Общие сведения об алгоритмах.

Лекция №1 по курсу «ОАИП»

Понятие и свойства алгоритма

Алгоритм – это набор точных предписаний, последовательное выполнение которых однозначно приводит задачу к решению за конечное число шагов.

Алгоритм обладает следующими свойствами:

- Детерминированность(определенность,точность) – четкость и ясность всех предписаний: исполнителю алгоритма должно быть точно известно, какая команда алгоритма выполняется следующей («Уходя, гасите свет»)

- Результативность – способность алгоритма приводить к решению задачи за конечное число шагов

- дискретность – предписание представляет собой последовательность четко выраженных отдельных команд, причем, выполнив одну команду, исполнитель выполняет другую команду, промежуточных состояний нет
- массовость (универсальность) – применимость алгоритма к решению задач определенного класса, чем шире этот класс, тем ценнее алгоритм

Существуют следующие способы записи алгоритмов:

- словесно-формульная запись
- графическая запись (схема алгоритма, иначе, графическая схема алгоритма, блок-схема)
- запись на конкретном языке программирования
- псевдокод

- ***Словесный способ*** **записи алгоритмов представляет собой описание последовательных этапов обработки данных. Алгоритм задается в произвольном изложении на естественном языке.**

Пример.

Записать алгоритм нахождения **наибольшего общего делителя (НОД)** двух натуральных чисел (алгоритм Евклида).

Алгоритм может быть следующим:

1. задать два числа
2. если числа равны, то взять любое из них в качестве ответа и остановиться, в противном случае продолжить выполнение алгоритма;
3. определить большее из чисел;
4. заменить большее из чисел разностью большего и меньшего из чисел;
5. повторить алгоритм с шага 2.

Псевдокод

```
Начало
        Ввести a, b и c;
        Если a>b
              то
                 max=a;
              иначе
                 max=b;
        все-если
        Если c>max
              то
                 max=c;
        все-если
Конец
```

Графическая схема алгоритма состоит из отдельных блоков, связанных линиями потоков

Каждый блок описывает конкретный шаг алгоритма

Схемы алгоритмов должны соответствовать действующим стандартам на оформление схем алгоритмов, программ, данных и систем

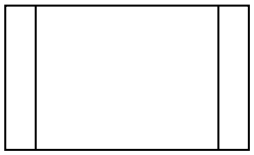
[ГОСТ 19.701-90].

Ниже приводятся некоторые символы, определенные в стандарте и рекомендуемые к использованию в графических схемах алгоритмов.

1.Процесс

Символ отображает функцию обработки данных любого вида.

2.Предопределенный процесс

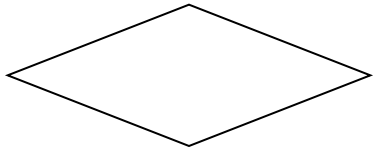


Символ отображает предопределенный процесс, состоящий из одной или нескольких операций или шагов программы, которые определены в другом месте (в подпрограмме, модуле).

3. **Данные**

Символ отображает данные, носитель данных не определен.

4. **Решение**

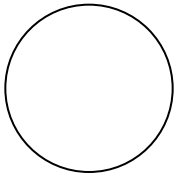


Символ отображает решение или функцию переключательного типа, имеющую один вход и ряд альтернативных выходов, один и из которых может быть активизирован после вычисления условий, определенных внутри этого символа.

5. **Линия**

Символ отображает поток данных или управления

6.Соединитель

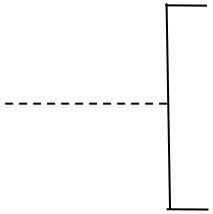


Символ отображает выход в часть схемы и вход из другой части этой схемы и используется для обрыва линии и продолжения ее в другом месте. Соответствующие символы соединители должны содержать одно и то же уникальное обозначение.

7. Терминатор

Символ отображает начало или конец схемы программы, внешнее использование и источник или пункт назначения данных.

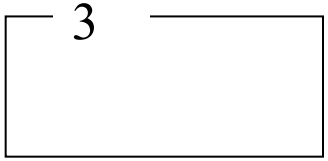
8. Комментарий

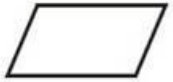
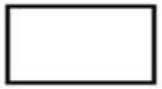
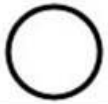
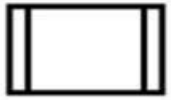
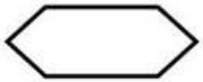
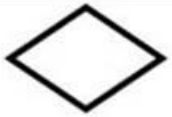
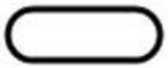
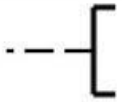


Текст, описывающий функцию символа, следует располагать внутри данного символа.

Если текст не помещается внутри символа, следует использовать символ комментария.

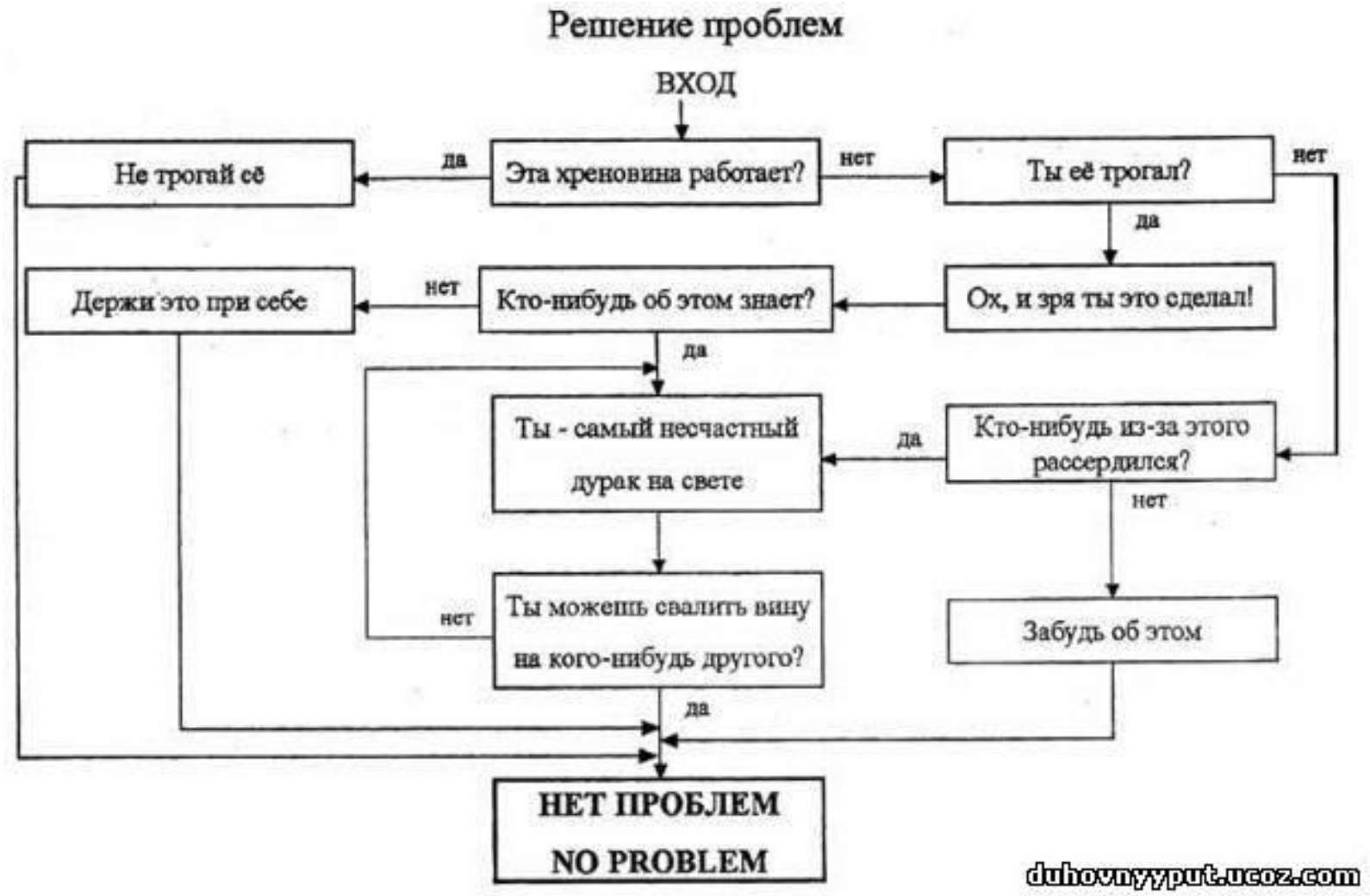
При необходимости блоки в схеме можно нумеровать (например, чтобы иметь возможность ссылаться на тот или иной символ) слева вверху в разъеме символа. Например,



