ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ДИСТАНЦИОННОГО ОБУЧЕНИЯ И ПОВЫШЕНИЯ КВАЛИФИКАЦИИ

Кафедра «Вычислительные системы и информационная безопасность»

Учебно-методическое пособие

«Алгоритмизация»
по дисциплине

«Информатика и ИКТ»

Авторы
Полуян А.Ю.,
Петренкова С.Б.

Ростов-на-Дону, 2018

Аннотация

Учебно-методическое пособие предназначено для студентов всех форм обучения направлений 10.03.01 «Информационная безопасность», 13.03.02 «Электроэнергетика и электротехника», 10.05.02 «Информационная безопасность телекоммуникационных систем»

Методические указания предназначены для проведения лабораторных работ по дисциплине "Информатика и ИКТ". Содержит общие сведения об алгоритмах, позволяет освоить: способы описания алгоритмов, основные приемы построения алгоритмов для решения различных задач. Лабораторная работа включает набор заданий, методические указания к ним и контрольные вопросы по изучаемой теме. Методические рекомендации могут быть использованы для самостоятельной работы.

Авторы

к.т.н., доцент кафедры «Вычислительные системы и информационная безопасность» Полуян А.Ю.,
к.т.н., доцент кафедры «Вычислительные системы и информационная безопасность» Петренкова С.Б.

Оглавление

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ АЛГОРИТМИЗАЦИИ

1.1 Понятие алгоритма. Свойства алгоритма.

Алгоритм - это последовательность действий, приводящих к требуемому результату.

Таким образом, при разработке алгоритма решения задачи математическая формулировка преобразуется в процедуру решения, представляющую собой последовательность арифметических действий и логических связей между ними. При этом алгоритм обладает следующими свойствами:

1) Дискретность - процесс преобразования данных, т.е. на каждом шаге алгоритма выполняется очередная одна операция;

2) Результативность - алгоритм должен давать некоторый результат;

3) Конечность - алгоритм должен давать результат за конечное число шагов;

4) Определенность - все предписания алгоритма должны быть однозначны, понятны пользователю;

5) Массовость - алгоритм должен давать решения для целой группы задач из некоторого класса, отличающихся исходными данными;

Действия в алгоритме выполняются в порядке их записи. Нельзя менять местами никакие два действия алгоритма, а так же нельзя не закончив одного действия переходить к следующему.

1.2 Формы записи алгоритмов

На практике наиболее распространены следующие формы представления

алгоритмов:

· словесная (запись на естественном языке);

· графическая (изображения из графических символов);

· псевдокоды (полуформализованные описания алгоритмов на услов-

ном алгоритмическом языке, включающие в себя как элементы языка

программирования, так и фразы естественного языка, общепринятые

математические обозначения и др.);

· программная (тексты на языках программирования).

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

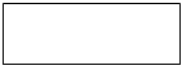
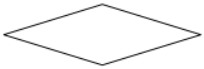
весный способ не имеет широкого распространения, так как такие описания:

· строго не формализуемы;

· страдают многословностью записей;

· допускают неоднозначность толкования отдельных предписаний.

Графический способ представления алгоритмов является более компактным и наглядным по сравнению со словесным. При графическом представлении алгоритм изображается в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий. Такое графическое представление называется схемой алгоритма. В схеме алгоритма каждому типу действий (вводу исходных данных, вычислению значений выражений, проверке условий, управлению повторением действий, окончанию обработки и т.п.) соответствует геометрическая фигура, представленная в виде блочного символа. Блочные символы соединяются линиями переходов, определяющими очередность выполнения действий. В таблице приведены наиболее часто употребляемые символы.

Название символа	Обозначение	Выполняемая функция
Начало/конец		Начало или конец алгоритма
Процесс вычислений		Выполняет вычислительное действие или группу действий
Логический блок		Выбор направления выполнения алгоритма в зависимости от условия

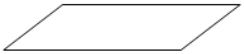
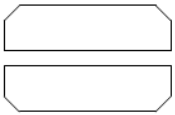
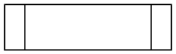
Ввод /вывод		Отображение данных
Граница цикла		Отображает начало и конец цикла
Предопределенный процесс		Выполнение операций в подпрограмме
Соединитель		Указание связи между прерванными линиями в пределах одной страницы
Комментарий		Пояснительная запись

Схема алгоритма выстраивается в одном направлении: либо сверху вниз, либо слева направо. Все повороты соединительных линий выполняются под углом 90 градусов.

