Алгоритмизация и основы программирования

Алгоритмизация

① *Понятие алгоритма*

Алгоритм *— это точное предписание, которое определяет процесс, ведущий от исходных данных к требуемому конечному результату.*

Алгоритмами, например, является умножение в столбик, деление уголком, сложение в столбик, нахождение НОК и НОД, решения алгебраических уравнений, умножения матриц и т.п.
Слово алгоритм происходит от ***algoritmi****, являющегося латинской транслитерацией арабского имени хорезмийского математика IX века* ***аль-Хорезми****. Благодаря латинскому переводу трактата аль-Хорезми европейцы в XII веке познакомились с позиционной системой счисления.*

② *Свойства алгоритма*

- ***Определённость***
  требует от алгоритма быть строгим, чётким, понятным. Все действия, символы операций должны быть или общепринятыми, или заранее чётко и однозначно определены. Не допускается двусмысленности, неоднозначности.

- ***Дискретность***
  алгоритм должен представлять процесс решения задачи как последовательное выполнение простых (или ранее определенных) шагов (этапов).

- ***Результативность***
  предполагает обязательное получение результата («отрицательный результат – тоже результат») за конечное число шагов.

- ***Массовость***
  требует от алгоритма возможности применения его для некоторого класса задач, различающихся лишь исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется ***областью применимости алгоритма****.*

③ *Способы описания алгоритмов*

К основным способам описания алгоритмов можно отнести следующие:

- ***словесно-формульный*** *(запись на естественном языке);*
- ***структурный*** *или* ***блок-схемный*** *(описание алгоритма с помощью графических символов или блоков);*
- ***программный*** *(текст на каком-либо языке программирования).*

Выбор способа записи зависит от характера задачи. Алгоритм вычислительного характера можно записать формулой или последовательностью формул. Алгоритм заваривания чая удобно записать словами в пронумерованных пунктах. Алгоритм решения квадратного уравнения будет наиболее понятие при записи словами и формулами.

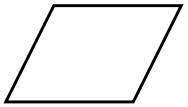
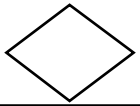
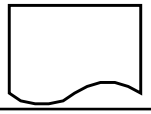
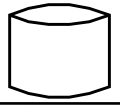
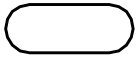
Из формальных способов записи алгоритмов чаще других используют язык блок-схем и языки программирования.

Блок-схемы алгоритмов.

При блок-схемном описании алгоритм изображается геометрическими фигурами (блоками), связанными по управлению линиями со стрелками. В блоках записывается последовательность действий.

Данный способ по сравнению с другими способами записи алгоритма имеет ряд преимуществ. Он наиболее нагляден: каждая операция вычислительного процесса изображается отдельной геометрической фигурой. Кроме того, графическое изображение алгоритма наглядно показывает разветвления путей решения задачи в зависимости от различных условий, повторение отдельных этапов вычислительного процесса и другие детали.

Оформление программ должно соответствовать определенным требованиям. Существует единая система программной документации (ЕСПД), которая устанавливает правила разработки, оформления программ и программной документации. В ЕСПД определены и правила оформления блок-схем алгоритмов (ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД).

Наименование	Обозначение	Функции
Процесс		Выполнение операции или группы операций, в результате которых изменяется значение, форма представления или расположение данных.
Ввод-вывод		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод).
Решение		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий.
Предопределенный процесс		Использование ранее созданных и отдельно написанных программ (подпрограмм).
Документ		Вывод данных на бумажный носитель.
Магнитный диск		Ввод-вывод данных, носителем которых служит магнитный диск.
Пуск-останов		Начало, конец, прерывание процесса обработки данных.
Соединитель		Указание связи между прерванными линиями, соединяющими блоки.
Межстраничный соединитель		Указание связи между прерванными линиями, соединяющими блоки, расположенные на разных листах.
Комментарий		Связь между элементом схемы и пояснением.

④ *Виды алгоритмов*

Алгоритмы обычно состоят из трёх основных базовых структур:
1. следование (линейные алгоритмы);
2. ветвление (ветвящиеся алгоритмы);
3. цикл (циклические алгоритмы).

Доказано, что этих трёх основных базовых структур достаточно, чтобы построить алгоритм любой сложности.

*Самые простые по структуре – **линейные алгоритмы**. Они образуется из последовательности действий, следующих одно за другим, без ветвлений и циклов. В блок-схемах таких алгоритмов отсутствуют блоки «решение» и обратные связи, позволяющие многократно выполнять некоторые действия.*

действие 1
действие 2
...
действие N

Ветвящиеся *алгоритмы содержат блок «решение» и обеспечивают в зависимости от результата проверки условия выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к общему выходу, так что работа алгоритма будет продолжаться независимо от того, какой путь будет выбран.*

Структуру ветвление можно описать четырьмя основными вариантами:

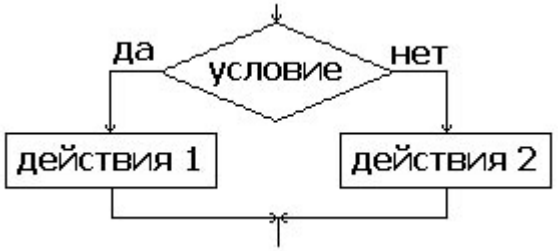
- *если-то;*
- *если-то-иначе;*
- *выбор;*
- *выбор-иначе.*

1. Вариант **если-то**.

