

**5982**

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**им. В.Ф. УТКИНА**

# **СТРУКТУРЫ И АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ**

Методические указания к курсовому проектированию

Рязань 2021

УДК 681.3

Структуры и алгоритмы обработки данных: методические указания к курсовому проектированию / Рязан. гос. радиотехн. ун-т; сост. С.В. Скворцов, В.И. Хрюкин. Рязань, 2021. 16 с.

Содержат методические рекомендации по выполнению курсового проекта по дисциплине «Структуры и алгоритмы обработки данных» в рамках тематики, связанной с разработкой и программной реализацией алгоритмов решения прикладных задач с использованием ЭВМ, включая основные требования к описанию алгоритмов и оформлению программных документов в соответствии со стандартами ЕСПД.

Предназначены для студентов, обучающихся по направлению подготовки 09.03.01 «Информатика и вычислительная техника». Могут быть полезны студентам других направлений и специальностей при выполнении курсового и дипломного проектирования, связанного с разработкой программных продуктов.

Табл. 4. Ил. 9. Библиогр.: 4 назв.

*Схема алгоритма, схема программы, псевдокод, структурограмма, описание применения, руководство системного программиста, руководство программиста, руководство оператора*

Печатается по решению редакционно-издательского совета Рязанского государственного радиотехнического университета.

Рецензент: кафедра САПР вычислительных средств Рязанского государственного радиотехнического университета (зав. кафедрой д-р техн. наук, проф. В.П. Корячко).

## Структуры и алгоритмы обработки данных

Составители: С к в о р ц о в Сергей Владимирович

Х р ю к и н Владимир Иванович

Редактор М.Е. Цветкова

Корректор С.В. Макушина

Подписано в печать 30.03.21. Формат бумаги 60x84 1/16.

Бумага писчая. Печать трафаретная. Усл. печ. л. 1,0.

Тираж 50 экз. Заказ

Рязанский государственный радиотехнический университет.

390005, Рязань, ул. Гагарина, 59/1.

Редакционно-издательский центр РГРТУ.

## 1. ОБЩИЕ ПОЛОЖЕНИЯ

Целями курсового проекта являются: закрепление и углубление теоретических знаний студентов в области структур данных и алгоритмов их обработки; разработка и программная реализация алгоритмов решения прикладных задач обработки данных; получение практических навыков оформления технической документации в соответствии с требованиями стандартов ЕСПД.

Темы курсового проектирования охватывают разнообразные прикладные задачи обработки числовых и текстовых данных, включая сортировку и поиск данных, лексический и синтаксический анализ текстов, построение путей на графах, обходы деревьев, оптимизацию, хеширование, эволюционные вычисления и др.

Работа над курсовым проектом состоит из трех этапов: 1) анализ и формализация решаемой задачи, выбор и обоснование необходимых структур данных, разработка алгоритма; 2) написание текста программы реализации разработанного алгоритма на языке высокого уровня, отладка программы и решение контрольных примеров на ЭВМ; 3) оформление программной документации.

Результаты курсового проектирования оформляются в виде пояснительной записки, которая должна включать: титульный лист; содержание; задание к курсовому проекту; основные разделы пояснительной записки; приложения.

*Титульный лист* является первым листом пояснительной записки. Пример оформления титульного листа приведен в приложении. В *содержании* пояснительной записки помещают заголовки разделов и подразделов с указанием их номеров и расположения, а также ссылки на библиографический список и приложения (требования по оформлению содержания приведены далее). *Задание* к курсовому проекту должно содержать: наименование темы; исходные данные; библиографический список; даты выдачи задания и защиты работы.

Рекомендуется следующая последовательность изложения материала в основной части *пояснительной записки*: введение; постановка задачи; разработка и анализ алгоритма; программная реализация алгоритма; применение программы; заключение; библиографический список. В зависимости от особенностей исходной задачи и программы допускается вводить дополнительные или объединять отдельные разделы/подразделы.

Во введении указываются область применения и практическая постановка задачи. В разделе «Постановка задачи» производится формализация задачи и приводится ее математическое описание. В случае невозможности такого подхода следует представить решение задачи в виде последовательности шагов и привести их словесное описание.