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

Правила выполнения соединений:

- Стандартное направление линий потока – слева направо и сверху вниз
- Если направление потока отличается от стандартного, это направление указывается стрелками
- В схемах следует избегать пересечения линий
- Линии в схемах должны подходить к символу либо слева, либо сверху, а выходить либо справа, либо снизу.
- Вход в блок и выход из блока следует размещать по центру символа


Решение проблем
ВХОД
Эта хреновина работает?
да
Не трогай её
нет
Ты её трогал?
нет
да
Ох, и зря ты это сделал!
Кто-нибудь об этом знает?
нет
Держи это при себе
да
Кто-нибудь из-за этого рассердился?
да
Ты - самый несчастный дурак на свете
нет
Забудь об этом
Ты можешь свалить вину на кого-нибудь другого?
нет
да
НЕТ ПРОБЛЕМ
NO PROBLEM
duhovnyyput.ucoz.com

Профессор на лекции:
- Студенты, не стесняйтесь, спрашивайте. Глупых вопросов не бывает, бывают только глупые ответы.

- Господин профессор, а если я встану на трамвайные рельсы двумя ногами, а руками схвачусь за токопроводящую линию, я поеду как трамвай?

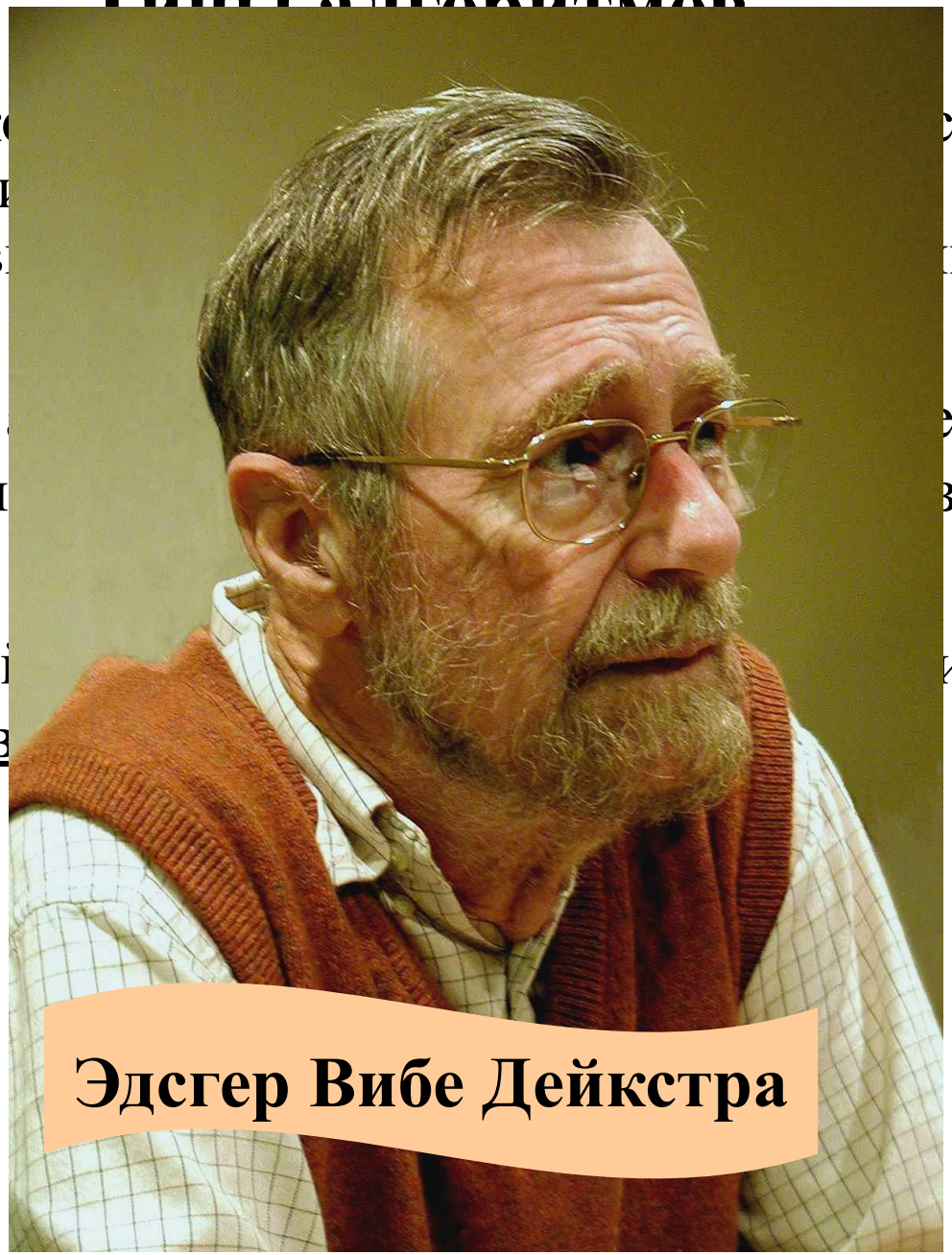
То, что не понял на лекции, поймешь на экзамене!