если условие
то действия
всё

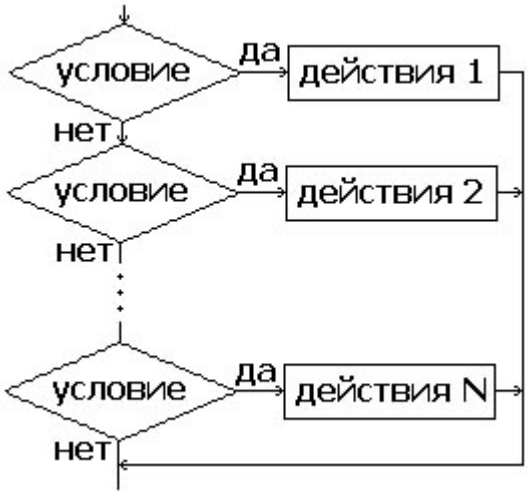
2. Вариант **если-то-иначе**.

если условие
то действия 1
иначе действия 2
всё



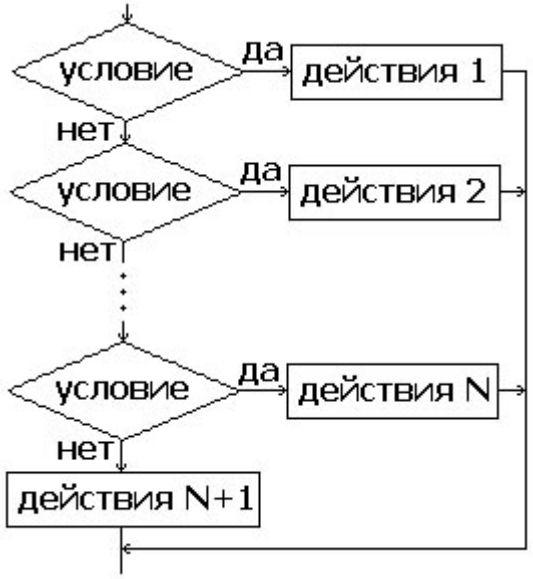
3. Вариант **выбор**.

выбор
при условие 1: действия 1
при условие 2: действия 2
. . . . . .
при условие N: действия N
всё



4. Вариант **выбор-иначе**.

выбор
при условие 1: действия 1
при условие 2: действия 2
. . . . . .
при условие N: действия N
иначе действия N+1
всё



Циклические алгоритмы *от линейных и ветвящихся отличаются наличием в структуре алгоритма* ***обратной связи****, которая позволяет некоторые действия повторять многократно, циклически. Эти действия составляют* ***тело цикла****. Циклические алгоритмы могут так же включать участки, характерные как для линейных, так и ветвящихся алгоритмов.*

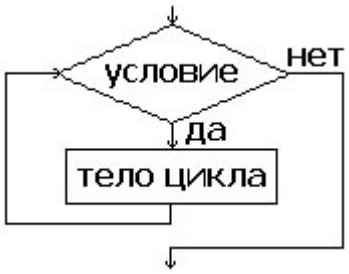
Циклические алгоритмы делятся на алгоритмы:
- с известным количеством повторений (циклы типа «для»);
- с неизвестным заранее количеством повторений (цикла типа «пока»).

В обоих случаях окончание циклического процесса определяется поставленным заранее условием.

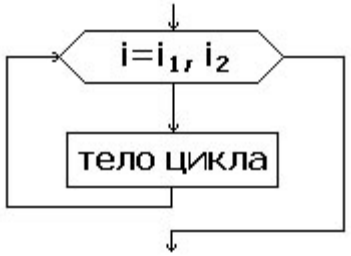
В циклических алгоритмах типа «для» этим условием служит явно заданное количество повторений.

Например: задача определения суммы чётных натуральных чисел от 2 до 100.

В циклических алгоритмах с неизвестным заранее числом повторений (циклы типа «пока») явно число повторений не задано, как например, в задаче с такой постановкой: «Сколько нужно взять, начиная с единицы, последовательно расположенных чисел натурального ряда, чтобы их сумма превысила 1000?»