Раздел «Разработка и анализ алгоритма» должен содержать подразделы: используемые данные; описание алгоритма; характеристики алгоритма. В подразделе «Используемые данные» должны быть представлены основные переменные, массивы и другие более сложные структуры данных, используемые в алгоритме. В подразделе «Описание алгоритма» приводятся словесное описание используемого алгоритма, его укрупненная схема или какое-либо другое формальное представление, например псевдокод. Уровень детализации должен быть таким, чтобы различные части алгоритма и взаимосвязь между ними были понятны в целом. Схема алгоритма должна составляться в соответствии с требованиями ЕСПД. В подразделе «Характеристики алгоритма» рассматриваются достоинства, недостатки, ограничения алгоритма, рекомендации по его применению и другие особенности.

Раздел «Программная реализация алгоритма» рекомендуется выполнять в соответствии с требованиями ГОСТ 19.402-78 «Описание программы». Он должен включать следующие подразделы: общие сведения; список используемых идентификаторов; описание логической структуры. В подразделе «Общие сведения» указываются обозначение и наименование программы, системное программное обеспечение, необходимое для функционирования программы, языки программирования, на которых написана программа, а также используемые технические средства (типы ЭВМ и устройств, которые используются при работе программы). В подразделе «Список используемых идентификаторов» указываются характер, формат и описание входных и выходных данных, временных переменных. В подразделе «Описание логической структуры» должны быть указаны: схема (алгоритм) программы; структура программы с описанием функций составных частей и связи между ними; связи программы с другими программами. Описание логической структуры программы выполняют с учетом текста программы на исходном языке и требований ЕСПД к оформлению схем программ.

Раздел «Применение программы» должен содержать подразделы: решение контрольного примера; программная документация. В подразделе «Решение контрольного примера» приводятся результаты выполнения на ЭВМ контрольного примера. Для подтверждения корректности разработанной программы этот пример решается также вручную с использованием заданного алгоритма. Пример следует подбирать так, чтобы он достаточно полно отражал возможности разработанной программы, реализующей заданный алгоритм. В подразделе «Программная документация» приводится один из эксплуатационных документов, устанавливаемый ГОСТ 19.101-77 «Виды программ и программных документов»: описание применения; руководство системного программиста; руководство программиста; руководство оператора.

В *заключении* представляются выводы по курсовому проекту, которые должны содержать анализ полученных результатов и рекомендации по их дальнейшему использованию и развитию. В *библиографическом списке* указываются выходные данные литературных источников, на которые в тексте пояснительной записки имеются ссылки. Оформление библиографического списка должно соответствовать ГОСТ Р 7.0.5-2008. В *приложении* включаются тексты программ и результаты решения тестовых примеров с помощью разработанной программы. Эти примеры следует подбирать так, чтобы они демонстрировали все возможности программы.

## **2. ВЫПОЛНЕНИЕ СХЕМ АЛГОРИТМОВ И ПРОГРАММ**

Схемы алгоритмов и программ должны выполняться в соответствии с требованиями ГОСТ 19.701-90 (ИСО 5807-85). Данный стандарт распространяется на условные обозначения (символы) в схемах алгоритмов, программ, данных и систем и устанавливает правила выполнения схем, используемых для отображения различных видов задач обработки данных и средств их решения [1].

Схемы алгоритмов и программ состоят из имеющих заданное значение символов, краткого пояснительного текста и соединяющих линий.

Символы подразделяются на четыре группы: *символы данных*, указывающие наличие данных и, возможно, вид носителя данных; *символы процесса*, определяющие действия, которые следует выполнить над данными; *символы линий*, указывающие потоки данных или управления между процессами и (или) носителями данных; *специальные символы*, используемые для облегчения написания и чтения схемы.



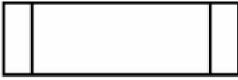
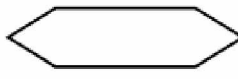

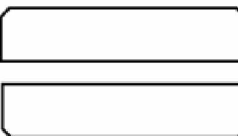
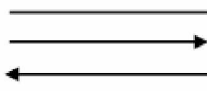

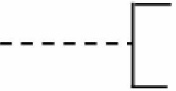
Символы первых трех групп также делятся на основные и специфические. Основные символы применяют тогда, когда точный тип (вид) процесса или носителя данных неизвестен или отсутствует необходимость в описании фактического носителя данных. В противном случае применяется специфический символ.

