МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Институт сферы обслуживания и предпринимательства (филиал)
федерального государственного бюджетного образовательного
учреждения высшего образования «Донской государственный
технический университет» в г. Шахты Ростовской области
(ИСОиП (филиал) ДГТУ в г. Шахты)

КОЛЛЕДЖ ЭКОНОМИКИ И СЕРВИСА

На правах рукописи

ТЕОРИЯ АЛГОРИТМОВ

Методические указания
по выполнению практических работ
для подготовки обучающихся специальности
09.02.03 «Программирование в компьютерных системах»
очной и заочной форм обучения

Рассмотрены и рекомендованы для
использования в учебном процессе на
заседании педагогического совета
Протокол № 1от «31» сентября 2018 г

Шахты
ИСОиП (филиал) ДГТУ в г. Шахты
2019

Составитель:

Преподаватель
КЭС ИСОиП (филиал) ДГТУ в г. Шахты _____ И.А. Топоркова
«___» _____ 2019 г.

Рецензенты:

ст.преподаватель каф. Информатика
ИСОиП (филиал) ДГТУ в г. Шахты, к.ф.н. _____ О. С. Бурякова
«___» _____ 2019 г.

Преподаватель высшей категории
КЭС ИСОиП _____ Л.В. Завгородняя
«___» _____ 2019 г.

Теория алгоритмов: метод. указания по выполнению практических работ для подгот. обучающ. спец. 09.02.03 «Программирование в компьютерных системах»оч. и и заоч. форм обучения / сост.преп И.А. Топоркова : Шахты, 2019. – 44с.

Настоящие методические указания определяют цели и задачи, содержание работ, общие требования к выполнению практических работ, форму отчетов, краткие теоретические сведения.

Данные методические указания предназначены для углубления и закрепления теоретических знаний, полученных обучающимися на уроках теоретического обучения, а также приобретения навыков самостоятельной работы по дисциплине Теория алгоритмов.

Предназначено для обучающихся специальности 09.02.03 «Программирование в компьютерных системах».

Режим доступа к электронной копии печатного издания: http://www.libdb.sssu.ru

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данные методические указания предназначены для обучающихся специальности 09.02.03 Программирование в компьютерных системах.

Методические указания по выполнению практических заданий разработаны в соответствии с требованиями Федерального государственного образовательного стандарта по специальности среднего профессионального образования 09.02.03 Программирование в компьютерных системах с учетом соответствующей учебной основной образовательной программы.

Методические указания могут быть использованы как для проведения практических занятий, так и для индивидуального усовершенствования имеющихся навыков работы с компьютерными программными продуктами.

В методических указаниях приведены 10 практических работ. Для выполнения практических работ необходимы программные среды: ОС Windows, MS Office, Visio.

Задания и вопросы методических указаний соответствуют уровню подготовленности студентов к изучению данной дисциплины.

В методических указаниях определены цели, требования к выполнению заданий и сдаче отчёта, приведены контрольные вопросы для самоподготовки и рекомендованы литературные источники.

Письменный отчет оформляется согласно «Правилам оформления и требованиям, введённым в действие приказом ректора ДГТУ № 227 от 30.12. 2015 года.

1. ОБЩИЕ ПОЛОЖЕНИЯ

Практическое занятие - это занятие, проводимое под руководством преподавателя в учебной аудитории, направленное на углубление теоретических знаний и овладение определенными методами самостоятельной работы. В процессе таких занятий вырабатываются практические умения.

Перед практическим занятием следует изучить конспект лекции и рекомендованную преподавателем литературу, обращая внимание на практическое применение теории и на методику решения типовых ситуаций. На практическом занятии главное – уяснить связь решаемых ситуаций с теоретическими положениями.

Для ведения записей на практических занятиях обычно заводят журнал практических занятий. Логическая связь лекций и практических занятий заключается в том, что информация, полученная на лекции, в процессе самостоятельной работы на практическом занятии осмысливается и перерабатывается, при помощи преподавателя анализируется до мельчайших подробностей, после чего прочно усваивается.

Успешное освоение курса «Теория алгоритмов» предполагает активное, творческое участие обучающихся путем планомерной, повседневной работы, которая позволит:

Знать:

– основные модели алгоритмов;

– методы построения алгоритмов;

– методы вычисления сложности работы алгоритмов.

Уметь:

– разрабатывать алгоритмы для конкретных задач;

– определять сложность работы алгоритмов.

Владеть:

– разработки алгоритмов для решения конкретных задач;

– оптимизации алгоритмов.

Представленные, в данных методических указаниях, практические задания направлены на формирование компетенций:

– ОК-1: Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес

– ОК-2: Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

– ОК-3: Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.

– ОК-4: Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

– ОК-5:.Использовать информационно-коммуникационные технологии в профессиональной деятельности

– ОК-6: Работать в коллективе и в команде, эффективно общаться с коллегами, руководством, потребителями.

– ОК-7: Брать на себя ответственность за работу членов команды (подчиненных), за результат выполнения заданий.

– ОК-8: Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

– ОК-9: Ориентироваться в условиях частой смены технологий в профессиональной деятельности.

– ПК-1.1: Выполнять разработку спецификаций отдельных компонент.

– ПК-1.2: Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля.

Наряду с формированием умений и навыков в процессе практических занятий обобщаются, систематизируются, углубляются и конкретизируются теоретические знания, вырабатывается способность и готовность использовать теоретические знания при решении задач.

При выполнении заданий обучающимся имеют возможность пользоваться лекционным материалом, с разрешения преподавателя осуществлять деловое общение с товарищами.

Оценка компетентности осуществляется следующим образом: по окончании выполнения задания студенты оформляют отчет, который затем выносится на завершающий этап формы изучения дисциплины. В процессе защиты выявляется информационная компетентность в соответствии с заданием на практическое занятие, затем преподавателем дается комплексная оценка деятельности обучающегося.

Задачи:

- подтверждение теоретических положений;
- закрепление нового материала;
- взаимосвязь нового материала с пройденными темами;
- формирование исследовательских умений (наблюдать, сравнивать, анализировать, устанавливать зависимости, делать выводы и обобщения, самостоятельно вести исследование, оформлять результаты);
- обучение навыкам работы с текстом (понимать текст, различать его виды, анализировать содержащуюся в тексте информацию, делать выводы, различать точки зрения);
- формирование навыков работы в группе;
- обучение формулированию и аргументации своего мнения.

Требования к оформлению практических работ:

- цель работы;
- оснащение (оборудование, материалы и др.);

- практическая часть (порядок выполнения);
- выводы по работе;
- источники (литература);
- форма отчета практической работы (приказ № 227, раздел 5).

Пример оформления практической работы показан в Приложении А, Б.

Критерии оценки выполненной работы:

- процент выполнения работы;
- достижение заданного результата;
- правильность выполнения заданий;
- наличие всех элементов работы;
- время выполнения работы.

2. ПРАКТИЧЕСКИЕ РАБОТЫ

Практическая работа №1 «Реализация линейных алгоритмов»

Цель работы: познакомить с основными способами представления алгоритмов и с классификацией и структурой алгоритмических языков. Научиться решать задачи с использованием блок-схем на составление алгоритмов линейной структуры.

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

Алгоритмом называется точное предписание, определяющее последовательность действий исполнителя, направленных на решение поставленной задачи. В роли исполнителей алгоритмов могут выступать люди, роботы, компьютеры.

Понятие алгоритма в программировании является фундаментальным. Для алгоритма важен не только набор определенных действий, но и то, как они организованы, т.е. в каком порядке они выполняются.

1.1 Свойства алгоритма:

- **понятность** - все действия должны входить в систему команд исполнителя, т.е. быть понятны ему;
- **дискретность** - алгоритм делится на отдельные элементарные шаги;
- **определенность** - каждая команда однозначно определяет действие исполнителя;
- **конечность** (результативность) - алгоритм должен завершаться за конечное число шагов.
- **массовость** – алгоритм позволяет решать целый класс похожих задач.

