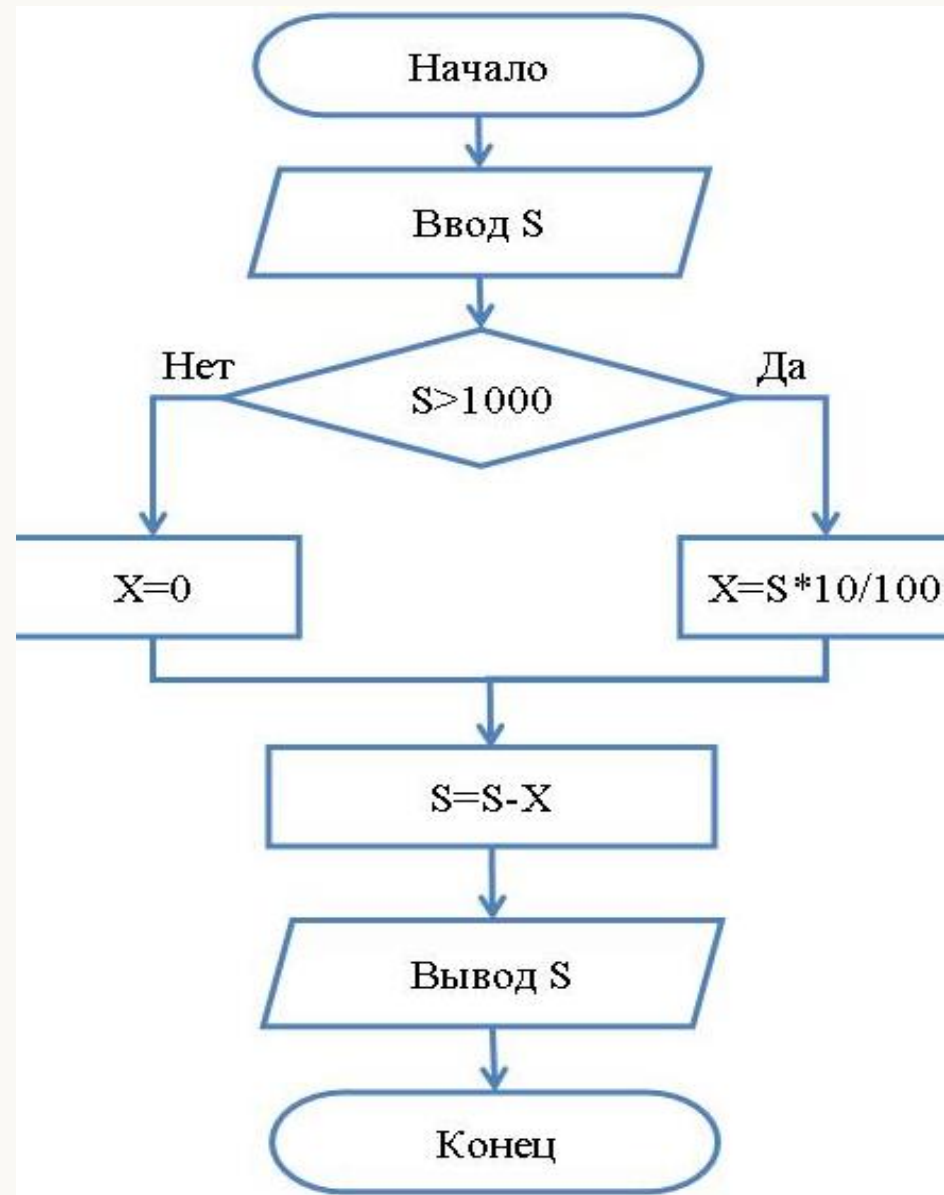
Действия выполняются строго **последовательно**, одно за другим, без условий и повторений.