Типы алгоритмов

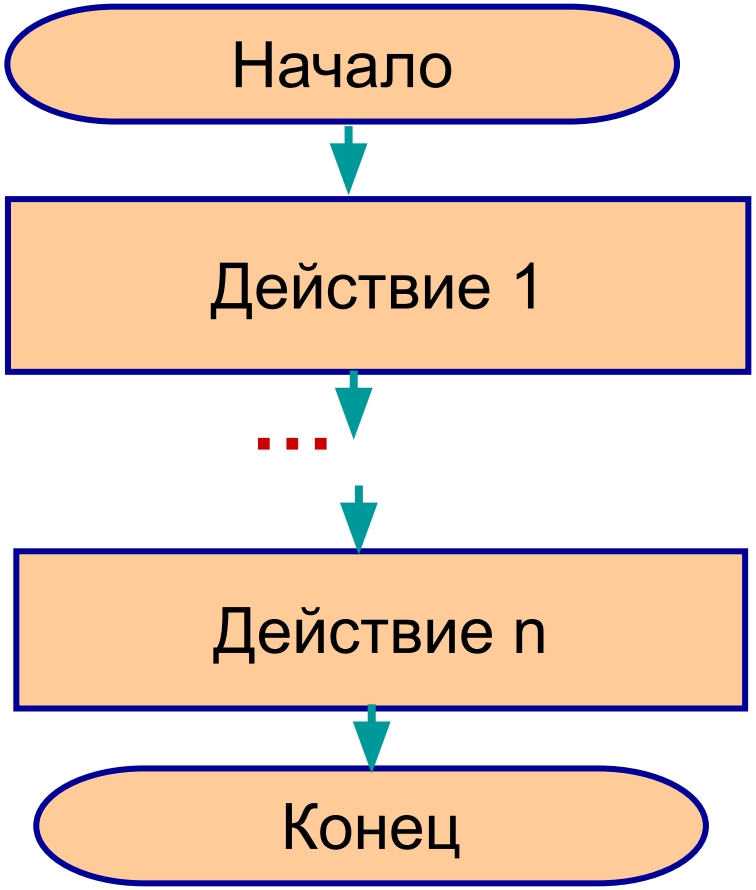
Теорема Дейксти можно реализовать, и следования (линейные), в клические).

Линейный - е действия выполняются оди записаны.

В схеме лине иде типовой структуры **следов**

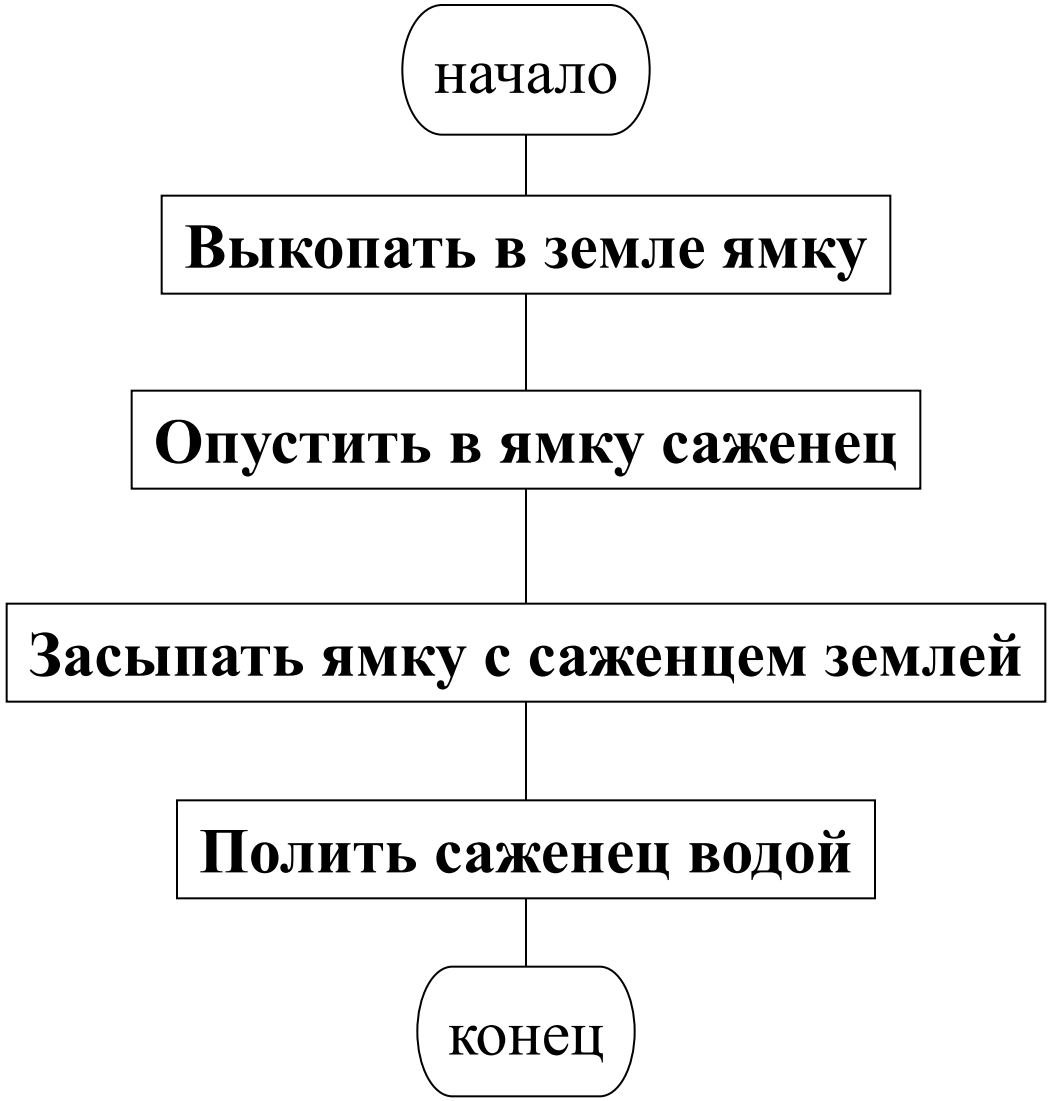


Эдсгер Вибе Дейкстра


Начало
Действие 1
...
Действие n
Конец

Например, алгоритм посадки дерева:

1) **Выкопать в земле ямку;**
2) **Опустить в ямку саженец;**
3) **Засыпать ямку с саженцем землей;**
4) **Полить саженец водой.**


начало
Выкопать в земле ямку
Опустить в ямку саженец
Засыпать ямку с саженцем землей
Полить саженец водой
конец

- **Разветвляющийся -** алгоритм, в котором некоторые действия выполняются один раз или не выполняются в зависимости от заданного условия.

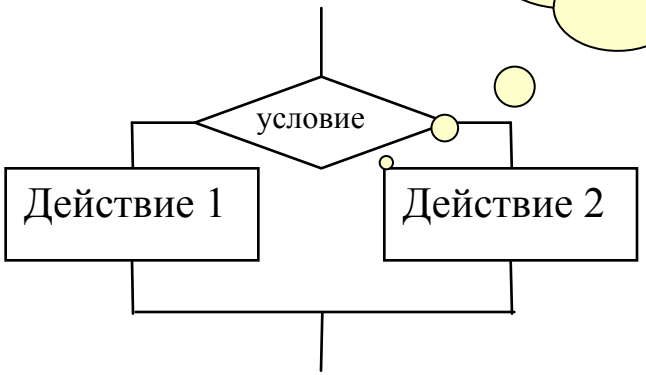
В схеме разветвляющийся алгоритм представляется в виде типовых структур