1.2 Способы записи алгоритма:

Словесно-формульный

Пример. Алгоритм деления обыкновенных дробей

1. Числитель первой дроби умножить на знаменатель второй;
2. Знаменатель второй дроби умножить на числитель второй;
3. Записать дробь, числитель которой есть результат выполнения пункта 1, а знаменатель - результат выполнения пункта 2.

Графический способ (в виде блок-схемы)

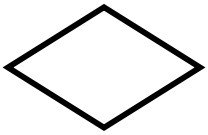
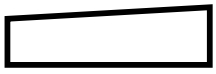
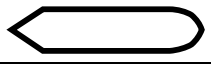
Блок схема – это графическое представление алгоритма при помощи стандартных обозначений. Блок схемы составляются в соответствии с ГОСТами.

ГОСТы алгоритмов: ГОСТ 19.002-80, ГОСТ 19.003-80.

На схемах алгоритмов выполняемые действия изображаются в виде отдельных блоков, которые соединяются между собой линиями связи в порядке выполнения действий. На линиях связи могут ставиться стрелки, причем, если направление связи слева направо или сверху вниз, то стрелки не ставятся.

Внутри блока дается информация о выполняемых действиях. В таблице 1 представлены символы обозначения и их назанчение.

Табица 1 - ГОСТ на составление блок-схем и обозначение блоков

Операция	**Символ**	**Назначение и функции назначение**
Присваивание		Процесс – выполнение операции или группы операций
		Предопределенный процесс – использование ранее разработанного алгоритма как составной части решения задачи, в том числе обращение к подпрограмме
Условный переход		Выбор (Решение), логический блок – определяет выбор направления выполнения алгоритма в зависимости от некоторого условия, записанного внутри блока. Направления обозначаются символами «Да» или «Нет» над каждой выходящей линией
Цикл		Модификация (подготовка) – определяет циклический вычислительный процесс. Внутри записывается заголовок цикла.
Начало, завершение		Пуск – начало, останов - конец
Ввод/вывод		Блок ввода
		Блок вывода
		Ручной ввод – вывод данных с клавиатуры
		Печать (Документ) – вывод, печать информации на бумажный носитель
		Дисплей – вывод информации на экран дисплея (монитора)
		Комментарий – пояснения к операции данного блока
		Соединитель (узел) – указание связи между прерыванием линиями потока
		Линии потока – изображение связи между символами. Линии без стрелки указывает направление потока слева направо или сверху вниз

Правила создания блок-схем:

- Линии, соединяющие блоки и указывающие последовательность связей между ними, должны проводится параллельно линиям рамки.
- Стрелка в конце линии может не ставиться, если линия направлена слева направо или сверху вниз.
- В блок может входить несколько линий, то есть блок может являться преемником любого числа блоков.
- Из блока (кроме логического) может выходить только одна линия.
- Логический блок может иметь в качестве продолжения один из двух блоков, и из него выходят две линии.
- Если на схеме имеет место слияние линий, то место пересечения выделяется точкой. В случае, когда одна линия подходит к другой и слияние их явно выражено, точку можно не ставить.
- Схему алгоритма следует выполнять как единое целое, однако в случае необходимости допускается обрывать линии, соединяющие блоки.

Этапы реализации задачи:

1) Выяснить исходные данные, результаты, назначить им имена.
2) Выбрать метод (порядок) решения задачи
3) Разбить метод решения задачи на этапы (с учетом возможностей ЭВМ)
4) Изобразить каждый этап в виде соответствующего блока схемы алго-ритма и указать стрелками порядок их выполнения.
5) В полученной схеме при любом варианте вычислений:

а) предусмотреть выдачу результатов или сообщений об их отсутствии;

б) обеспечить возможность после выполнения любой операции, так или иначе, перейти к блоку «Конец» (к выходу схемы).

Линейная структура алгоритма.

Линейным называется такой **алгоритм**, все действия которого выполняются однократно и последовательно один за другим.

Пример 1: составить алгоритм обмена значений переменных **a** и **b**.

Схема обмена значений переменных **a** и **b**, с использованием дополнительной ячейки **c**, представлена на рисунке 1.

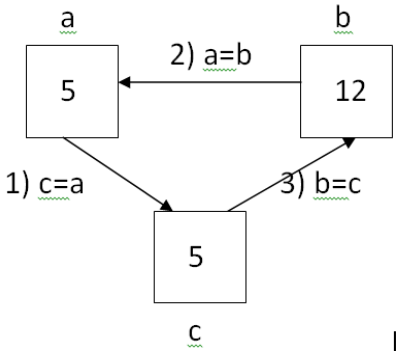
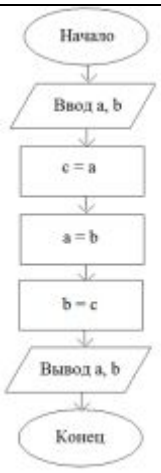


Рисунок 1 – Схема обмена занчений двух переменных

Этапы решения задачи представлены в таблице 2.

Таблица 2 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a и b – переменные Произвести обмен значений этих переменных, используя дополнительную переменную c.
2 этап: формализация	c:=a, a:= b, b:=c
3 этап: блок-схема	

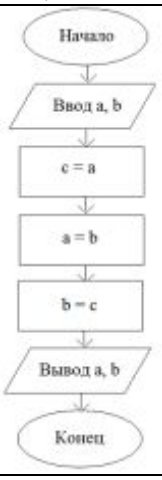
Трассировочная таблица:

Команды	**a**	**b**	**c**
a=5, b=12	5	12	-
c:=a	5	12	5
a:=b	12	12	5
b:=c	12	5	5

<u>Пример 2:</u> составить алгоритм обмена значений переменных a и b без использования дополнительной переменной.

Этапы решения задачи представлены в таблице 3.

Таблица 3 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a и b – переменные Произвести обмен значений этих переменных, без использования дополнительной переменной.
2 этап: формализация	c:=a, a:= b, b:=c
3 этап: блок-схема	

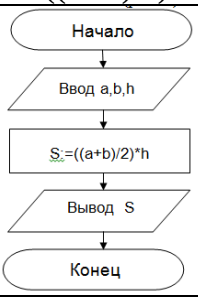
Трассировочная таблица

Команды	a	b
a=3, b=7	3	7
a:=a+b	10	7
b:=a-b	10	3
a:=a-b	7	3

Пример 3: определить площадь трапеции по введенным значениям оснований (a и b) и высоты (h).

Этапы решения задачи представлены в таблице 4.

Таблица 4 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a и b – основание трапеции h – висота Найти площадь трапеции: S - ?
2 этап: формализация	S:=((a+ b)/2)* h
3 этап: блок-схема	

Трассировочная таблица

Команды	a	b	h	S
a=5, b=2, h=8	5	2	8	
S:=((a+ b)/2)* h				28

Практическая работа

1. Решить задачи по этапам.

2. Разработать блок-схемы в программе Microsoft Visio

2. Выполнить проверку алгоритма с помощью трассировочной таблицы.

Задача 1. Стороны прямоугольника a и b. Найдите периметр, площадь и диагональ прямоугольника. Для вычисления диагонали использовать формулу D=SQR(a2+b2). P:=2*(a+b), S := a*b

Задача 2. Стороны параллелограмма a и b, угол между ними x. Определите периметр, площадь, диагонали параллелограмма.

Задача 3. Длины сторон треугольника равны a,b,c. Найдите длины высот треугольника H1,H2,H3. Для определения высот треугольника использовать формулу Ha=2S/a.