### **2.1. Описание символов и правила их применения**

Символ предназначен для графической идентификации функции, которую он отображает. Символы в схеме должны быть расположены равномерно и по возможности иметь один размер. Следует придерживаться разумной длины соединений и минимального числа длинных линий. Символы могут быть вычерчены в любой ориентации, но предпочтительной является горизонтальная ориентация.

Символы, наиболее часто используемые в схемах программ и алгоритмов, представлены в табл. 1.

Таблица 1. Основные символы для схем программ и алгоритмов

Название символа	Изображение	Вид символа и назначение
Данные		Символ данных, основной. Ввод и вывод данных
Документ		Символ данных, специфический. Вывод данных в удобочитаемой форме, например, на печатающее устройство
Процесс		Символ процесса, основной. Обработка данных
Предопределенный процесс		Символ процесса, специфический. Вызов подпрограммы
Подготовка (модификация)		Символ процесса, специфический. Цикл с заданным числом повторений
Решение		Символ процесса, специфический. Проверка условия
Граница цикла		Символ процесса, специфический. Начало и конец цикла любого типа
Линия		Символы линий, основные. Поток данных или управления
Терминатор (пуск-останов)		Специальный символ. Начало и конец алгоритма, программы, подпрограммы
Соединитель		Специальный символ. Маркировка разрывов линий
Комментарий		Специальный символ. Пояснения к операциям

Поясним их использование в схемах в соответствии с ГОСТ 19.701-90. Символы данных отображают функции преобразования данных в форму, пригодную для обработки (ввод данных) или отображения

результатов обработки (вывод данных). Основной символ «Данные» отображает данные, носитель которых не определен. Специфический символ «Документ» отображает данные, представленные на носителе в удобочитаемой форме.

*Символы процесса* указывают операции, которые следует выполнить над данными, включая операции определения логических путей в программе или алгоритме. Основной символ «Процесс» отображает функцию обработки данных любого вида (выполнение операции или группы операций, приводящее к изменению значения, формы или размещения информации).

Специфический символ «Предопределенный процесс» отображает предопределенный процесс, который состоит из одной или нескольких операций (шагов) программы, определенных в другом месте (в подпрограмме, модуле). Специфический символ «Подготовка» отображает модификацию команды или группы команд с целью воздействия на некоторую последующую функцию (установка переключателя, модификация индексного регистра или инициализация программы). Специфический символ «Решение» отображает решение или функцию переключательного типа, имеющую один вход и ряд альтернативных выходов, один и только один из которых может быть активизирован после вычисления условий, определенных внутри этого символа. Соответствующие результаты вычисления могут быть записаны по соседству с линиями, отображающими эти пути. Специфический символ «Граница цикла», состоящий из двух частей, отображает начало и конец цикла. Обе части символа имеют один и тот же идентификатор. Условия для инициализации, приращения, завершения и т. д. помещаются внутри символа, в начале или в конце в зависимости от расположения операции, проверяющей условие (для циклов с предусловием и постусловием).

*Символы линий.* Основной символ «Линия» отображает поток данных или управления. При необходимости или для повышения удобочитаемости могут быть добавлены стрелки-указатели. Специфический символ «Пунктирная линия» отображает альтернативную связь между двумя или более символами, также используется для отображения комментария.

*Специфические символы.* Символ «Соединитель» отображает выход в часть схемы и вход из другой части этой схемы и используется для обрыва линии и продолжения ее в другом месте. Соответствующие символы-соединители должны содержать одно и то же уникальное обозначение. Символ «Терминатор» отображает выход во внешнюю среду и вход из внешней среды (начало или конец схемы алгоритма или программы). Символ «Комментарий» используется для добавления описательных комментариев или пояснительных записей в целях объяснения или примечаний.

## 2.2. Правила выполнения схем

Схемы алгоритмов и программ, данных и систем состоят из имеющих заданное значение символов, краткого пояснительного текста и соединяющих линий. Схемы могут использоваться на различных уровнях детализации, причем число уровней зависит от размеров и сложности задачи обработки данных. Уровень детализации должен быть таким, чтобы различные части и взаимосвязь между ними были понятны в целом.