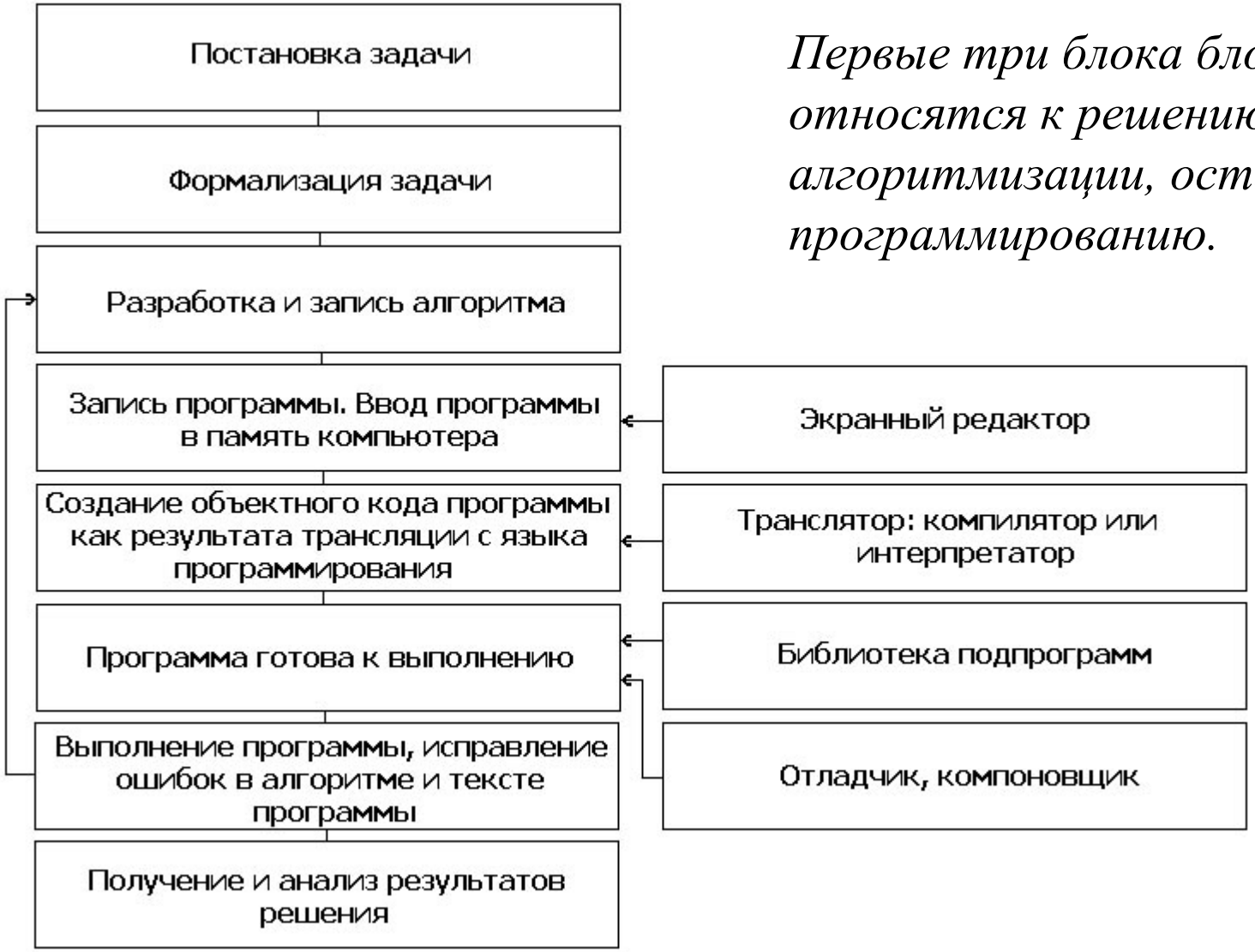


Возможны случаи, когда внутри тела цикла необходимо повторять некоторую последовательность операторов, т.е. организовать внутренний цикл. Такая структура получила название ***вложенных циклов****. Глубина вложения циклов (то есть количество вложенных друг в друга циклов) может быть различной.*



⑤ Этапы решения задач на компьютере.

Вариант обобщённой блок-схемы последовательности действий при решении задач на компьютере.



Первые три блока блок-схемы относятся к решению вопросов алгоритмизации, остальные – к программированию.

Решение задач с помощью компьютера включает в себя следующие основные этапы, часть из которых осуществляется без участия компьютера.

1. Постановка задачи:

- сбор информации о задаче;
- формулировка условия задачи;
- определение конечных целей решения задачи;
- определение формы выдачи результатов;
- описание данных (их типов, диапазонов величин, структуры и т.п.).

2. Анализ и исследование задачи, модели:

- анализ существующих аналогов;
- анализ технических и программных средств;
- разработка математической модели;
- разработка структур данных.

3. Разработка алгоритма:

-выбор метода
- проектирования алгоритма;
- выбор формы записи алгоритма (блок-схемы, псевдокод и др.);
- выбор тестов и метода тестирования;
- проектирование алгоритма.

4. Программирование:

- выбор языка программирования;
- уточнение способов организации данных;
- запись алгоритма на выбранном языке программирования.

5. Тестирование и отладка:

- синтаксическая отладка;

- отладка семантики и логической структуры;

- тестовые расчеты и анализ результатов тестирования;

- совершенствование программы.

6. Анализ результатов решения задачи и уточнение в случае необходимости математической модели с повторным выполнением этапов 2 - 5.

7. Сопровождение программы:

- доработка программы для решения конкретных задач;

- составление документации к решенной задаче, к математической модели, к алгоритму, к программе, к набору тестов, к использованию.