Задача 4. Длины сторон треугольника k,l,m. Найдите площадь треугольника, радиус вписанной и радиус описанной окружности. Радиус вписанной окружности определяется по формуле R=S/p, а радиус описанной окружности -Ro=abc/4S.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Какими способами можно представить алгоритм?
2. Каким требованиям должен удовлетворять алгоритм?
3. Как называется способ записи алгоритма в произвольном изложении?
4. Что такое графическое представление алгоритма?
5. Перечислить правила создания блок-схем.
6. Что такое линейный алгоритм?
7. Из каких фигур может быть составлена блок-схема линейного алгоритма?
8. Какими способами можно задать значение переменной?
9. Сколько переменных можно задать в блоке ввода данных?
10. Что означает запись команды: a: = a + 3?

Практическая работа №2 «Составление алгоритмов, содержащих ветвление»

Цель работы: научиться решать задачи на составление алгоритмов разветвляющей структуры

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

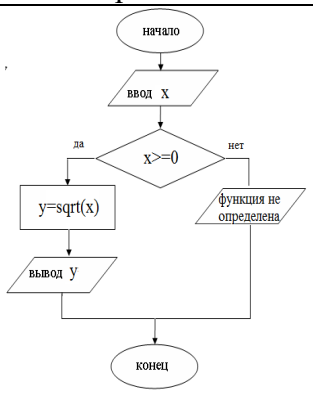
Разветвление (ветвление, развилка) – это такая структура организации действий в алгоритме, когда в зависимости от выполнения или невыполнения некоторого условия выполняется либо одна, либо другая последовательность действий.

Имеется две формы ветвлений – полная, имеющая две ветви и неполная, имеющая одну ветвь. В каждой из них указывается условие, которое надо проверять, и наборы действий, которые надо исполнять при выполнении или невыполнении условия. Ясно, что проверка условия должна быть допустимым действием исполнителя.

Пример 1. Вычислить выражение $y = \sqrt{x}$ для введенного x с клавиатуры. Если x >=0, тогда выполнить вычисления, в противном случае вывести сообщение «функция не определена».

Этапы решения задачи представлены в таблице 1.

Таблица 1 – Этапы решения задачи.

1 этап: постановка задачи	Дано: x – аргумент функции Вычислить y.
2 этап: формализация	Если x >=0, тогда y:=SQRT(x) иначе вывести сообщение «функция не определена»
3 этап: блок-схема	

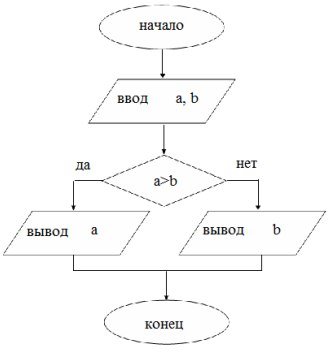
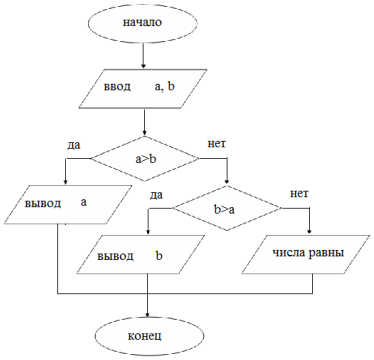
Трассировочная таблица

проверяемый случай	**x>=0**	**y**	**результат**
x=9	9>=0, да	y= $\sqrt{9} = \pm 3$	y= ± 3
x=-9	-9>=0, нет	-	функция не определена

<u>Пример 2.</u> Выбрать максимальное значение из 2х чисел a и b.

Этапы решения задачи представлены в таблице 2.

Таблица 2 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a и b – переменные (вводимые числа) Найти максимальное значение из 2-х чисел
2 этап: формализация	Если $a > b$, тогда a - большее число, вывести на печать, иначе b - большее число, вывести на печать.
3 этап: блок-схема	1 вариат решения  2 вариат решения 

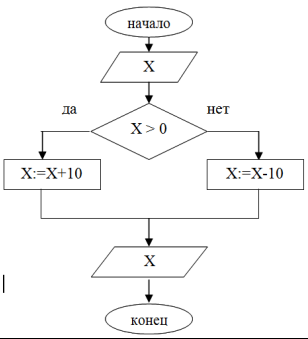
Трассировочная таблица для 2 варианта решения задачи.

проверяемый случай	***a > b***	***b > a***	**результат**
a=9 b=15	9>15 нет	15>9 да	15
a=18 b=3	18>3 да	-	18
a=7 b=7	7>7 нет	7>7 нет	числа равны

Пример 3. Составить блок-схему для решения задачи: дано число *X*. Увеличить его на 10, если оно положительное, во всех остальных случаях уменьшить его на 10.

Этапы решения задачи представлены в таблице 3.

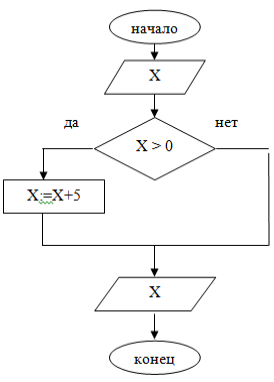
Таблица 3 – Этапы решения задачи.

1 этап: постановка задачи	Дано: x – переменная (вводимое число) Найти максимальное значение из 2-х чисел
2 этап: формализация	Если *x > 0*, тогда *x:=x+10*, иначе *x:=x-10.* Результат вывести на печать.
3 этап: блок-схема	

Пример 4. Составить блок-схему для решения задачи. Дано число *X*. Увеличить его на 5, если оно положительное

Этапы решения задачи представлены в таблице 4.

Таблица 4 – Этапы решения задачи.

1 этап: постановка задачи	Дано: x – переменная (вводимое число) Найти максимальное значение из 2-х чисел
2 этап: формализация	Если *x > 0*, тогда *x:=x+5* Результат вывести на печать.
3 этап: блок-схема	

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio
3. Выполнить проверку алгоритма с помощью трассировочной таблицы.

Задача 1. Ввести 4 числа. Если сумма этих чисел четная, найти произведение, в противном случае, найти частное этих чисел.

Задача 2. Ввести 2-а числа. Вычислить из большего меньшее.

Задача 3. Ввести число. Если оно больше 20, разделить его на 2, если меньше или равно 20, то умножить на 5.

Задача 4. Ввести рост человека. Вывести на экран сообщение «ВЫСОКИЙ», если рост превышает 180 см, в противном случае вывести сообщение « НЕ ОЧЕНЬ ВЫСОКИЙ».

Задача 5. Вычислить значение функции

$$Y = \begin{cases} 2X, \text{ если } X < 0 \\ 2XZ, \text{ если } X = 0 \\ Z, \text{ если } \sin X > 0 \end{cases}$$

Задача 6. Составить блок-схему по следующему алгоритму:

Алг Задача-6
вещ X
нач ввод X
если X < 0
то вывод "отрицательное число"
иначе вывод "положительное число"
кв
кон

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Что такое алгоритм разветвляющей структуры?
2. Какие виды разветвления вы знаете?
3. Что представляет собой условие?
4. Какие два значения может принимать выражение логического типа?
5. Перечислите известные вам операторы сравнения.
6. Какие логические операторы вы знаете? Что они означают и для

чего используются?

7. Приведите пример сложного условия.

8. Как в блок-схеме изображается условие?

9. Можно ли вкладывать условие друг в друга?

Практическая работа №3 «Составление алгоритмов выбора»

Цель работы: научиться решать задачи на составление алгоритмов разветвляющей структуры.

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

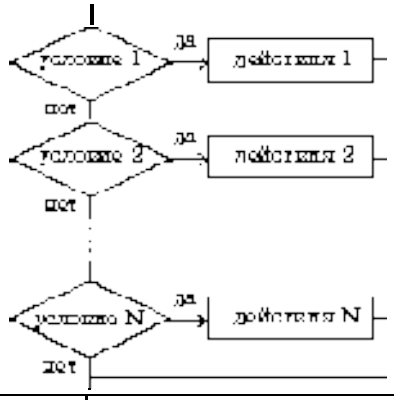
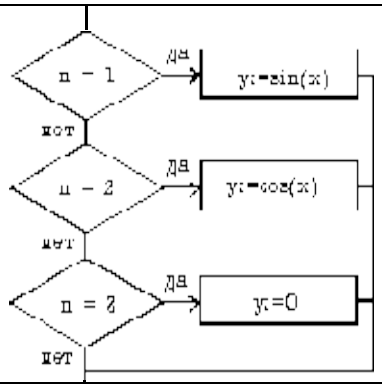
Теоретическая часть

Алгоритмическая структура «выбор» применяется для реализации ветвления со многими вариантами серий команд. В структуру выбора входят несколько условий, проверка которых осуществляется в строгой последовательности их записи в команде выбора. При истинности одного из условий выполняется соответствующая последовательность команд.