Общими правилами при построении схем алгоритмов являются следующие:

- В начале алгоритма должны быть блоки ввода значений входных данных.
- После ввода значений входных данных могут следовать процесс вычислений и блоки условия.
- В конце алгоритма должны располагаться блоки вывода значений выходных данных.
- В алгоритме должен быть только один блок начала и один блок окончания.

Связи между блоками указываются направленными или ненаправленными линиями.

Псевдокод представляет собой систему обозначений и правил, предназначенную для единообразной записи алгоритмов. Псевдокод занимает промежуточное место между естественным и формальным языками. С одной стороны, он близок к обычному естественному языку, поэтому алгоритмы могут на нем записываться и читаться как обычный текст. С другой стороны, в псевдокоде используются некоторые формальные конструкции и математическая символика, что приближает запись алгоритма к общепринятой математической записи.

1.3 Алгоритмы линейной структуры

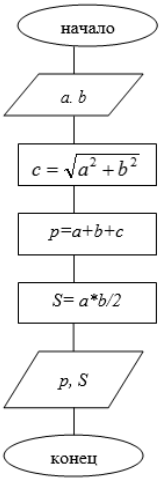
Алгоритм, в котором действия выполняются последовательно друг за другом, называется ***линейным алгоритмом.***

Рассмотрим пример составления схемы линейного алгоритма:

Пример 1. Вычислить периметр и площадь прямоугольного треугольника по длинам двух катетов.

Решение. Гипотенуза прямоугольного треугольника вычисляется по формуле Пифагора $c = \sqrt{a^2 + b^2}$, где *a, b* – катеты. Периметр и площадь прямоугольного треугольника определяются по формулам: *p=a+ b+c*, *S= a*b/2.*

Схема алгоритма примера (рис.1) иллюстрирует, что сначала вводятся исходные данные *a, b* затем вычисляется гипотенуза, периметр, площадь. Все вычисления производятся последовательно и менять их местами нельзя.



1.4 Алгоритмов разветвляющейся структуры

На практике редко удается представить схему алгоритма решения задачи

в виде линейной структуры. В программу может быть включено условие (на-

пример, выражение отношения или логическое отношение), в зависимости от

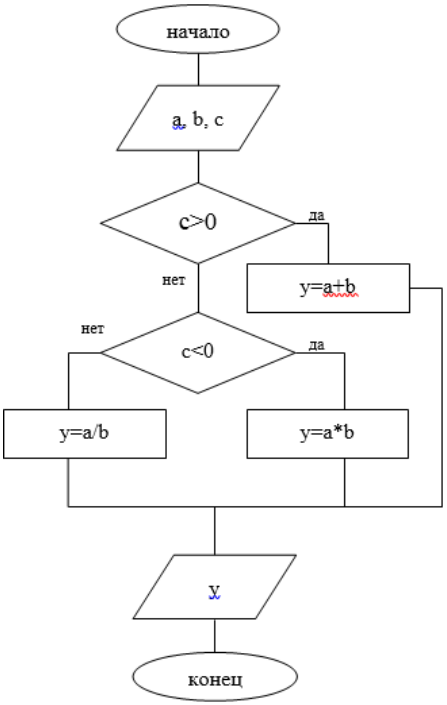
которого, вычислительный процесс идет по той или иной ветви. Алгоритм такого вычислительного процесса называется

алгоритмом разветвляющейся структуры. В общем случае количество ветвей в таком алгоритме не обязательно равно двум.

Пример 2. Составить схему алгоритма вычисления выражения

$$y = \begin{cases} a+b, & c > 0 \\ a*b, & c < 0 \\ a/c, & c = 0 \end{cases}$$

После входа исходных данных (переменных a, b, c) проверяется значение переменной c в логическом блоке. Если условие выполняется (истина), то после выполнения блоков первой ветви нет необходимости выполнять блоки второй ветви, осуществляется переход сразу к выводу результата и концу алгоритма. Если условие не выполняется (ложь), то переходим к проверки следующего логического условия. Решение указывается условием перехода на соответствующую ветвь.



