Информатика

Лекция 2

Решение задач на ЭВМ

Решение задач на ЭВМ

- Этапы разработки программы
  1. Содержательная постановка задачи
  2. Формализованная постановка (математическая модель)
  3. Алгоритмизация
  4. Разработка структур данных
  5. Программирование и отладка
  6. Испытания программы
  7. Документирование программы

Содержательная постановка

- Формулировка задачи специалистом предметной области
  - Необходимые исходные данные
  - Что должно быть получено в результате
  - Какие ограничения наложены на область применения программы
  - Дополнительные требования по быстродействию, памяти и др.
- Контрольные примеры, отражающие сущность задачи

Формализованная постановка

- Описание задачи и метода ее решения с помощью соответствующего математического аппарата
  - Определить и описать математическую форму представления исходных данных и конечных результатов
  - Сформулировать метод решения (необходимые преобразования, численные методы, правила получения результатов по исходным данным)

Алгоритмизация

- Формальное описание вычислительного процесса, направленного на получение результатов из произвольных исходных данных
  - Преобразование формул к виду, удобному для алгоритмизации
  - Проектирование схемы алгоритма с пошаговой детализацией процесса вычислений (ГОСТ 19.701-90)
  - Обработка исключительных ситуаций

Разработка структуры данных

- Определение структур, типов и имен для объектов программы
  - (учет диапазонов значений и требуемой точности)
- Формирование физической и логической структур данных
- Разработка физической структуры внешнего представления данных
  - (выбор форматов и расположения данных на конкретных носителях)

Программирование и отладка

- Кодирование
  - Запись на языке программирования с комментариями и стилем оформления текста программы
  - Соблюдение правил структурного программирования
- Автономная отладка модулей, комплексирование и отладка в целом
  - Определение промежуточных результатов и узловых точек программы

Испытания программы

- Проверка соответствия техническому заданию
  - Разработка тестов (контрольных примеров для всех блоков программы)
  - Проверка всех вариантов ввода-вывода
  - Выполнение всех ветвей алгоритма
- Внесение изменений в программу и соответствующую документацию

Документирование программы

Вид программного документа	Содержание программного документа по ГОСТ 19.101-77
Ведомость эксплуатационных документов	Перечень эксплуатационных документов на программу
Формуляр	Основные характеристики программы, комплектность и сведения об эксплуатации программы
Описание применения	Сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств
Руководство системного программиста	Сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения
Руководство программиста	Сведения для эксплуатации программы
Руководство оператора	Сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы
Описание языка	Описание синтаксиса и семантики языка
Руководство по техническому обслуживанию	Сведения для применения тестовых и диагностических программ при обслуживании технических средств

Отображение

- Закон по которому каждому элементу некоторого заданного множества X ставится в соответствие вполне определенный элемент другого заданного множества Y

$x \in X, \quad y \in Y, \; y = f(x)$ либо $f: X \to Y$

(множество X - область определения отображения, а множество Y - множество значений отображения)

Понятие отображения совпадает с понятиями функция, оператор, преобразование

Функция

- Величина **y** называется функцией переменной величины **x**, если каждому из тех значений, которые может принимать **x** соответствует одно или несколько определенных значений **y**

(совокупность всех значений, которые может принимать аргумент **x** функции **f(x)** называется областью определения функции)

Способы задания функции: табличный, графический, аналитический

Алгоритмизация

Алгоритм

- Точное предписание, задающее вычислительный процесс, начинающийся с произвольного набора исходных данных и направленный на получение полностью определяемых ими результатов
- Абстрактный вычислительный алгоритм применяется к математическим объектам, записывается в математических терминах и не связан с конкретным языком программирования

Свойства алгоритма

- Конечность (финитность)
  - Всегда заканчивается после конечного числа шагов
- Определенность (детерминированность)
  - Однозначность и недвусмысленность выполняемых действий на каждом шаге
- Результативность (направленность)
  - Получение результата, в том числе в точках неопределенности с выдачей соответствующей информации
- Массовость
  - Алгоритм служит для решения целого класса задач

Способы задания алгоритма

- Словесный
- Алгоритмические системы
- Языки программирования
- Языки схем алгоритмов
- Схемы алгоритмов Янова, Ляпунова, операторные схемы Лаврова и проч.