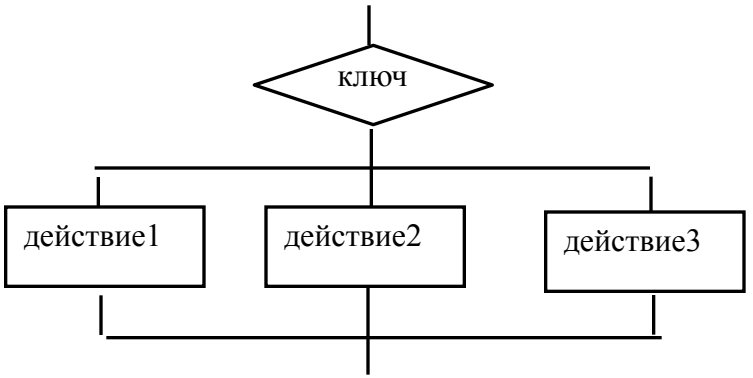
Ветвление и **выбор**

Ветвление

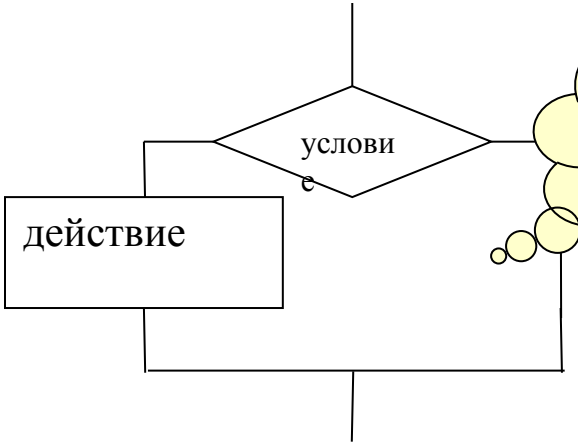
выбор

Полная форма

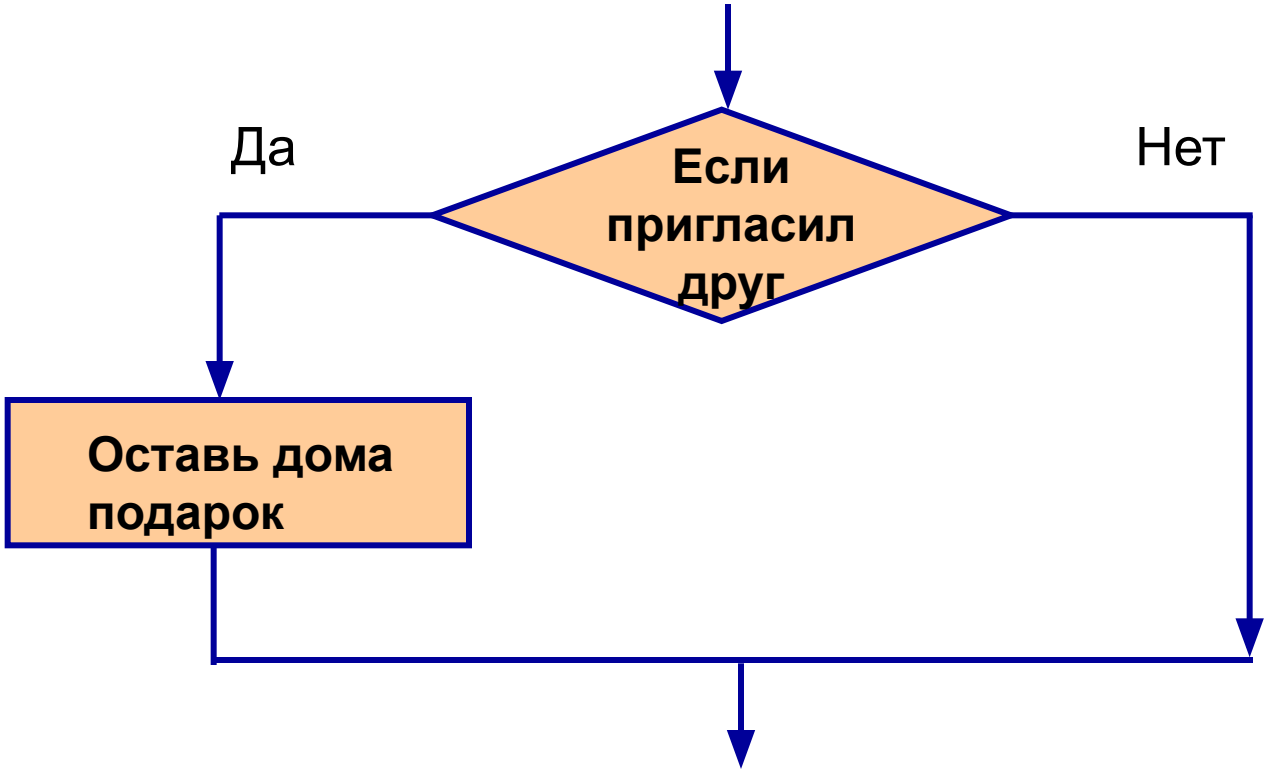


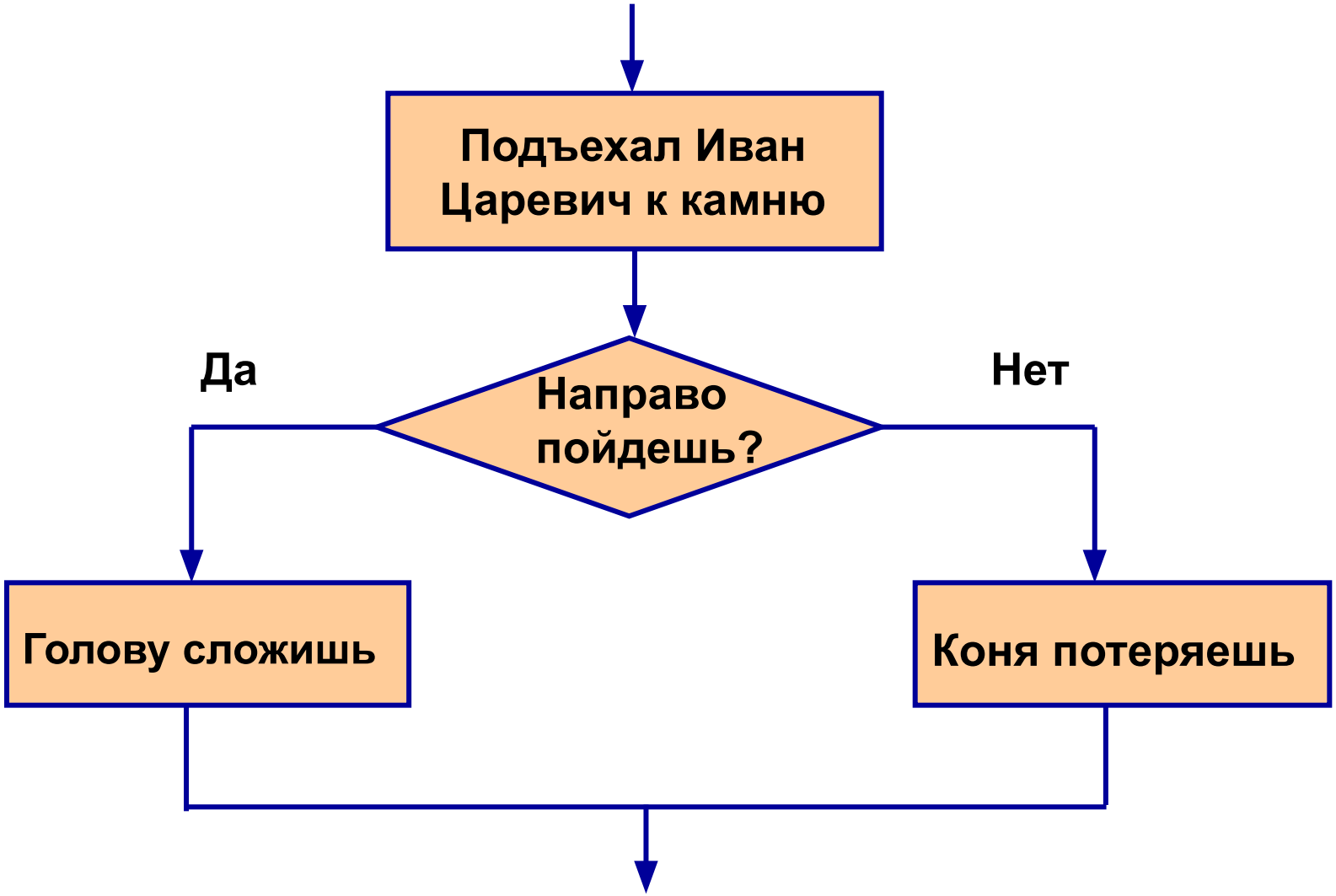


Неполная форма

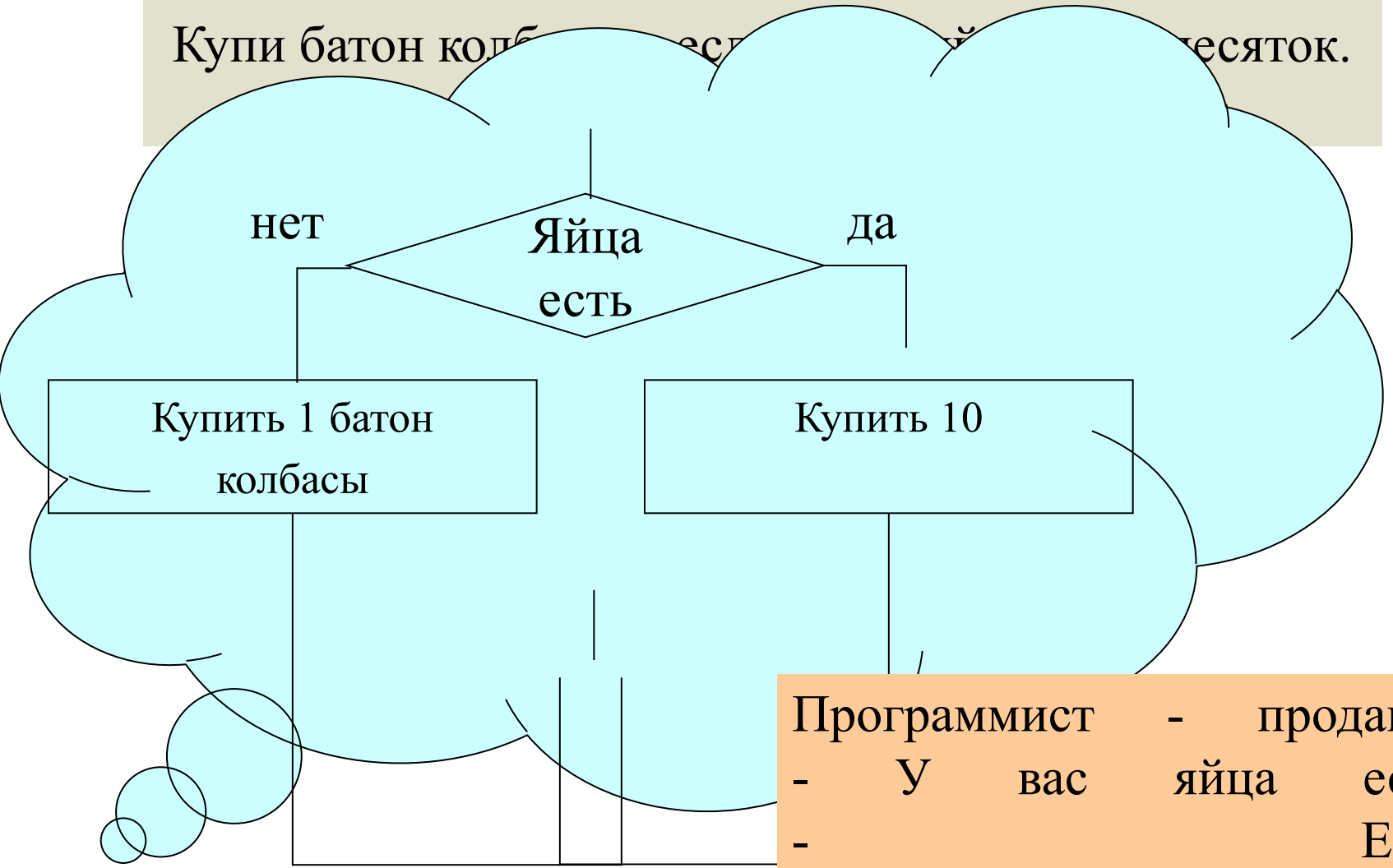


Если друг на день рожденья
Пригласил тебя к себе,
То оставь подарок дома –
Пригодится самому…




Подъехал Иван Царевич к камню
Направо пойдешь?
Да
Нет
Голову сложишь
Коня потеряешь

Жена отправляет программиста в магазин.
Купи батон кол[illegible] десяток.



Программист - продавцу.
- У вас яйца есть?
- Есть!
- ОК. Мне 10 батонов колбасы.