Потоки данных или потоки управления в схемах показываются линиями. Направление потока слева направо и сверху вниз считается стандартным. Для удобства чтения схем допускается использование стрелок на линиях, причем если поток имеет направление, отличное от стандартного, стрелки должны указывать это направление. Линии в схемах должны подходить к символу либо слева, либо сверху, а исходить - либо справа, либо снизу и должны быть направлены к центру символа.

В схемах следует избегать пересечения линий (рис. 1). Пересекающиеся линии не имеют логической связи между собой, поэтому изменения направления в точках пересечения не допускаются. Две или более входящие линии могут объединяться в одну исходящую линию. В этом случае место объединения должно быть смещено (рис. 2).

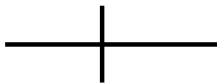


Рис. 1. Пересечение линий потока

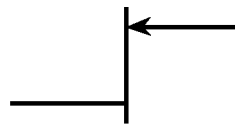


Рис. 2. Объединение линий потока

Когда схема состоит из нескольких страниц, а также при необходимости избежать лишних пересечений или большой длины линий потока, выполняют их разрыв с помощью символов «Соединитель». Соединитель в начале разрыва называется внешним соединителем, а соединитель в конце разрыва - внутренним соединителем. Ссылки на страницы могут быть приведены совместно с символом комментария для их соединителей (рис. 3).

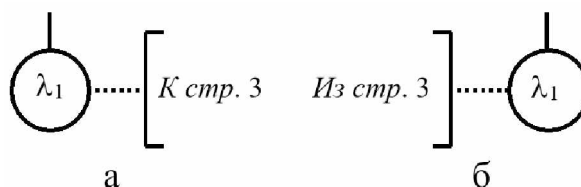


Рис. 3. Разрыв линий потока:

а) внешний соединитель; б) внутренний соединитель

В схемах могут использоваться идентификаторы символов, которые определяют конкретные символы для использования в справочных целях в



других элементах документации, например, в листинге программы или текстовом описании алгоритма. Обычно идентификатором символа служит его порядковый номер в схеме. Идентификатор должен располагаться слева над символом (рис. 4).

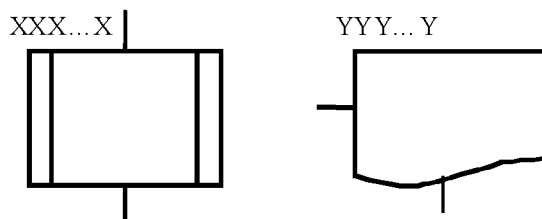


Рис. 4. Примеры размещения идентификатора символа

Некоторые части схем могут представляться укрупненно. Для условного обозначения каждой такой части схемы используется символ с полосой, который указывает, что в этом же комплекте документации в другом месте имеется более подробное представление этого символа (рис. 5).

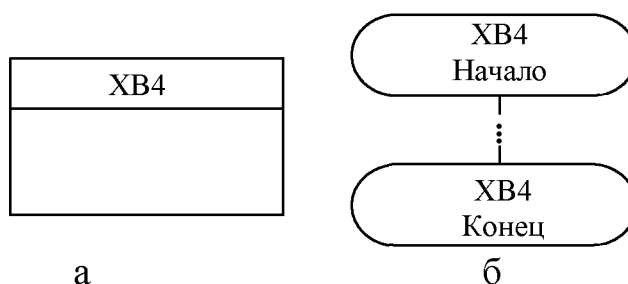


Рис. 5. Подробное представление в схемах:  
а) символ с полосой; б) представление символа с полосой

Символ с полосой представляет собой любой символ, внутри которого в верхней части проведена горизонтальная линия. Между этой линией и верхней линией символа помещен идентификатор, указывающий на подробное представление данного символа. В качестве первого и последнего символов подробного представления используется символ «Терминатор». Первый и последний символы должны содержать ссылку, которая имеется в символе с полосой.

Некоторые символы в схемах могут иметь несколько выходов, которые показывают множеством линий от данного символа к другим символам или одной линией от данного символа, которая затем разветвляется в соответствующее число линий (рис. 6).

Каждый выход из символа должен сопровождаться соответствующими значениями условий, показывающими логический путь, который он представляет (чтобы эти условия и соответствующие ссылки были идентифицированы).