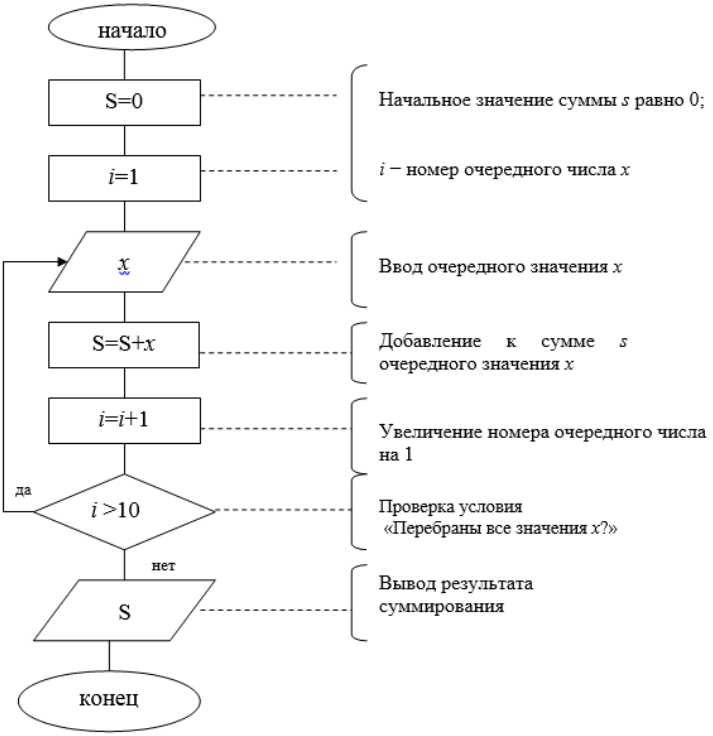
1.5 Алгоритмы циклической структуры

Алгоритм, в котором вычисления повторяются по одной и той же совокупности формул, называется ***циклическим алгоритмом.*** *Цикл* – это многократно повторяемый участок алгоритма.

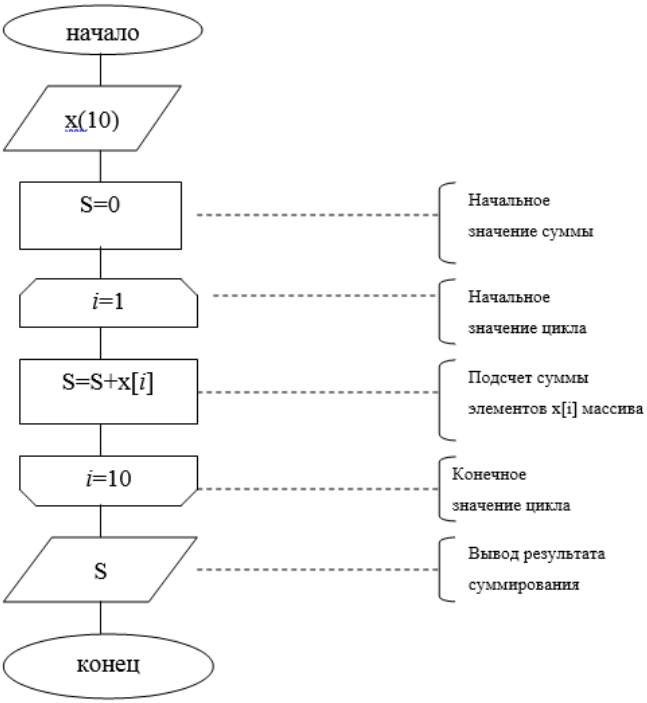
Массив это совокупность переменных, которые имеют одно и то же имя и тип. Элементы массива различаются по индексу. Имя общее, индекс оригинальный. Упорядоченность данных в массиве позволяет обращаться к любому элементу массива по его номеру (индексу), а однотипность данных позволяет использовать циклическую обработку всех элементов Различают одномерные массивы (1 индекс) – они используются для представления векторов и двумерные массивы (2 индекса) они используются для представления матриц.

Пример 3. Составить циклический алгоритм вычисления суммы десяти чисел $S=\sum_{i=1}^{10} x_i$.

Вариант 1 построения алгоритма. Здесь в качестве переменной цикла используется переменная *i* с начальным значением, равным единице, и конечным значением, равным 10, и шагом, равным единице. В этом цикле проверка условия выхода из цикла выполняется в конце цикла. При этом тело цикла повторится десять раз.