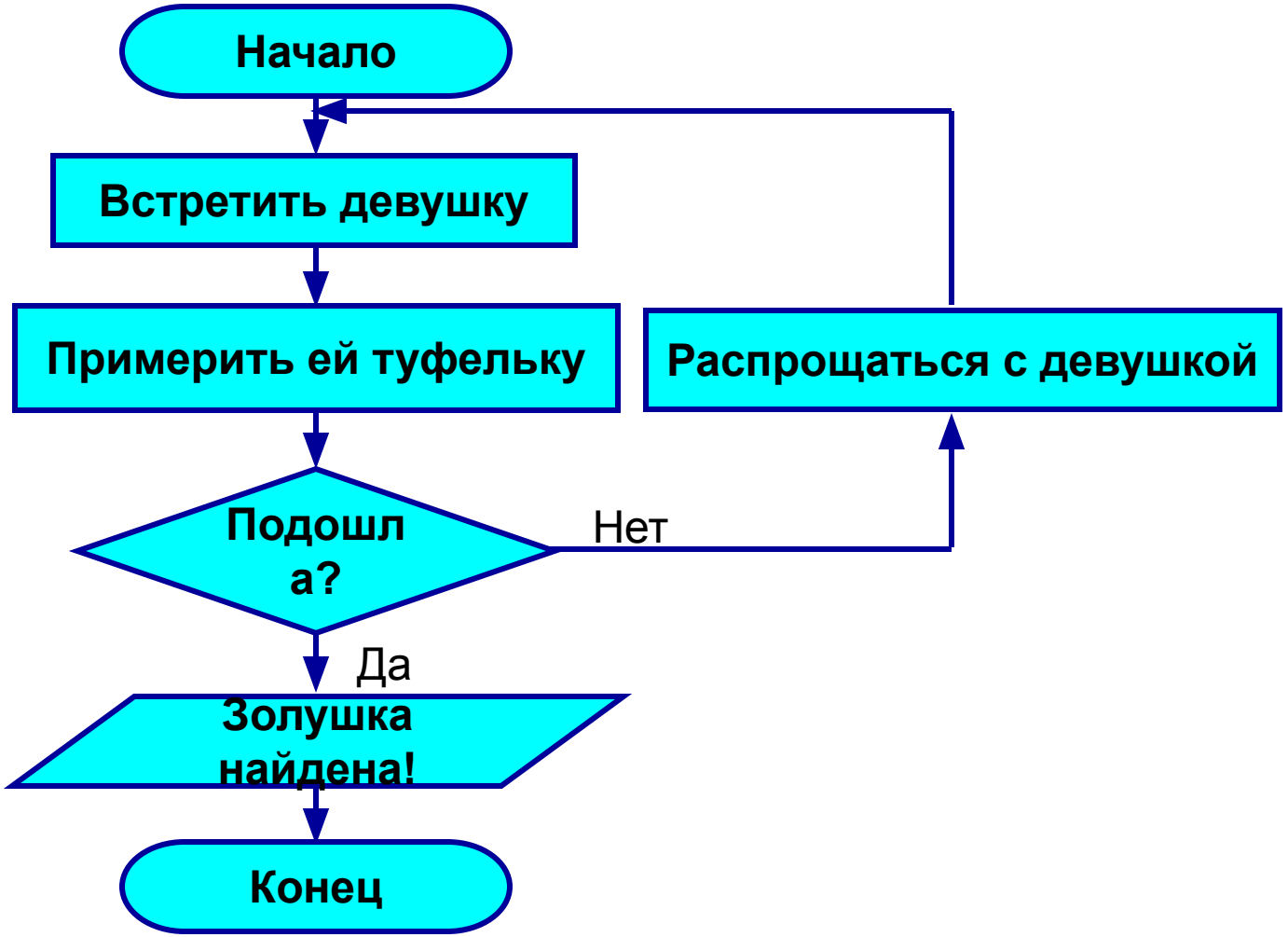
Циклический - алгоритм, в котором некоторая последовательность действий может выполняться несколько раз в зависимости от заданного условия.

В схеме циклический алгоритм представляется в виде типовой структуры **<u>цикл:</u>**


Цикл "Пока"
(цикл с предусловием)
Условие
Нет
Да
Действия


Цикл с постусловием
Действия
Да
Условие
Нет

Алгоритм поиска Золушки:



Итак, алгоритмы делятся на

- линейные
- разветвляющиеся
- циклические

(можно также выделить в отдельный тип смешанные).

Пример1. Вычислить значение функции

$$f = 2\sin^3 h + \frac{x - 2{,}3}{\cos x}$$

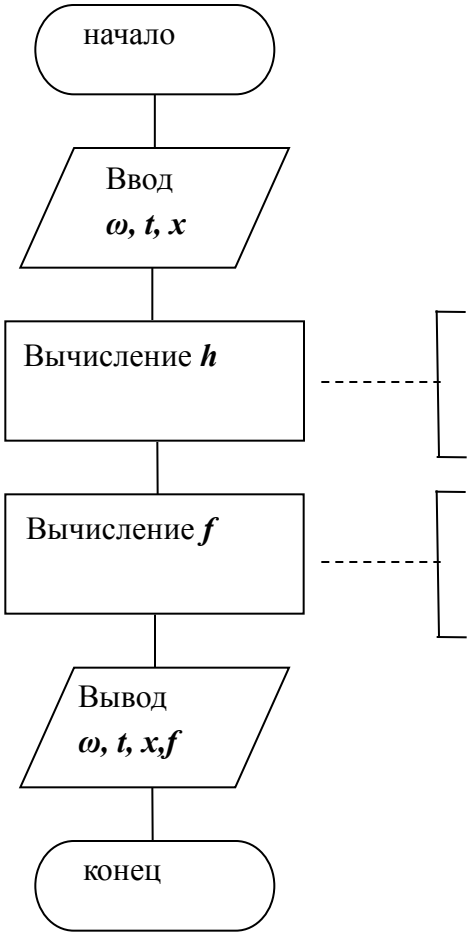
,где

$$h = 4\omega t - x$$

Исходными данными являются ω, t, x.

Результат – f.

Промежуточная величина – h.