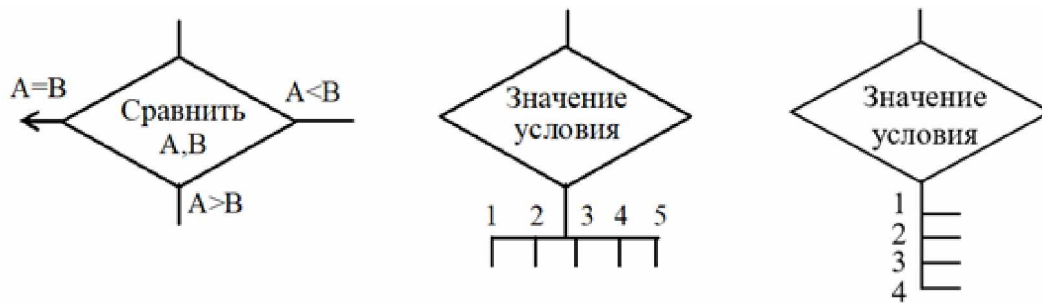


Рис. 6. Примеры использования символа «Решение» при наличии трех и более выходов

Необходимые пояснения к элементам схемы выполняются с использованием символов «Комментарий». Пунктирные линии в символе комментариев показывают связь с соответствующим символом или могут объединять группу символов. Текст комментариев или примечаний должен быть помещен около ограничивающей фигуры (рис. 7).

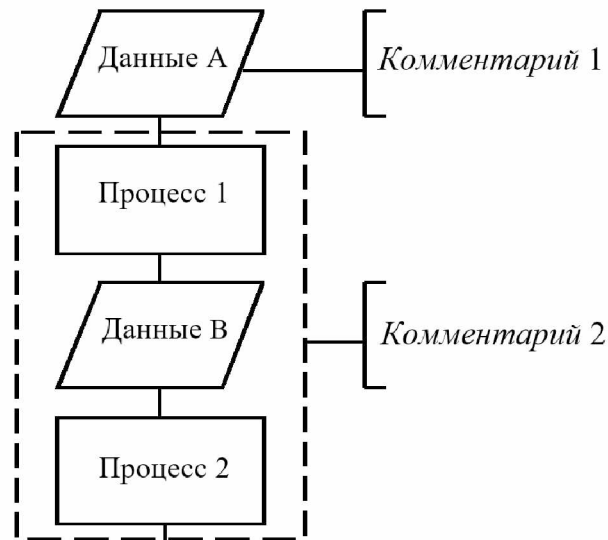


Рис. 7. Примеры использования символа «Комментарий»

### 3. СПОСОБЫ ОПИСАНИЯ АЛГОРИТМОВ

На практике применяют различные формы представления алгоритмов. Выбор формы определяется необходимой степенью детализации, поставленными целями, используемыми методами и техническими средствами решения задачи. Наиболее распространенными являются следующие способы: словесный или формульно-словесный, например описание по шагам; схемы алгоритмов; псевдокод; структурные диаграммы; языки программирования [2].

Рассмотрим некоторые формы представления на примере простого алгоритма нахождения наибольшего общего делителя двух натуральных чисел. Алгоритм Евклида [3]. Большее из двух чисел заменяется на разность большего и меньшего. Процесс повторяется, пока полученные числа

не равны. Вычисления завершаются, когда числа становятся равными. Это и есть их общий делитель. Примеры работы алгоритма определения наибольшего общего делителя (НОД) для двух пар чисел (225, 125) и (13, 4) показаны в табл. 2 и 3 соответственно [3], где итоговые результаты имеют вид:  $\text{НОД}(225, 125) = 25$ ;  $\text{НОД}(13, 4) = 1$ .

Таблица 2. Вычисление НОД(225, 125)

Номер итерации	Значения переменных	
	$A = 225$	$B = 125$
1	$225 - 125 = 100$	125
2	100	$125 - 100 = 25$
3	$100 - 25 = 75$	25
4	$75 - 25 = 50$	25
5	$50 - 25 = 25$	25

Таблица 3. Вычисление НОД(13, 4)

Номер итерации	Значения переменных	
	$A = 13$	$B = 4$
1	$13 - 4 = 9$	4
2	$9 - 4 = 5$	4
3	$5 - 4 = 1$	4
4	1	$4 - 1 = 3$
5	1	$3 - 1 = 2$
6	1	$2 - 1 = 1$

### 3.1. Текстовая пошаговая форма и схема алгоритма

Словесное описание алгоритма может быть полезным на начальных этапах его разработки и детализации. Такая форма является наименее формализованной, но не обеспечивает необходимую наглядность и удобство чтения.