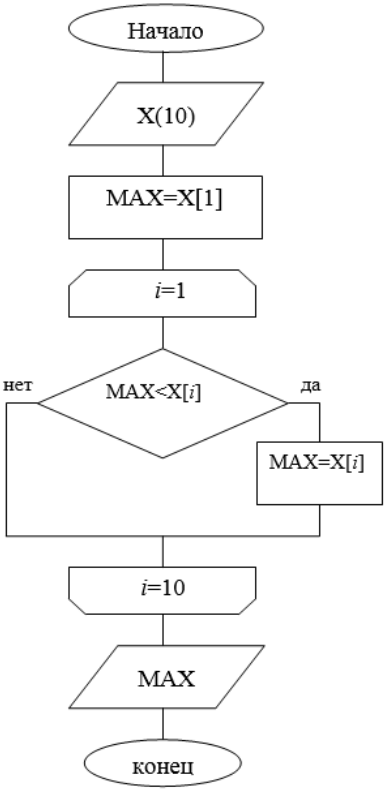


Вариант 2 построения алгоритма. Схема алгоритма получается во многих случаях более компактной и наглядной, если для ее построения использовать блоки начала и конца цикла, который выполняет все функции, необходимые для его организации. В цикле последовательно суммируются все элементы x_i массива с начальным значением S=0.

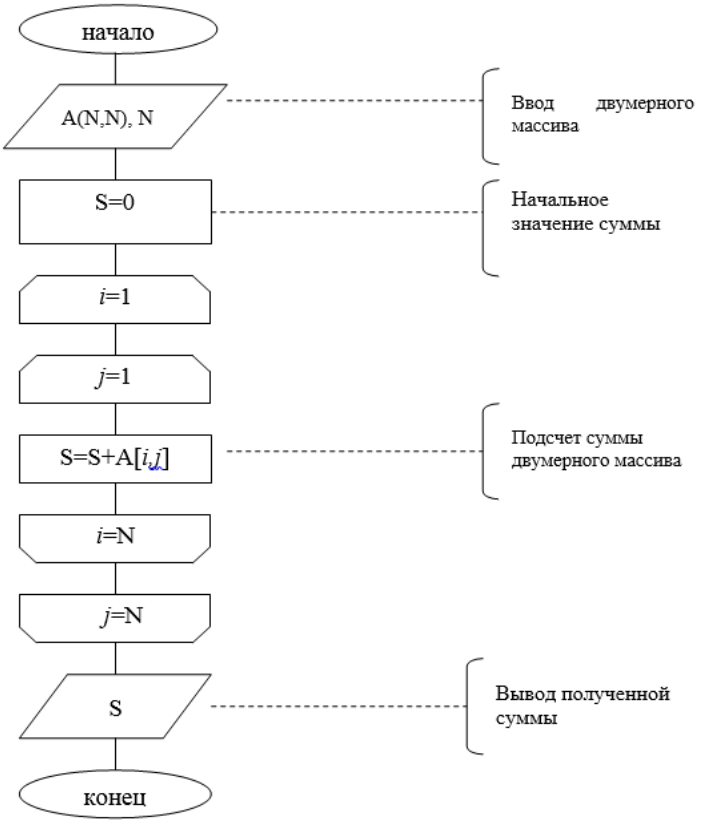


Пример 4. Найти наибольший элемент одномерного массива X(10).

Процесс определения наибольшего элемента заключается в цикле сравнивая текущий элемент массива с некоторым, например, с первым, условно принятым за наибольшим. Если текущее значение массива окажется больше наибольшего из предыдущих значений, то его надо считать новым наибольшим значением.



Пример 5. Задан двумерный массив A(N,N), найти сумму элементов массивов.



ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Понятие алгоритма
2. Свойства алгоритмов
3. Способы записи алгоритмов
4. Формы представления алгоритмов
5. Алгоритмизация линейных вычислительных процессов
6. Алгоритмизация ветвящихся вычислительных процессов
7. Алгоритмизация циклических процессов

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

1. Построить схему алгоритма $y=\sqrt{a^2+b^2}$ где

$$a=\begin{cases} tg\ x, если\ x>2 \\ x^2, если\ 0\le x\le 2 \\ 4, если\ -1\le x<0 \\ \sqrt{\sin x}, в\ остальных\ случаях \end{cases}$$

2. Построить схему алгоритма

$$y=\frac{a\sqrt{b}+b\sqrt{a}}{2} \quad где \quad a=\begin{cases} e^x, если\ b>0 \\ 0, если\quad b=0 \\ \sin\ x, в\ остальных\ случаях \end{cases}$$

3. Построить схему алгоритма

$$y=ax^2+b, где\ a=\begin{cases} e^x, если\ x<b \\ \sin x, если\ x=b \\ \log_{\sqrt{b}}(\sin x), если\ x>b \end{cases}$$