В алгоритмической структуре «выбор» выполняется одна из нескольких последовательностей команд при истинности соответствующего условия.

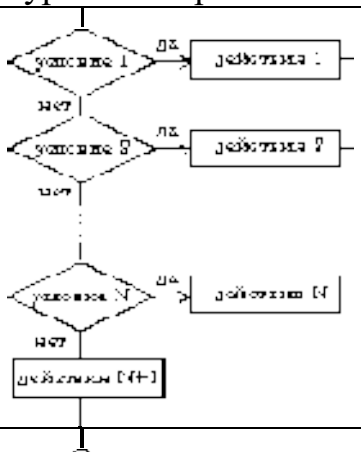
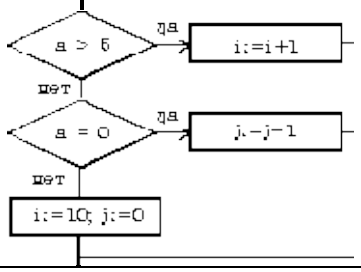
В таблице 1 представлены алгоритмические структуры «выбор» неполной формы.

Таблица 1 - алгоритмические структуры «выбор» неполной формы

выбор **при** условие 1: действия 1 **при** условие 2: действия 2 . . . . . . **при** условие N: действия N **все**	
выбор **при** n = 1: y := sin(x) **при** n = 2: y := cos(x) **при** n = 3: y := 0 **все**	

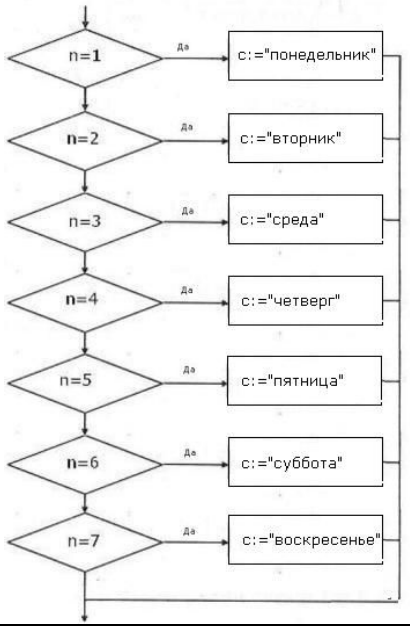
В таблице 2 представлены алгоритмические структуры «выбор» полной формы.

Таблица 2 - алгоритмические структуры «выбор» полной формы

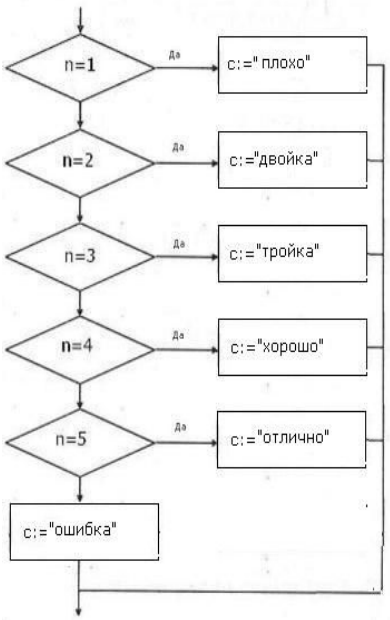
выбор **при** условие 1: действия 1 **при** условие 2: действия 2 . . . . . . **при** условие N: действия N **иначе** действия N+1 **все**	
выбор **при** a > 5: i := i+1 **при** a = 0: j := j+1 **иначе** i := 10; j:=0 **все**	

В таблице 3 представлены примеры решения задач с использованием алгоритмической структуры «выбор».

Таблица 3 – Примеры прешения задач.

Пример 1: Дано целое число в диапазоне 1–7. Составить строку — название дня недели, соответствующее данному числу (1 — «понедельник», 2 — «вторник» и т. д.).	
Алгоритмический язык	***Блок-схема***
Выбор **при** n=1: c:=«понедельник» **при** n=2: c:=«вторник» **при** n=3: c:=«среда» **при** n=4: c:=«четверг» **при** n=5: c:=«пятница» **при** n=6: c:=«суббота» **при**n=7:c:=«воскресенье» **всё**	

Пример 2: Дано целое число n. Составить строку-описание оценки, соответствующей числу n (1 — «плохо», 2 — «двойка», 3 — «тройка», 4 — «хорошо», 5 — «отлично»). Если n не лежит в диапазоне 1–5, то вывести строку «ошибка»

Алгоритмический язык	***Блок-схема***
Выбор - иначе **при** n=1: с:=«плохо» **при** n=2: с:=«двойка» **при** n=3: с:=«тройка» **при** n=4: с:=«хорошо» **при** n=5: с:=«отлично» **иначе** с:=«ошибка» **всё**	

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio
3. Выполнить проверку алгоритма с помощью трассировочной таблицы.

Задача 1. Единицы длины пронумерованы следующим образом: 1 – дециметр, 2 – километр, 3 – метр, 4 – миллиметр, 5 – сантиметр. Дан номер единицы длины и длина отрезка L (в этих единицах вещественное число). Вывести длину данного отрезка в метрах.

Задача 2. Составить алгоритм программы в виде блок-схемы, которая по возрасту человека (вводится с клавиатуры как целое число) определяет его принадлежность к возрастной группе: от 0 до 13 – мальчик; от 14 до 20 – юноша; от 21 до 70 – мужчина; более 70 – старец.

Задача 3. Дан номер месяца (1 – январь, 2 – февраль, ...). Вывести название соответствующего времени года («зима», «весна» и т. д.).

Задача 4. Локатор ориентирован на одну из сторон света («С» – север, «З» – запад, «Ю» – юг, «В» – восток) и может принимать одну из трех цифровых команд: -1 – поворот налево, 2 – поворот направо, 3 – поворот на 180 градусов. Дан символ С – исходная ориентация локатора и число N – посланная ему команда. Вывести ориентацию локатора после выполнения команды.

Задача 5. Составить алгоритм программы в виде блок-схемы, которая по номеру семестра печатает курс, к которому относится введенный семестр (1 и 2 семестр – 1 курс, 3 и 4 семестр – 2 курс и т. д.).

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. В каком случае в алгоритмической структуре «выбор» выполняется последовательность команд Серия 1? Последовательность команд Серия 2?

2. В каком случае можно использовать сокращенную форму алгоритмической структуры «выбор»?

Практическая работа №4 «Составление алгоритмов циклической структуры»

Цель работы: научиться решать задачи на составление алгоритмов в виде блок-схем циклической структуры.

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

Цикл - форма организации действий, при которой одна и та же последовательность действий совершается несколько раз до тех пор, пока выполняется какое - либо условие.

Циклический алгоритм – это алгоритм, содержащий один или несколько циклов.

1. Циклы с предусловием используются тогда, когда выполнение цикла связано с некоторым логическим условием. Структура цикла с предусловием имеет две части: условие выполнения цикла и тело цикла.

На рисунке 1 представлена структура цикла с предуловием.

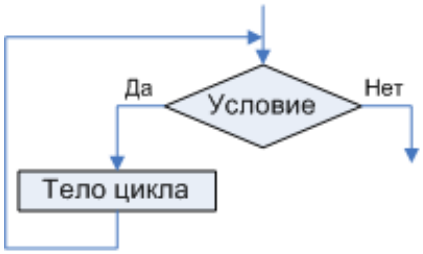


Рисунок 1 - Структура цикла с предуловием

На русском языке это звучит примерно так: пока выполняется это условие, делай от начала группа операторов до конца.