Виды блок-схем алгоритмов

Разветвляющиеся

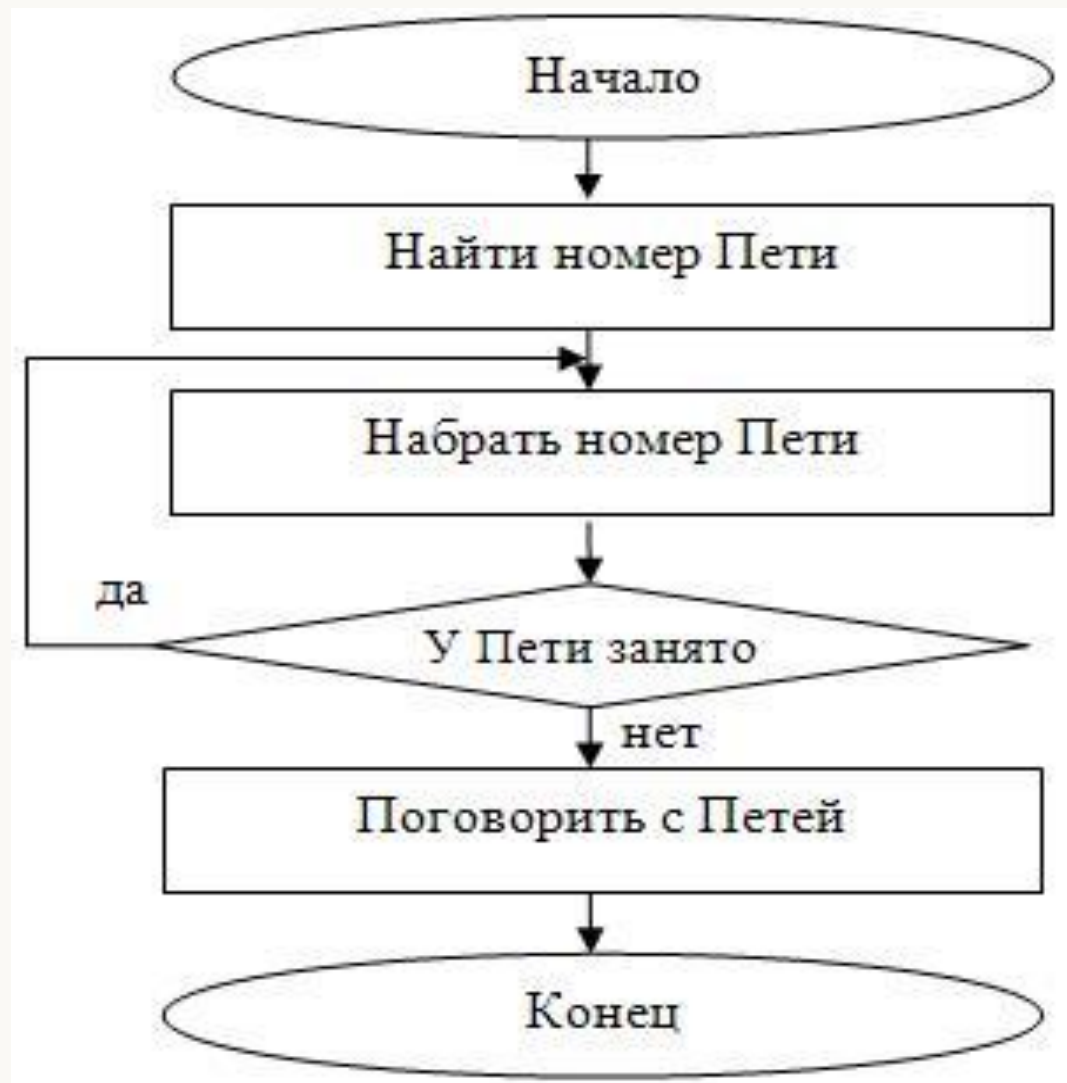
Содержат **условия**, которые определяют дальнейший путь выполнения алгоритма.