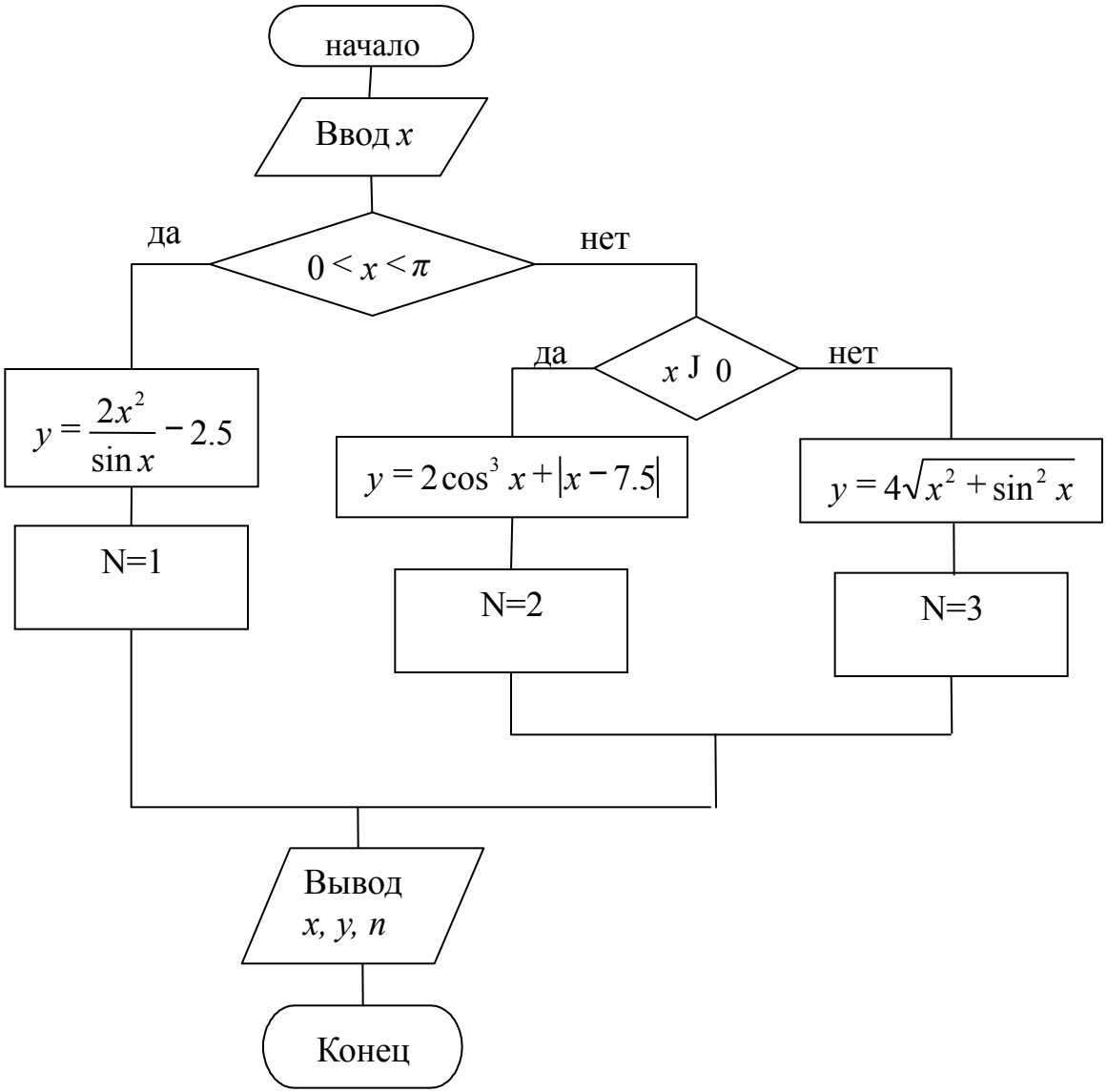
$$h = 4\omega t - x$$

$$f = 2\sin^3 h + \frac{x - 2{,}3}{\cos x}$$

Пример 2. Составить алгоритм вычисления функции.

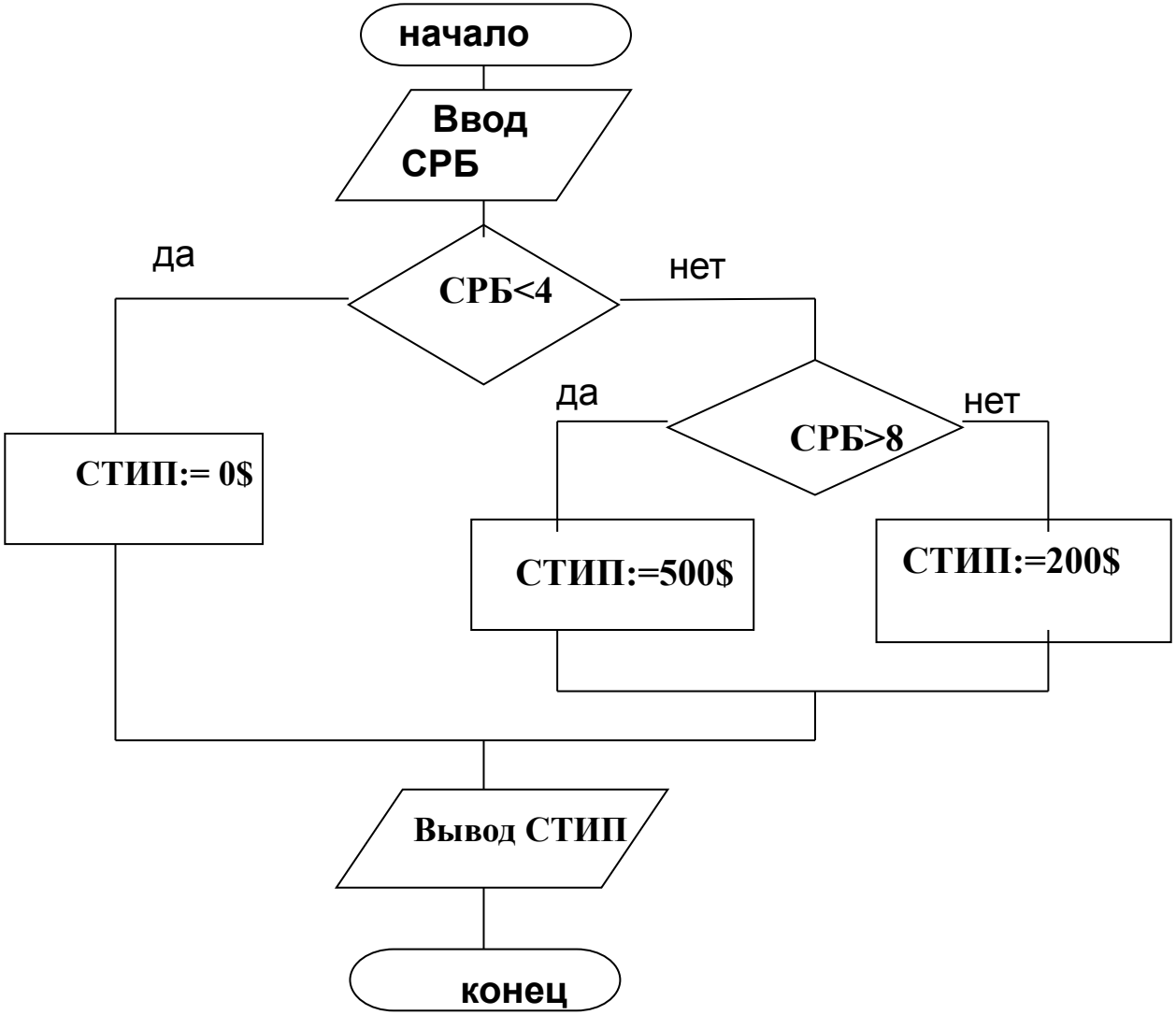
$$y = \begin{cases} \dfrac{2x^2}{\sin x} - 2{,}5 & \textit{если} \quad 0 < x < \pi \\ 2\cos^3 x + |x - 7{,}5| & \textit{если} \quad x \le 0 \\ 4\sqrt{x^2 + \sin^2 x} & \textit{в остальных случаях} \end{cases}$$

Предусмотреть вывод номера расчетной формулы.


начало
Ввод x
да
$0 < x < \pi$
нет
да
x J 0
нет
$y = \frac{2x^2}{\sin x} - 2.5$
$y = 2\cos^3 x + |x - 7.5|$
$y = 4\sqrt{x^2 + \sin^2 x}$
N=1
N=2
N=3
Вывод
x, y, n
Конец