При использовании цикла с предусловием надо помнить следующее:

1. значение условия выполнения цикла должно быть определено до начала цикла;

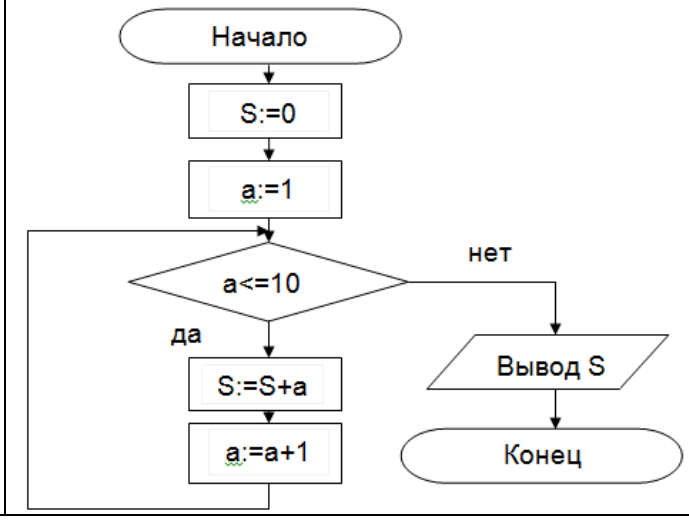
2. если значение условия истинно, то выполняется тело цикла, после чего повторяется проверка условия. Если условие ложно, то происходит выход из цикла;

хотя бы один из операторов, входящих в тело цикла, должен влиять на значение условия выполнения цикла, иначе цикл будет повторяться бесконечное число раз.

Пример 1. Вычислить сумму 10 целых чисел.

Этапы решения задачи представлены в таблице 1.

Таблица 1 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a – переменная (вводимое число) Найти сумму 10 чисел
2 этап: формализация	S:=0; a:=1; S:=S+a; a:=a+1
3 этап: блок-схема	

2. Цикл с постусловием – это цикл, в котором условие проверяется **после** выполнения тела цикла. Отсюда следует, что тело **всегда выполняется** хотя бы один раз.

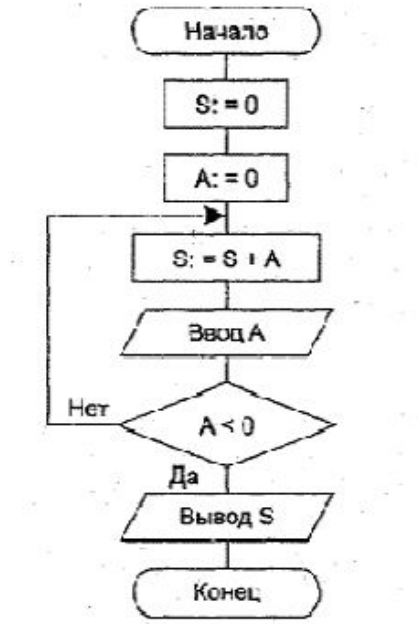
На рисунке 2 представлена алгоритмическая структура цикл с постусловием.

Рисунок 2 - Структура цикл с постусловием

Пример 2. Вычислить сумму чисел, вводимых с клавиатуры. Сумму необходимо подсчитывать до первого введенного отрицательного числа.

Этапы решения задачи представлены в таблице 2.

Таблица 2 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a – переменная (вводимое число) Найти сумму 10 чисел
2 этап: формализация	S:=0; a:=0; S:=S+a
3 этап: блок-схема	

3. Цикл с известным числом повторений (цикл с параметром, цикл типа «Для»).

На рисунке 3 представлена алгоритмическая структура цикл с параметром.

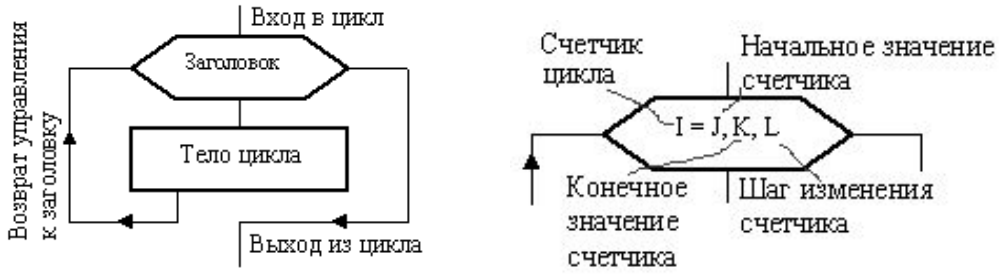


Рисунок 3 - Структура цикл с параметром

Цикл с параметром имеет особенности:

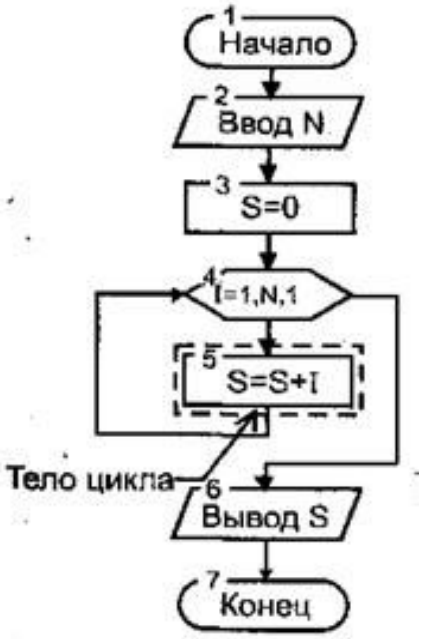
1. в цикле с известным числом повторений параметр изменяется в заданном диапазоне.

2. Если в цикле изменяется простая переменная, то она является параметром цикла; если в цикле изменяется переменная с индексом, то индекс этой переменной является параметром цикла.

Пример 3. Вычислить сумму числового ряда до N.

Этапы решения задачи представлены в таблице 3.

Таблица 3 – Этапы решения задачи.

1 этап: постановка задачи	Дано: a – переменная (вводимое число) Найти сумму числового ряда до n
2 этап: формализация	S:=0; I:=1; S:=S+I
3 этап: блок-схема	

Практическая работа

1. Решить задачи по этапам.

2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Найдите сумму следующей последовательности $a_1+a_2-a_3+a_4-...+a_n$, где n - количество элементов, задаваемых пользователем (при построении блок-схемы использовать структуру цикла с предусловием).

Задача 2. Составить программу нахождения суммы чётных чисел, находящихся в промежутке от 26 до 88 (при построении блок-схемы использовать структуру цикла с предусловием).

Задача 3. Запросить имя пользователя и напечатать "Привет, Вася!" 10 раз (если Вася – имя пользователя), при построении блок-схемы использовать структуру цикла с постусловием.

Задача 4. Найдите все натуральные числа от 1 до 1000, кратные 3 (при построении блок-схемы использовать структуру цикла с постусловием).

Задача 5. Составить таблицу значений функции y = 5x-2 на отрезке [1; 20] с шагом h = 2 (при построении блок-схемы использовать структуру цикла с параметром).

Задача 6. Вычислить произведение нечетных чисел, меньших 16. (при построении блок-схемы использовать структуру цикла с параметром).

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Какой алгоритм называется циклическим?
2. Каковы базовые структуры циклических алгоритмов?
3. Каким образом реализуется цикл с предусловием?
4. Каким образом реализуется цикл с постусловием?
5. В каком из базовых типов цикла возможна ситуация, когда ни разу не выполнится тело цикла?
6. В каком из базовых типов цикла число повторов выполнения тела цикла точно задается заранее?
7. В чем выражается зацикливание алгоритма?
8. В каких базовых типах циклических алгоритмов возможно зацикливание?