Виды блок-схем алгоритмов

Циклические

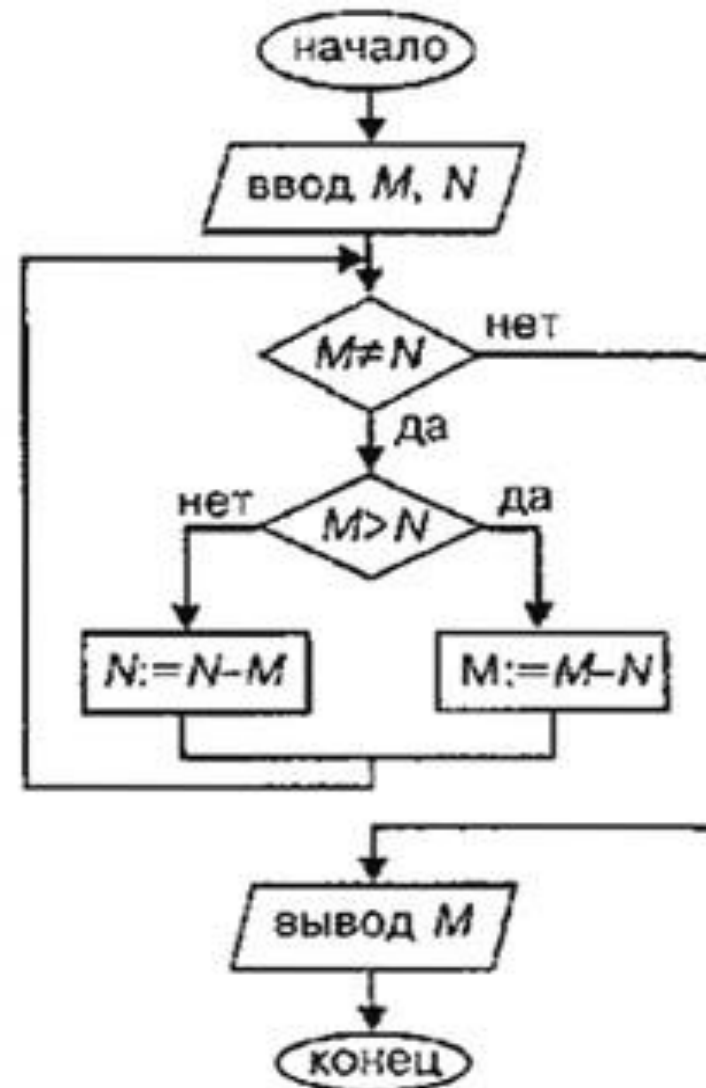
Предполагают **повторное выполнение** одного или нескольких действий до выполнения определённого условия.



Виды блок-схем алгоритмов

Сложные



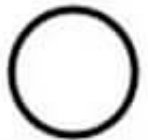
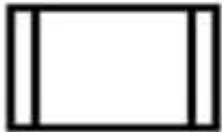
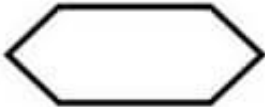
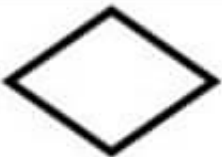

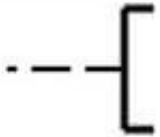
Комбинируют **все предыдущие виды** и могут включать вызовы процедур и функций.



Интерактив: распознай символы и назначь им функции

Перед вами изображения основных символов блок-схем из ГОСТ. Ваша задача — правильно определить их назначение.

- Что обозначает каждый из этих символов и какое действие он представляет в алгоритме?
- Какой поток управления (например, ввод данных, решение) он задаёт?