Способы задания алгоритма

Алгоритм Евклида

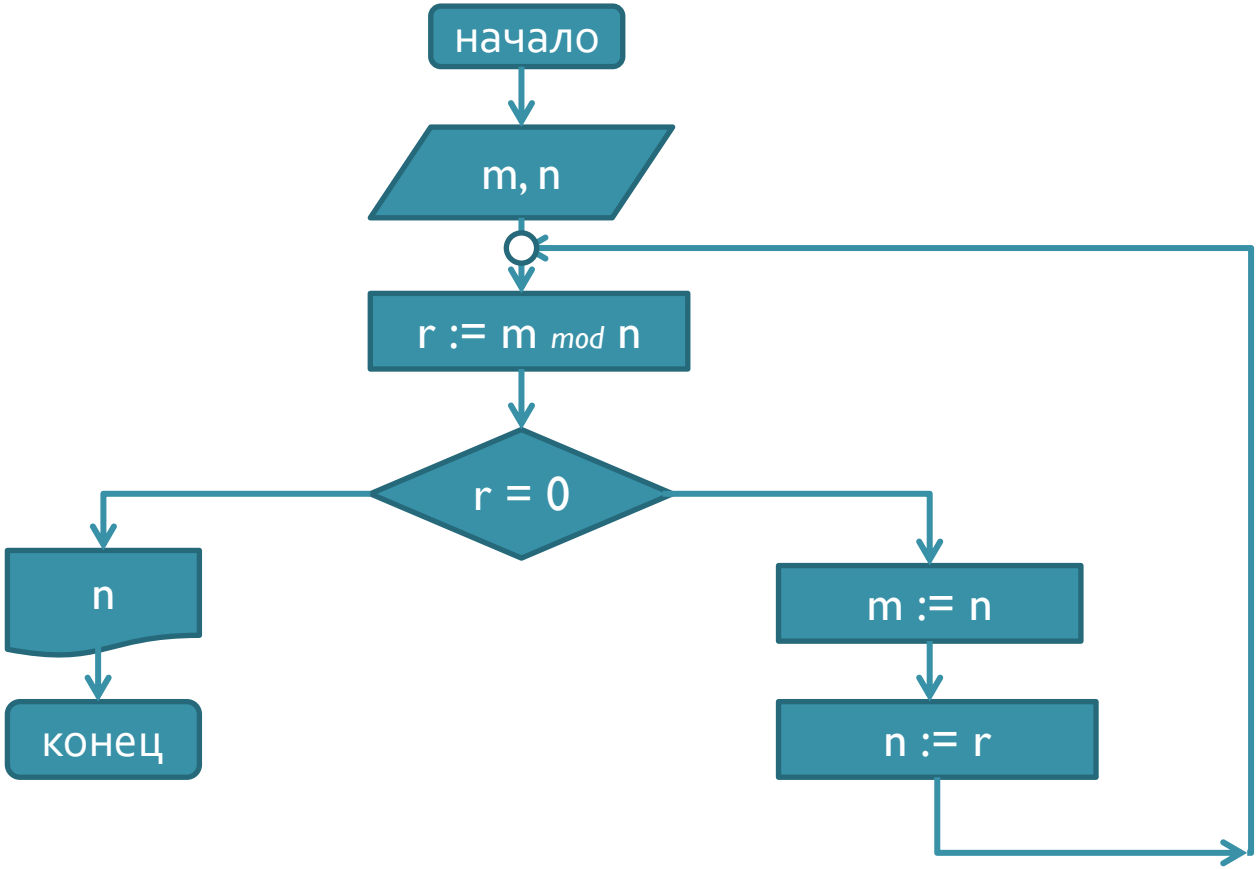
Даны два целых положительных числа m и n. Требуется найти их наибольший общий делитель, т.е. наибольшее положительное целое число, которое нацело делится как на m, так и на n.

E1. {Нахождение остатка} Разделим m на n. Пусть остаток равен r, где $0 \le r \le n$.