Вход: натуральные числа  $A, B$ .

Выход: наибольший общий делитель  $A$ .

1. Выполнить ввод значения первого числа  $A$ .
2. Выполнить ввод значения второго числа  $B$ .
3. Если  $A = B$ , то выполнить переход к п. 6.
4. Если  $A > B$ , то присвоить переменной  $A$  значение  $A - B$ , иначе присвоить переменной  $B$  значение  $B - A$ .
5. Выполнить переход к п. 3.
6. Выполнить вывод значения переменной  $A$ .
7. Конец алгоритма.

Схема алгоритма заметно повышает наглядность и удобство чтения алгоритма. Она также обеспечивает определенную строгость формального описания на выбранном уровне детализации. При этом использование унифицированных алгоритмических структур, таких как ветвление, циклы с предусловием и постусловием, выбор и др., позволяет реализовать технологию структурного программирования и тем самым уменьшить число возможных ошибок при написании исходного текста программы на алгоритмическом языке.

На рис. 8 приведена схема алгоритма Евклида, в основе которой лежит итерационный цикл, количество повторений которого заранее не известно [3].

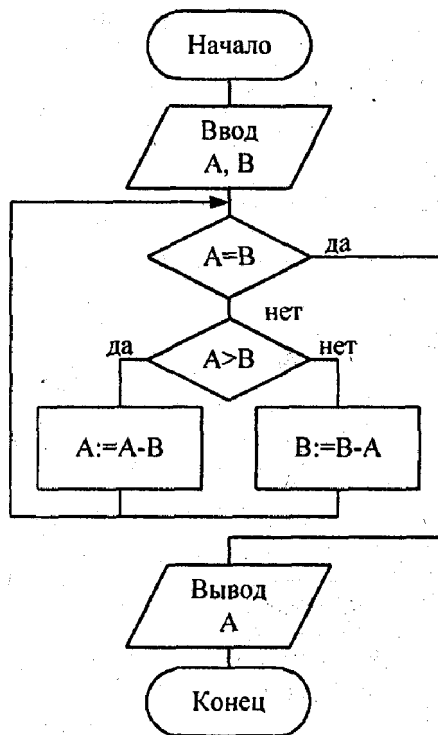


Рис. 8. Схема алгоритма Евклида

Условие выхода из цикла - получение одинаковых чисел  $A$  и  $B$ . Поскольку нельзя исключить, что пользователь введет равные числа, проверка осуществляется на входе в цикл, т.е. используется цикл с предусловием.

Если числа не равны, то на каждой итерации цикла одно из чисел (большее) заменяется разностью большего и меньшего. Для выполнения этой замены необходимо описать оба варианта, для чего применяется ветвление с проверкой, какое из чисел больше. После выхода из цикла выводится результат - любое из двух полученных чисел, так как они равны между собой.

### 3.2. Псевдокод и описание на языке программирования

Псевдокод по своим возможностям занимает промежуточное положение между словесным пошаговым описанием и описанием алгоритма на конкретном языке программирования. При использовании псевдокода обычно вначале кратко описывается его синтаксис, который представляет собой некоторую абстракцию языка программирования. Ключевые слова могут записываться как на русском языке, так и на английском. Далее приводится псевдокод алгоритма Евклида.

```

НОД (А, В)
  ввод (А, В);
  пока (А≠В) повторять
    если (А>В), то А:=А-В,
    иначе В:= В-А;
  конец цикла;
  вывод (А);
конец алгоритма

```

Описание этого же алгоритма в виде функции языка программирования Паскаль имеет следующий вид.

```

Function NOD(a,b: integer):integer;
Begin
While a<>b then
If a>b then a:=a-b
else b:=b-a;
NOD:=a;
end;

```

### 3.3. Структурограмма алгоритма

Структурограмма или структурная диаграмма является еще одним способом графического представления алгоритма. Она также называется диаграммой Нэсси - Шнейдермана или *N-S-диаграммой* по первым буквам фамилий авторов - Nassi - Shneiderman [4].