Символ	Название	Назначение
	Данные	Общее обозначение ввода или вывода данных
	Процесс	Обработка данных, операция или группа операций
	Соединитель	Соединение прерванных линий потока
	Предопределенный процесс	Вычисления по подпрограмме (модулю)
	Подготовка	Осуществляет задание изменений параметров цикла
	Решение	Проверка условия
	Терминатор	Вход или выход во внешнюю среду
	Комментарий	Для записи пояснений к алгоритму

Практическое задание: опиши алгоритм тремя способами

Словесно-формульное описание

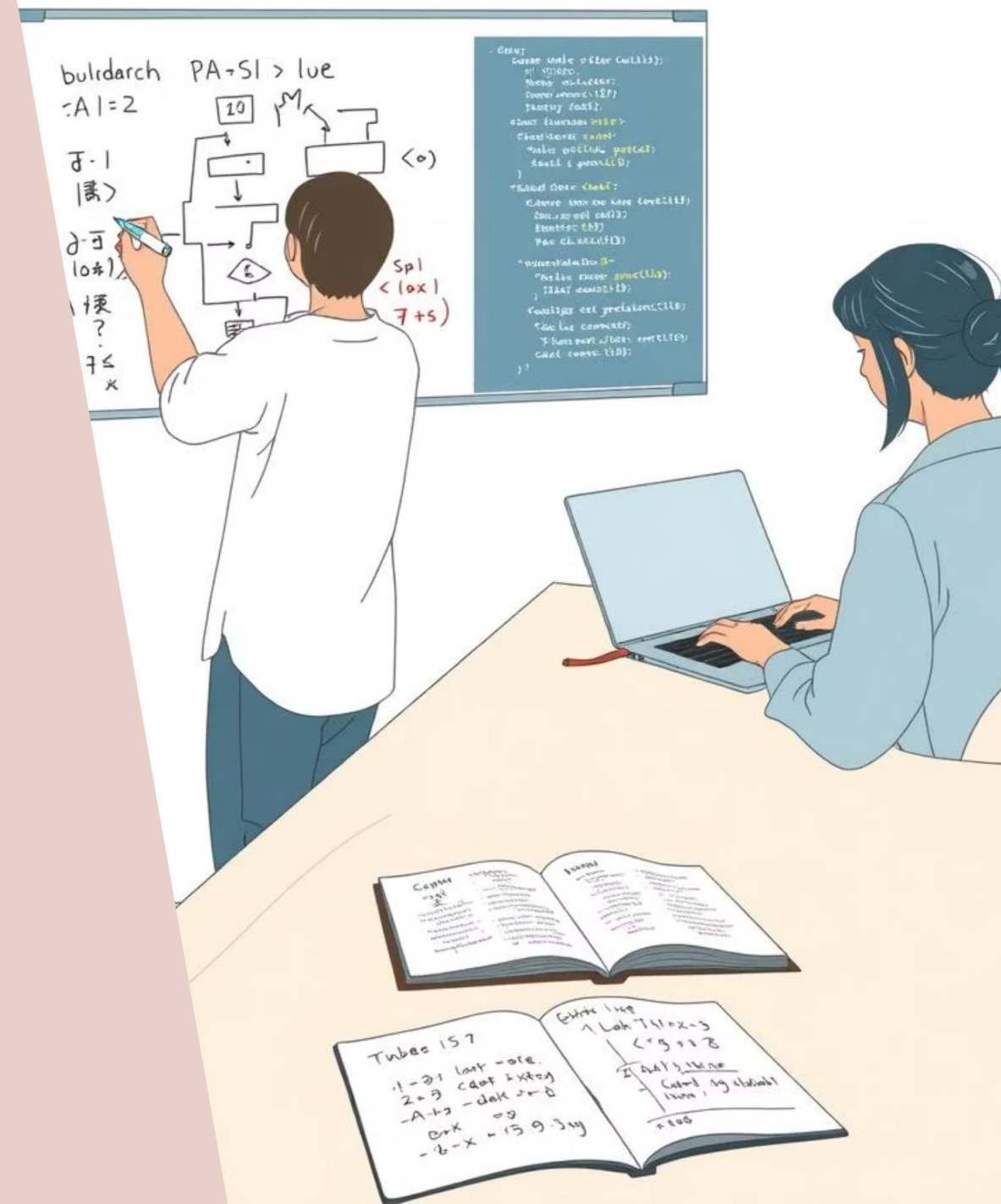
Опишите алгоритм вычисления суммы чисел от 1 до N (например, $1+2+3+\dots+N$) с использованием текстового описания и математической формулы ($S = N * (N + 1) / 2$).