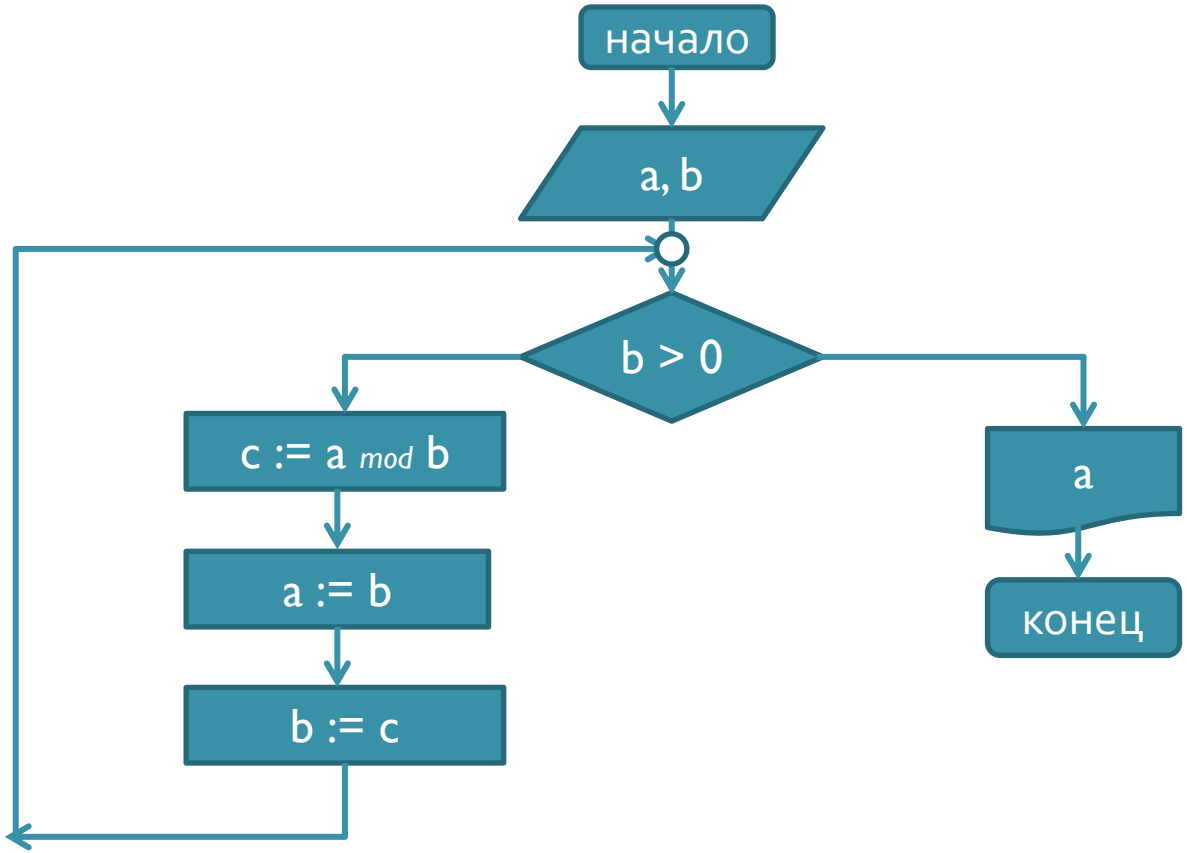
E2. {Проверка на нуль} Если $r = 0$, алгоритм заканчивает работу и ответ n.

E3. {Замена} Положить $m \leftarrow n$, $n \leftarrow r$ и возврат к шагу E1.

Схема алгоритма



Алгоритм Евклида



Алгоритмический язык

- Формальный язык, предназначенный для записи алгоритмов (грамматика, семантика)
- В каждом алгоритмическом языке должны быть средства для задания операторов, осуществляющих переработку информации, и операторов перехода (распознавателей), определяющих порядок выполнения этих операторов
- На практике служат теоретической основой для разработки языков программирования

Классические системы

- Алгоритмические системы:
  - нормальные алгоритмы Маркова
  - рекурсивные функции
  - машины Тьюринга
  - машины Поста и др.

Ориентированы на рассмотрение фундаментальных теоретических вопросов теории алгоритмов и не служат для описания алгоритмов при их реализации на ЭВМ

Язык программирования

- Формальная знаковая система, служащая общению человека с ЭВМ

Основное назначение языка программирования – быть средством программирования, т.е. представлять последовательность действий, подлежащих выполнению на ЭВМ

Lisp – что делать

Prolog – как делать

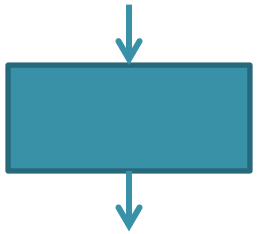
Язык схем алгоритмов

- Предназначен для записи алгоритмов безотносительно к каким-либо вычислительным средствам

(графическое изображение алгоритма, пояснения к алгоритму)

Функциональный узел имеет один вход и один выход

(оператор присваивания вида **y := x**)



—— [**y** := 2 × **x** + **z**] комментарий

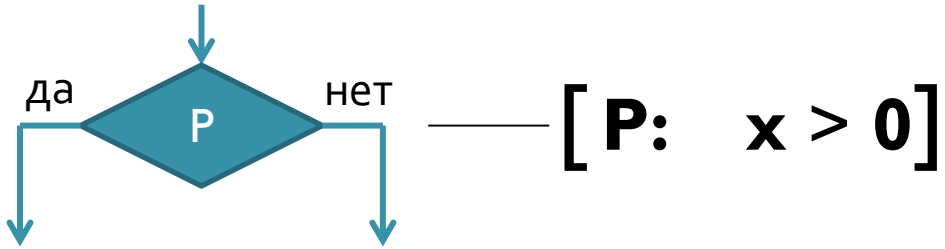
Язык схем алгоритмов (2)