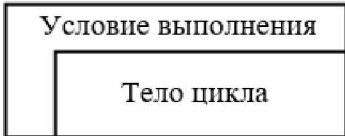
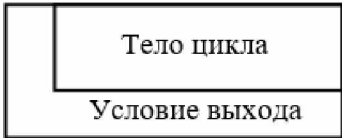
Основная идея структурограмм базируется на принципах структурного программирования. В частности, если оператор `goto` не должен использоваться в программе, то соответственно линии потоков не нужны в графическом представлении алгоритмов.

Структурограммы широко используются в ряде стран. Например, в Германии их применение при документировании программ обусловлено требованиями государственного стандарта этой страны.

Очевидными достоинствами структурограмм являются: наглядность; отсутствие линий со стрелками, что помогает избежать случайных ошибок; компактность, так как даже относительно длинный алгоритм несложно разместить на одной странице; простота использования.

Структурограммы строятся с использованием элементарных блоков (символов), представленных в табл. 4.

Таблица 4. Элементарные блоки структурограммы алгоритма

Название блока	Изображение	Назначение
Обработка		Обработка и ввод-вывод данных
Подпрограмма		Вызов подпрограммы
Следование		Последовательность действий
Решение		Ветвление алгоритма
Выбор		Множественное ветвление
Цикл с предусловием		Повторение, пока условие истинно
Цикл с постусловием		Повторять до тех пор, пока условие выхода ложно

В качестве примера на рис. 9 приведена структурограмма алгоритма Евклида.

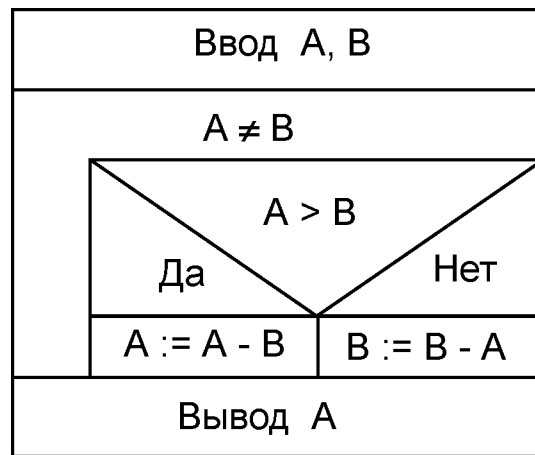


Рис. 9. Структурограмма алгоритма Евклида

## 4. РАЗРАБОТКА ПРОГРАММНЫХ ДОКУМЕНТОВ

### 4.1. Описание применения

Состав и требования к содержанию программного документа «Описание применения» устанавливается ГОСТ 19.502-78. Текст документа должен состоять из следующих разделов: назначение программы; условия применения; описание задачи; входные и выходные данные.

В разделе «Назначение программы» указывают назначение и возможности программы, ее основные характеристики, а также ограничения, накладываемые на область применения.

В разделе «Условия применения» указывают условия, необходимые для выполнения программы (требования к необходимым для данной программы техническим средствам и другим программам, общие характеристики входной и выходной информации, а также требования и условия организационного, технического и технологического характера и т. п.).

В разделе «Описание задачи» должны быть указаны определения (формулировки, постановки) задачи и методы ее решения.

В разделе «Входные и выходные данные» должны быть указаны сведения о входных и выходных данных.

В приложение к общему описанию могут быть включены справочные материалы (иллюстрации, таблицы, графики, примеры и т.п.).

### 4.2. Руководство системного программиста

Требования к содержанию и оформлению программного документа «Руководство системного программиста» устанавливаются ГОСТ

19.503-79. Текст документа должен содержать следующие разделы: общие сведения о программе; структура программы; настройка программы; проверка программы; дополнительные возможности; сообщения системному программисту.

В разделе «Общие сведения о программе» должны быть указаны назначение и функции программы и сведения о технических и программных средствах, обеспечивающих выполнение данной программы.

В разделе «Структура программы» должны быть приведены сведения о структуре программы, ее составных частях, о связях между составными частями и о связях с другими программами.