Псевдокод

Напишите псевдокод для того же алгоритма, включая инициализацию переменных, цикл и операцию суммирования.

Графическое представление

Постройте блок-схему этого алгоритма в соответствии с требованиями ГОСТ, используя подходящие символы для начала/конца, ввода/вывода, цикла и операции.



Словесно-формульное описание

Алгоритм использует знаменитую формулу, приписываемую юному Карлу Фридриху Гауссу. Вместо того чтобы складывать числа последовательно (что для больших N может быть долго), мы используем быстрый математический прием.

Шаги алгоритма:

Получить входные данные: Взять целое положительное число N .

Применить формулу:

Умножить число N на число, следующее за ним ($N + 1$).

Разделить полученный результат на 2.

Вернуть результат: Полученное число и будет являться искомой суммой (S).

Псевдокод

ФУНКЦИЯ вычислить_сумму(N)

ЕСЛИ $N < 1$ ТОГДА

Вернуть 0 // Обработка невалидного ввода

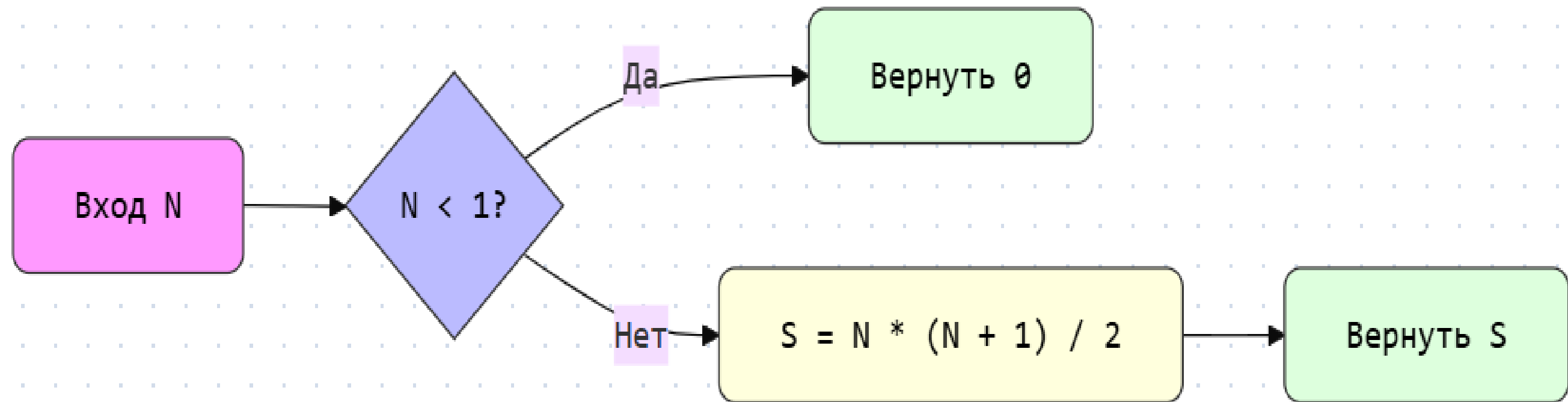
КОНЕЦ ЕСЛИ

$S := (N * (N + 1)) / 2$

Вернуть S

КОНЕЦ ФУНКЦИИ

Графическое представление





Итог: зачем знать ГОСТ и способы описания алгоритмов?

Унификация

Унификация и стандартизация — ключ к эффективному пониманию и обмену знаниями между специалистами.



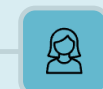
Дополнение

Разные способы описания дополняют друг друга, значительно повышая качество и надёжность разработки.



Читаемость

ГОСТ гарантирует читаемость и правильность блок-схем, минимизируя ошибки и недопонимания.



Закрепление

Используйте интерактивы и практику для глубокого закрепления полученных навыков!