4. Построить схему алгоритма

$$y=\begin{cases} 1-e^{ax}\sin(ax+b), при\ x>\pi \\ 1-e^{-ax}(ax+b), при -\pi\le x\le\pi \\ 1-\left(e^{-ax}+e^{-bx}\right), при\ x<-\pi \end{cases}$$

$$где\ x=ab^2-\sin b$$

5. Построить схему алгоритма

$$y=\begin{cases} \frac{1}{\sqrt{d}}arctg\frac{b+c}{\sqrt{d}}, при\ d>0 \\ -\frac{1}{b+c}, при\ d=0 \\ \frac{1}{2\sqrt{d}}\ln\sqrt{d}, при\ d>5 \end{cases} \qquad где\ d=ac-b^2$$

6. Построить схему алгоритма

$$n=\begin{cases} 1, если\ x\ge 0\ и\ y\ge 0 \\ 2, если\ x<0\ и\ y\ge 0 \\ 0, в\ остальных случаях \end{cases} \qquad где\ y=\sqrt[3]{\left|tgx^2+\sqrt{\sin x+\cos x}\right|}$$

7. Построить схему алгоритма

$$y=\frac{1}{x}+\frac{\sqrt{e^x+e^z}}{x} \quad где \quad z=\begin{cases} x^3, если\ x>b \\ 0, если\ x<b \\ 7x, если\ x=b \end{cases}$$

1. Построить схему алгоритма: $\sum \frac{1}{(2i)^2}$
2. Дано натуральное n. Построить схему алгоритма: $\sum_{K=1}^{n} \frac{1}{K}$
3. Дано натуральное n. Построить схему алгоритма: $\sum_{K=1}^{n} \frac{1}{K^5}$
4. Дано натуральное n. Построить схему алгоритма: $\sum_{K=1}^{n} \frac{1}{(2K+1)^2}$
5. Задан одномерный массив из n элементов. Найти среднее арифметическое всех элементов массива.
6. Задан одномерный массив из n элементов. Найти наименьший элемент в массиве.
7. Задан одномерный массив из n элементов. Найти количество положительных элементов массива.
8. Задан одномерный массив из n элементов. Найти количество отрицательных элементов массива.
9. Задан одномерный массив из n элементов. Определить, сколько раз встречается число 7 среди элементов массива.
10. Задан одномерный массив из n элементов. Определить, сколько элементов массива меньше, чем число 6.
11. Задан одномерный массив из n элементов. Определить, сколько элементов массива больше, чем число 3.
12. Задан одномерный массив из n элементов. Найти сумму всех неотрицательных элементов массива.
13. Задан двухмерный массив. Найти сумму всех элементов массива, имеющих четные индексы.
14. Задан двухмерный массив. Найти наибольший из элементов массива, имеющих нечетные индексы.
15. Задан двухмерный массив. Найти среднее арифметическое всех положительных элементов массива.
16. Задан двухмерный массив. Найти среднее арифметическое всех отрицательных элементов массива.
17. Задан двухмерный массив. Найти сумму элементов массива, превышающих число 5.
18. Задан двухмерный массив. Найти наибольший элемент в третьем столбце матрицы.
19. Задан двухмерный массив. Расположить все элементы мат-

рицы в строку в порядке возрастания.

20.Задан двухмерный массив. Найти сумму всех положительных элементов матрицы.

СПИСОК ЛИТЕРАТУРЫ

1. А.В. Басова, О.В. Смирнова и др. Краткий курс информатики, Ростов н/Д. ЮФУ, 2008г.
2. Ю.А.Стоцкий. Самоучитель. Office XP. — Питер, 2003г.
3. С.В. Симонович. Информатика. Базовый курс. — Питер, 2002г.
4. О.Э. Згадзай, С.Я. Казанцев, Л.А. Казанцева. Информатика для юристов. — Москва, 2001г.
5. Н. Угринович. Информатика и информационные технологии — М., БИНОМ, 2003г.
6. Немнюгин С.А. Turbo Pascal. СПб: Питер, 2000.
7. Информатика. Базовый курс/ Под ред. С.В. Симонович – СПб: Издательство «Питер», 2000. – 640 с.
8. Информатика: Учебник. /Под ред. Н.В. Макаровой. – М: Финансы и статистика, 2001. – 768 с.
9. Фигурнов В.Э. IBM PC для пользователя. — М.: Информ. — М., 2000 — 432 с.
10. Р. Хершель. TURBO PASCAL — М.: МП «МИК», 1991. — 342 с.