Практическая работа №5 «Составление алгоритмов обработки одномерных массивов»

Цель работы: научиться решать задачи на составление алгоритмов в виде блок-схем обработки одномерных массивов

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

Массив – структурированный тип данных, состоящий из фиксированного числа элементов одного типа.

При работе с массивами будем пользоваться следующей терминологией:

Имя массива - один или несколько латинских символов, к которым в зависимости от типа «хранимых» переменных добавляются знаки: **$, %, !.**

Ячейка - место для хранения данных;

Элемент массива - данные хранимые в ячейке массива;

Индекс - номер ячейки;

Размерность - количество индексов.

На рисунке 1 представлен пример одномерного массива.

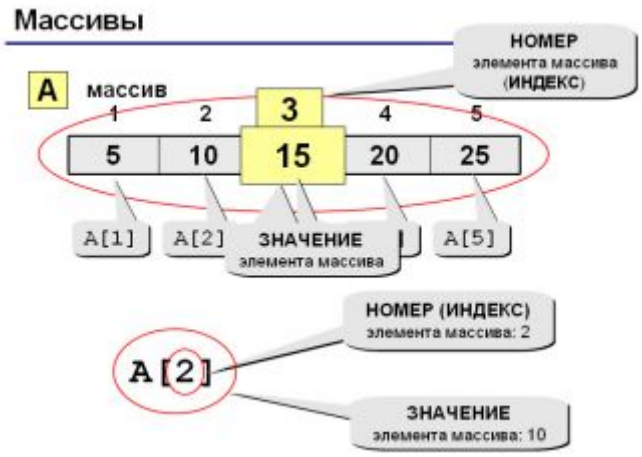


Рисунок 1 – Пример одномерного массива

Действия над элементами осуществляются аналогично действиям над простыми переменными.

Например: A(2) + A(3) = 13,5

Обратите внимание, что действия осуществляются не над индексами, а над числами находящимися в ячейках.

Над элементами массивами чаще всего выполняются такие действия, как поиск, подсчет, перестановка, вставка, удаление элементов в массиве, удовлетворяющих заданному условию, сортировка элементов в порядке возрастания или убывания;

Сумму элементов массива можно подсчитать по формуле s=s+a[i] первоначально задав s=0. Количество элементов массива можно подсчитать по формуле к=к+1, первоначально задав к=0. Произведение элементов массива можно подсчитать по формуле p=p*a[i], первоначально задав p=1.

Пример 1. Ввод-вывод элементов массива представляют собой циклический процесс. Параметром цикла является текущее значение индекса i, изменяющееся в общем случае от начального до конечного значения с шагом 1.

На рисунке 2 представлены фрагменты алгоритмов ввода-вывода элементов массива.

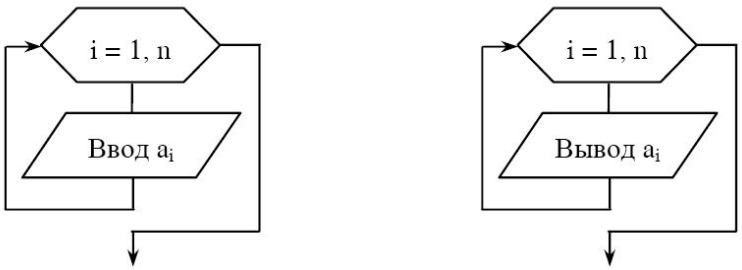


Рисунок 2 - Фрагменты алгоритмов ввода-вывода элементов массива

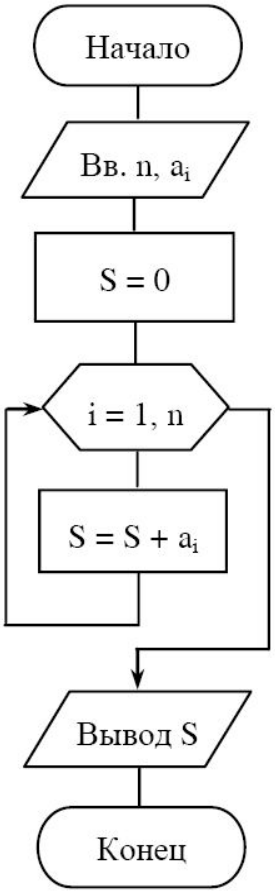
Пример 2. Задан одномерный массив A(n), состоящий из n элемен-

$$S = \sum_{i=1}^{n} a_i$$

тов. Вычислить сумму всех его элементов:

Этапы решения задачи представлены в таблице 1.

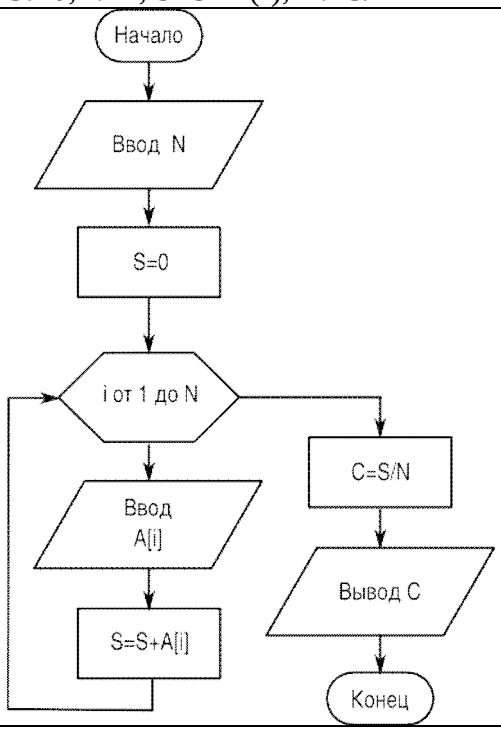
Таблица 1 – Этапы решения задачи.

1 этап: постановка задачи	Дано: A – имя массива A(N) - размер массива A(I) – элемент масива Найти сумму элементов массива, S-?
2 этап: формализация	S:=0; I:=1; S=S+A(I)
3 этап: блок-схема	

Пример 3. Задан одномерный массив A(n), состоящий из n элементов. Вычислить среднеарифметическое значение элементов массива.

Этапы решения задачи представлены в таблице 2.

Таблица 2 – Этапы решения задачи.

1 этап: постановка задачи	Дано: A – имя массива A(N) - размер массива A(I) – элемент масива Найти: сумму элементов массива, S-? среднеарифметическое значение элементов массива, C-?
2 этап: формализация	S:=0; I:=1; S=S+A(I), C:=S/N
3 этап: блок-схема	

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Ввести и вывести элементы массива R размером M.

Задача 2. Вычислить сумму элементов массива K размером T.

Задача 3. Вычислить произведение элементов массива D размером 25.

Задача 4. Вычислить среднеарифметическое значение элементов массива C размером N.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Дайте определение массиву.
2. Как задается размерность массива?
3. Что такое «индекс массива»?
4. Как происходит обращение к элементам массива?
5. Ограниченно ли количество измерений массивов?
6. Какие данные могут выступать в качестве индексов и элементов массива?
7. В чем состоит особенность организации цикла при обработке массива?
8. Какие способы задания исходных значений элементов массива вам известны?
9. Как осуществляется доступ к каждому элементу массива?

Практическая работа №6 «Составление алгоритмов формирования новой последовательности данных»

Цель работы: научиться решать задачи на составление алгоритмов в виде блок-схем по формированию новой последовательности данных

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

В задачах формирования нового массива требуется создать массив из элементов существующего массива (массивов) удовлетворяющих заданному условию. В новый массив элементы заносятся, последовательно начиная с нулевого индекса. Максимально число элементов в формируемом массиве может достигать количества элементов в исходном массиве (массивах), минимальное значение равняется нулю. В этом случаи считается, что новый массив не сформирован.