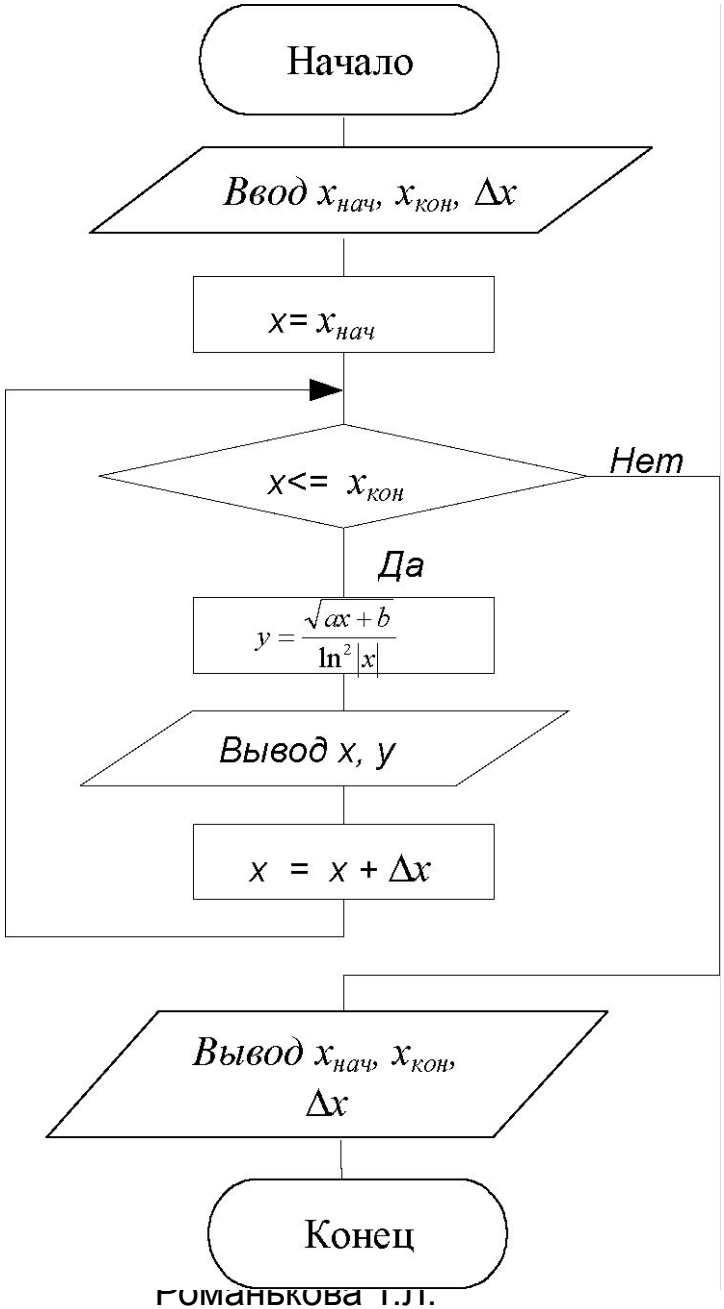
Пример3. Примером разветвляющегося алгоритма может служить алгоритм начисления стипендии по среднему баллу.

- в качестве исходного данного задается значение среднего балла сдачи сессии студеном;
- если средний балл меньше **4**, то стипендия – **0$**;
- если средний балл больше **8**, то начисляется стипендия в **500$**;
- в остальных случаях начисляется стипендия размером в **200$**;
- выводится значение начисленной стипендии.

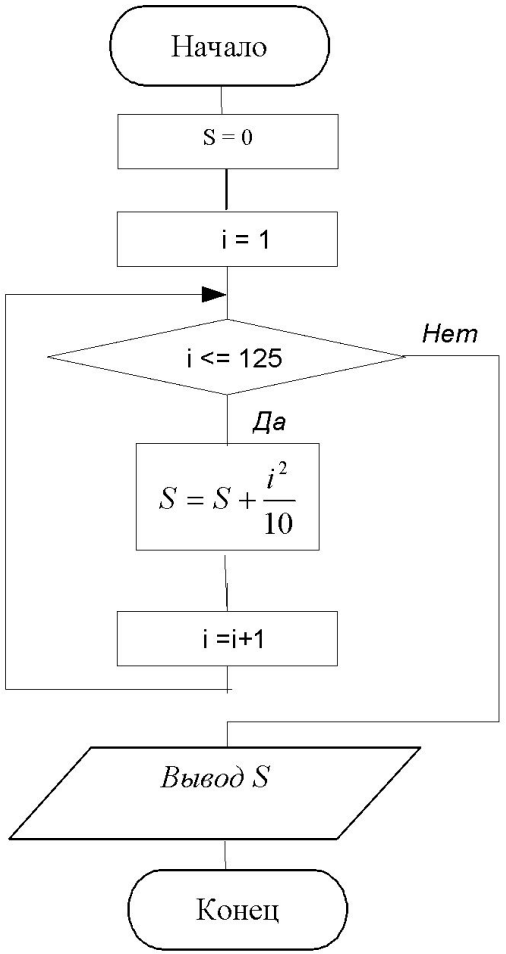

начало
Ввод СРБ
да
СРБ<4
нет
СТИП:= 0$
да
СРБ>8
нет
СТИП:=500$
СТИП:=200$
Вывод СТИП
конец

Пример 4. Составить алгоритм вычисления функции для значений аргумента x, изменяющегося в интервале от хнач до хкон с шагом Δx, и заданных констант a и b. Такая задача называется задачей о **табулировании функции**.

$$y = \frac{\sqrt{ax + b}}{\ln^2 |x|}$$


Начало
Ввод $x_{нач}$, $x_{кон}$, Δx
$x = x_{нач}$
$x <= x_{кон}$
Нет
Да
$y = \frac{\sqrt{ax+b}}{\ln^2 |x|}$
Вывод x, y
$x = x + \Delta x$
Вывод $x_{нач}$, $x_{кон}$, Δx
Конец

Пример 5. Ниже приведен алгоритм вычисления $\sum_{i=1}^{125} \frac{i^2}{10}$



Алгоритмы могут классифицироваться и по другому направлению.

- Комбинаторные алгоритмы:
  - ❖ Общие комбинаторные алгоритмы (например, генерация случайных чисел)
  - ❖ Алгоритмы на графах
  - ❖ Алгоритмы поиска
  - ❖ Алгоритмы сортировки
  - ❖ Алгоритмы слияния
  - ❖ Алгоритмы работы со строками

- Алгоритмы сжатия данных
- Криптографические алгоритмы
- Теоретико-числовые алгоритмы
- Цифровая обработка сигналов

И т.д.

Основные принципы разработки и анализа алгоритмов

При построении алгоритма для сложной задачи используют системный подход с использованием ***декомпозиции*** (нисходящее проектирование сверху-вниз) и ***синтеза*** (программирование снизу-вверх). При формировании алгоритма используют дедуктивный и индуктивный методы.

При дедуктивном подходе рассматривается частный случай общеизвестных алгоритмических моделей. Здесь при заданных предположениях известный алгоритм приспосабливается к условиям решаемой задачи.

Индуктивный способ предполагает эвристический системный подход. В этом случае общих и наиболее удачных методов не существует. Возможны некоторые подходы, позволяющие в каждом конкретном случае находить и строить алгоритмы.

Одним из системных методов разработки алгоритмов является ***структурное программирование***.

Принципы структурного программирования:

Принцип абстракции.

Этот принцип позволяет разработчику рассматривать программу в нужный момент без лишней детализации. Детализация увеличивается при переходе от верхнего уровня абстракции к нижнему.

Принцип формальности.

Он предполагает строгий методический подход к программированию, придает творческому процессу определенную строгость и дисциплину

Принцип модульности.

В соответствии с этим принципом программа разделяется на отдельные законченные фрагменты, модули, которые просты по управлению и допускают независимую отладку и тестирование. В результате отдельные ветви программы могут создаваться разными группами программистов.

Принцип иерархического упорядочения.

Взаимосвязь между частями программы должна носить иерархический, подчиненный характер.

Цели структурного программирования:

1) повысить надеж[программа легко поддавалась тести[адке. Достигается это хорошим струк[ровании;

2) повысить эфф[достигнута при структурировании [так, чтобы можно было бы легко нах[гобы текст любого модуля с целью [ты можно было переделать незави[

3) уменьшить вре[. Достижимо при повышении произ[

4) улучшить читабельность программ;