- Выбор направления выполнения алгоритма (*решение*)

Этот функциональный узел имеет один вход и два выхода, где P – предикат, т.е. выражение, принимающее в зависимости от значений входящих в него величин одно из двух возможных значений

(true / false, истина / ложь, да / нет)

(*оператор выбора*)

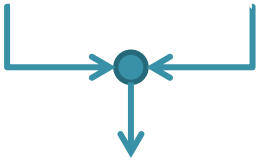


Язык схем алгоритмов (3)

пуск/останов – начало и конец алгоритма

ввод/вывод – преобразование данных из внешнего представления во внутреннее и наоборот

документ – ввод/вывод данных на твердый носитель



соединитель – узел слияния для двух потоков (последовательно при большом количестве потоков)

Пример

$a = 2^{-x}(x + |y|^{1/4})^{1/2}$

Пусть $b = 2^{-x}$

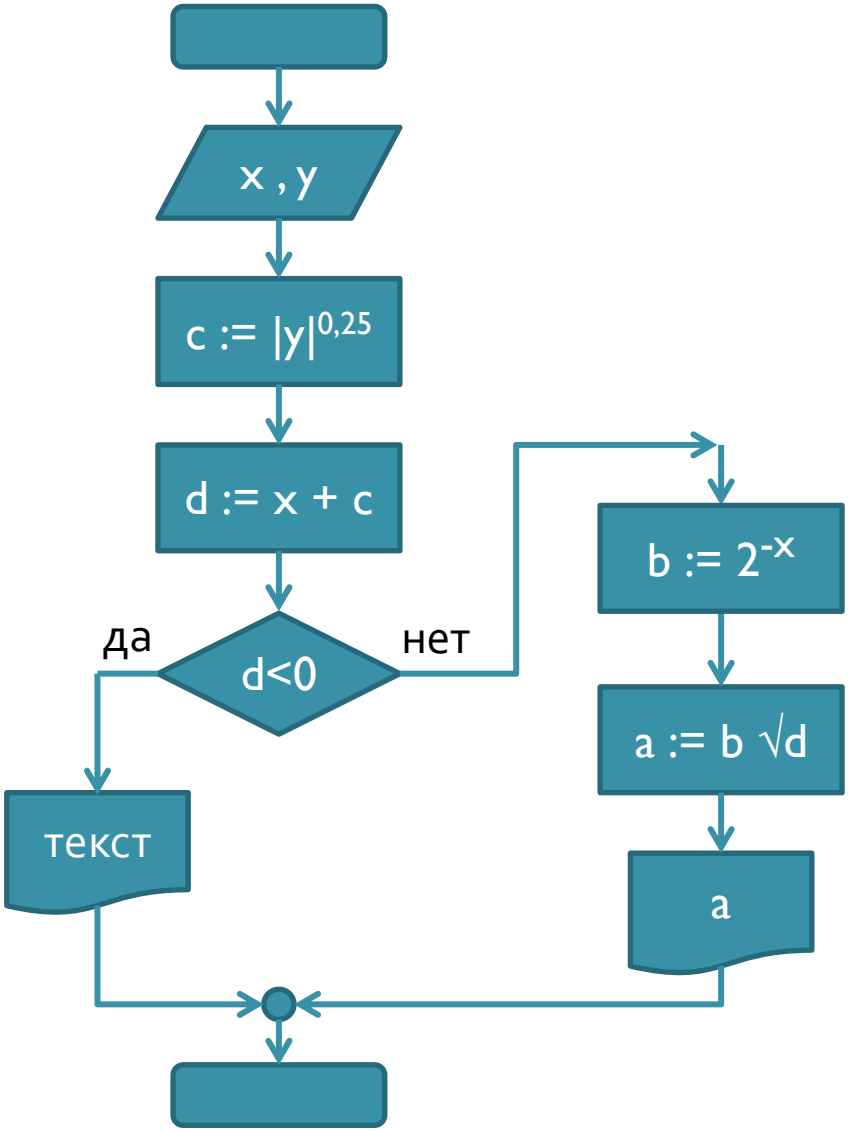
$c = |y|^{1/4}$

$d = x + c$

Тогда

$a = b\sqrt{d}$

– результат декомпозиции



Цикл лекций подготовлен в 2013/2014уч. году Кузнецовым Игорем Ростиславовичем, доцентом кафедры радиоэлектронных средств Санкт-Петербургского Государственного электротехнического университета «ЛЭТИ»

Прочитан в дисциплине
«Информатика»