Для последовательной записи элементов в новый массив используется дополнительная переменная ***k*** - *счетчик элементов в новом массиве*. Начальное значение этой переменной принимается равной *нулю*, т.е. считается, что в новом массиве нет элементов. При обнаружении в исходном массиве элемента удовлетворяющего заданному условию, его значение заносится в новый массив на позицию ***k***, а после счетчик элементов увеличивается на единицу (***k=k+1***). Таким образом, после обработки всего исходного массива по значению счетчика ***k*** можно определить, сформирован новый массив (***k>0***) и сколько в нем элементов (***k***).

Пример 1. Формирование нового массива из элементов заданного массива, удовлетворяющих заданному условию, и подсчет их количества.

Требуетсяиззаданногомассива A(n), состоящегоиз n элементов, выбрать элементы, удовлетворяющие заданному условию, A(I) ≤ T исформи-

роватьизних массив B. Учитывая, что не все элементы массива A войдут в массив B, необходимо выполнять подсчет количества j элементов, удовлетворяющих заданному условию. Блок-схема алгоритма приведена на рисунке 1. Особенностью решения этой задачи является то, что индексы элементов массивов A и B несовпадают, поэтому для обозначения индекса элементов массива B используют переменную j, значение которой изменяется на 1 перед занесением в массив B нового значения.

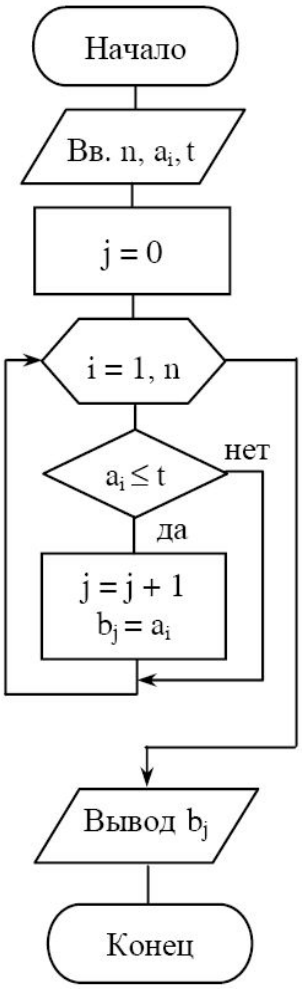


Рисунок 1 - Блок-схема алгоритма формирования нового массива удовлетворяющего условие

Пример 2. Объединение двух массивов в один с чередованием исходных элементов.

Требуется объединить два заданных массива A(n) и B(n), содержащих по n элементов, в один массив C(2·n), который будет содержать 2n элементов, т.е. получить массив C = {a1, b1, a2, b2, …,an, bn}.

Индекс элемента массива C зависит от индекса пересылаемого в него элемента массива A или B следующим образом: C(2·i–1) = A(i), C(2·i) = B(i), i = 1, 2, 3, …, n.

Блок-схема алгоритма объединения двух массивов с чередованием исходных элементов приведена на рисунке 2.

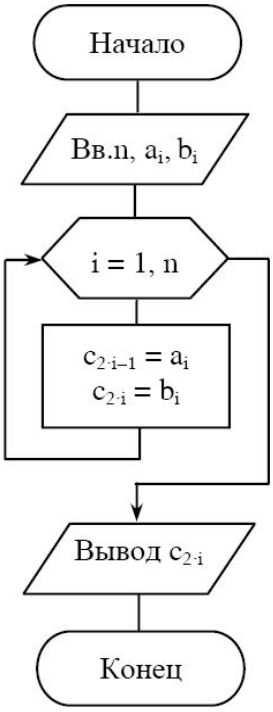


Рисунок 2 - Блок-схема алгоритма объединения двух массивов

Пример 2. Необходимо найти все чётные элементы в массиве A и поместить их в массив B.

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Дан массив Z. Сформировать новый массив из отрицательных значений элементов массива.

Задача 2. Сформировать новый массив L из элементов заданного массива N, удовлетворяющих условию: если значение элемента N[i]<5, и подсчитать их количество.

Задача 3. Сформировать новый массив G из элементов заданного массива P в следующей последовательности: положительные значения элементов, затем отрицательные значения элементов.

Задача 4. Объединить 2-а заданных массива K и N, состоящих из 6-ти элементов в один массив.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.

3. Задание и его решение, скриншоты.
4. Вывод по работе.

Практическая работа №7 «Составление алгоритмов поиска данных в массивах»

Цель работы: научиться решать задачи на составление алгоритмов в виде блок-схем по поиску данных в массивах

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

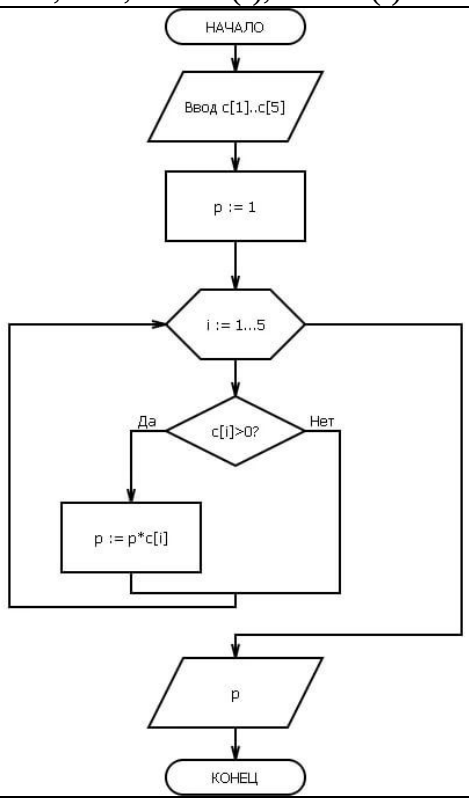
Теоретическая часть

При работе с элементами массива часто приходится изменять значения некоторых элементов, а так же переставлять элементы в массиве. При перестановке необходимо вводить дополнительную переменную.

Пример 1. Дан одномерный массив С(5). Составьте блок схему алгоритма вычисления произведения для положительных элементов массива С.

Этапы решения задачи представлены в таблице 1.

Таблица 1 – Этапы решения задачи.

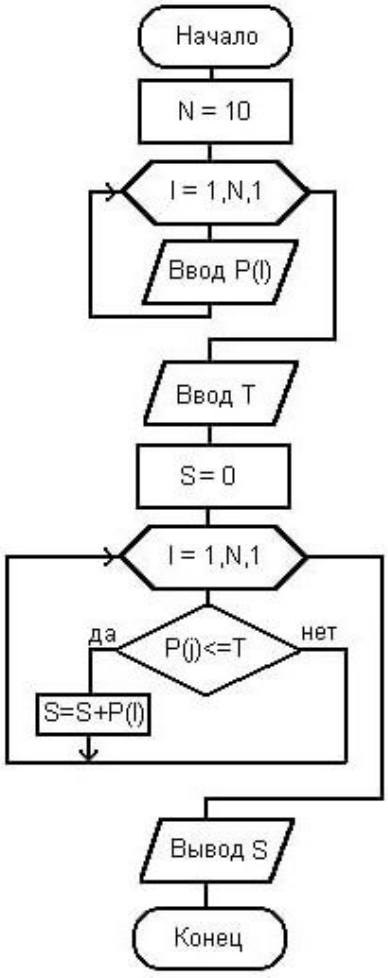
1 этап: постановка задачи	Дано: С – имя массива С(5) - размер массива С(I) – элемент масива Найти: произведение положительных элементов массива, Р-?
2 этап: формализация	P:=1; I:=1; P=P*C(I), если C(I)>0
3 этап: блок-схема	

Пример 2. Требуется определить, сколько элементов заданного массива P, содержащего n элементов, удовлетворяет заданному условию (для определенности пусть условие имеет вид Pi>T, где T - заданное число).

Указание к решению задачи. Для решения задачи следует организовать цикл по i и для каждого значения i проверять условие Pi>T. Если условие удовлетворяется, то к счетчику числа элементов прибавляется 1, а если не удовлетворяется (т.е. Pi≤T), осуществляется переход к следующему элементу массива.