В разделе «Настройка программы» должно быть приведено описание действий по настройке программы на условия конкретного применения (настройка на состав технических средств, выбор функций и др.). При необходимости приводят поясняющие примеры.

В разделе «Проверка программы» должно быть приведено описание способов проверки, позволяющих дать общее заключение о работоспособности программы (контрольные примеры, методы прогона, результаты).

В разделе «Дополнительные возможности» должно быть приведено описание дополнительных функциональных возможностей программы и способов их выбора.

В разделе «Сообщения системному программисту» должны быть указаны тексты сообщений, выдаваемых в ходе выполнения настройки и проверки программы, а также в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

В приложении могут быть приведены дополнительные материалы (примеры, иллюстрации, таблицы, графики и т.п.).

### **4.3. Руководство программиста**

Требования к содержанию и оформлению программного документа «Руководство программиста» устанавливаются ГОСТ 19.504-79. Текст документа должен содержать следующие разделы: назначение и условия применения программы; характеристики программы; обращение к программе; входные и выходные данные; сообщения.

В разделе «Назначение и условия применения программы» должны быть указаны назначение и функции, выполняемые программой, а также условия, необходимые для выполнения программы (объем оперативной памяти, требования к составу и параметрам периферийных устройств, требования к программному обеспечению и др.).

В разделе «Характеристики программы» должно быть приведено описание основных характеристик и особенностей программы (временные



характеристики, режим работы, средства контроля правильности выполнения и самовосстанавливаемости программы и т. п.).

В разделе «Обращение к программе» должно быть приведено описание процедур вызова программы (способы передачи управления и параметров данных и др.).

В разделе «Входные и выходные данные» должно быть приведено описание организации используемой входной и выходной информации и, при необходимости, ее кодирования.

В разделе «Сообщения» должны быть указаны тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действия, которые необходимо предпринять по этим сообщениям.

В приложении могут быть приведены дополнительные материалы (примеры, иллюстрации, таблицы, графики и т.п.).

#### **4.4. Руководство оператора**

Требования к содержанию и оформлению программного документа «Руководство оператора» устанавливаются ГОСТ 19.505-79. Текст документа должен содержать следующие разделы: назначение программы; условия выполнения программы; выполнение программы; сообщения оператору.

В разделе «Назначение программы» должны быть указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» должны быть указаны условия, необходимые для выполнения программы [минимальный и (или) максимальный состав аппаратурных и программных средств и т.п.].

В разделе «Выполнение программы» должны быть: указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы; приведены описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также реакция программы на эти команды.

В разделе «Сообщения оператору» должны быть приведены тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора (действия в случае сбоя, повторного запуска программы и т. п.).

Содержание указанных разделов допускается иллюстрировать поясняющими примерами, таблицами, схемами, графиками.

## Библиографический список

1. Единая система программной документации. М.: Стандартинформ, 2005. 115 с.
2. Гагарина Л.Г., Колдаев В.Д. Алгоритмы и структуры данных. М.: Финансы и статистика; ИНФРА-М, 2009. 304 с.
3. Иванова Г.С. Основы программирования. М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. 416 с.
4. Nassi I., Shneiderman B. Flowchart Techniques for Structured Programming / SIGPLAN Notices 8, 8 (August, 1973).

## ОГЛАВЛЕНИЕ

1. ОБЩИЕ ПОЛОЖЕНИЯ. . . . .	1
2. ВЫПОЛНЕНИЕ СХЕМ АЛГОРИТМОВ И ПРОГРАММ. . . . .	3
2.1. Описание символов и правила их применения. . . . .	3
2.2. Правила выполнения схем. . . . .	6
3. СПОСОБЫ ОПИСАНИЯ АЛГОРИТМОВ. . . . .	8
3.1. Текстовая пошаговая форма и схема алгоритма. . . . .	9
3.2. Псевдокод и описание на языке программирования. . . . .	10
3.3. Структурограмма алгоритма. . . . .	11
4. РАЗРАБОТКА ПРОГРАММНЫХ ДОКУМЕНТОВ. . . . .	13
4.1. Описание применения. . . . .	13
4.2. Руководство системного программиста. . . . .	13
4.3. Руководство программиста. . . . .	14
4.4. Руководство оператора. . . . .	15