Этапы решения задачи представлены в таблице 2.

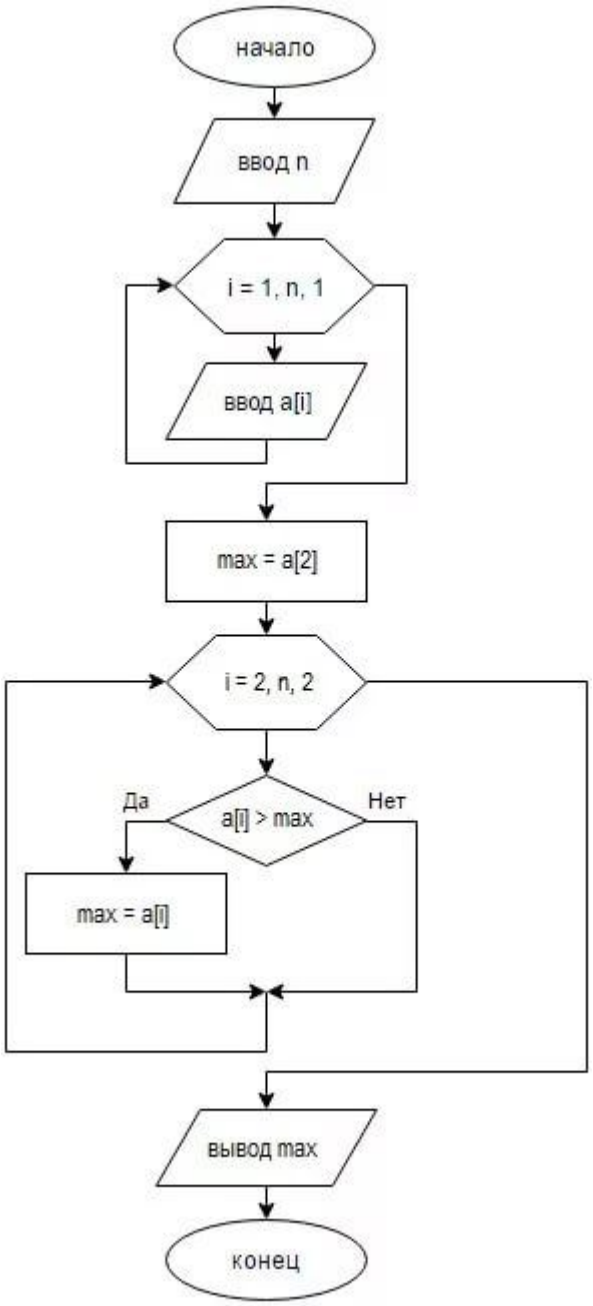
Таблица 2 – Этапы решения задачи.

1 этап: постановка задачи	Дано: P – имя массива; P(N) - размер массива; P[I] – элемент масива; T - заданное значение, с которым сравниваются элементы массива. Найти: элементы массива удовлетворяющие условие P[I] ≤T и подсчетать их сумму, S-?
2 этап: формализация	S:=0; I:=1; S=S+P[I], если P[I]<=T
3 этап: блок-схема	

Пример 3. Дан массив A. Найти максимальный элемент среди четных элементов массива.

Этапы решения задачи представлены в таблице 3.

Таблица 3 – Этапы решения задачи.

1 этап: постановка задачи	Дано: A – имя массива; A(N) - размер массива; A[I] – элемент масива; Найти: максимальный элемент, max -?
2 этап: формализация	max:= A[2], Если A[I]> max, тогда max:= A[I]
3 этап: блок-схема	

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Дан массив D размером R. Найти максимальное и мини-

мальное значение элементов массива.

Задача 2. Дан целочисленный массив. Найти и подсчитать количество отрицательных, положительных и кратных 6-ти значений элементов массива.

Задача 3. Дан действительный массив B. Найти и вывести на дисплей четные элементы массива.

Задача 4. Дан массив. Найти элементы массива кратные 3 и вычислить их сумму.

Задача 5. Дан массив температуры воздуха за месяц. Найти самый теплый день месяца и подсчитать количество холодных дней.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Что такое массив?
2. Какие массивы вы знаете?
3. Какие способы формирования (заполнения) массивов вы знаете?
4. Как обратиться к элементу массива?
5. Этапы обработки массива?
6. Дайте определение матрице.
7. Как задается размерность двухмерного массива?
8. Что такое «индекс матрицы»?
9. Как происходит обращение к элементам двухмерного массива?
10. Какие данные могут выступать в качестве индексов и элементов матрицы?
11. В чем состоит особенность организации цикла при обработке матрицы?
12. Как осуществляется доступ к каждому элементу матрицы?

Практическая работа №8 «Составление алгоритмов сортировки данных в массивах»

Цель работы: научиться решать задачи на составление алгоритмов в виде блок-схем сортировки одномерных массивов.

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

Сортировка – упорядочение данных определенным образом. Такие задачи встречаются часто потому, что разбираться в отсортированных данных легче. Пример такой сортировки – упорядочение ФИО студентов в классном журнале по алфавиту, распределение абитуриентов по набранному баллу и т. д.

Обычно говорят о сортировке записей по ключам – фрагментам этих записей, допускающих отношение упорядочения. Ключи могут быть

1 числами (используется естественный порядок возрастания и убывания);

2 строками (в этом случае упорядочение идет по алфавиту – по кодам ASCII).

Сортировка пузырьковым методом является наиболее известной. Ее популярность объясняется запоминающимся названием, которое происходит из-за подобия процессу движения пузырьков в резервуаре с водой, когда каждый пузырек находит свой собственный уровень, и простотой алгоритма.

Сортировка методом «пузырька» использует метод обменной сортировки и основана на выполнении в цикле операций сравнения и при необходимости обмена соседних элементов. Рассмотрим алгоритм пузырьковой сортировки более подробно.

Сравним первый элемент массива со вторым, если первый окажется больше второго, то поменяем их местами, как показано на рисунке 1. Те же действия выполним для второго и третьего, третьего и четвертого, i–го и (i+1)–го, (n–1)–го и n–го элементов. В результате этих действий самый большой элемент станет на последнее (n-е) место. Теперь повторим данный алгоритм сначала, но последний (n-й) элемент, рассматривать не будем, так как он уже занял свое место. После проведения данной операции самый большой элемент оставшегося массива станет на (n–1)-е место. Так повторяем до тех пор, пока не упорядочим весь массив.

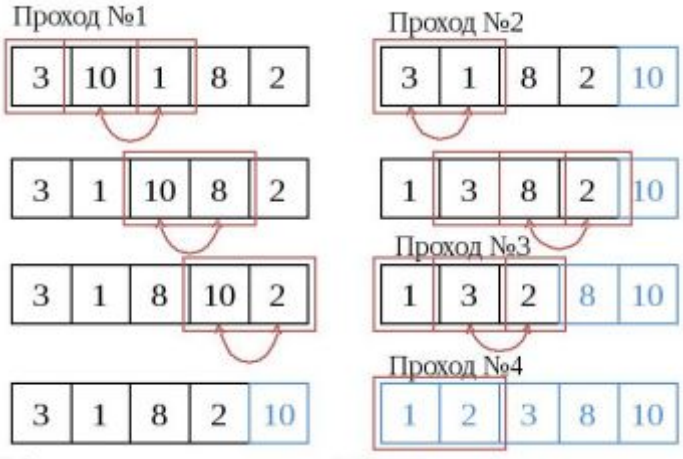


Рисунок 1 – Сортировка массива методом «Пузырька»

Пример процесса упорядочивания элементов в массиве по возрастанию.

В таблице 1 подробно расписан процесс упорядочивания элементов в массиве. Для преобразования массива, состоящего из n элементов, необходимо просмотреть его n–1 раз, каждый раз уменьшая диапазон просмотра на один элемент.

В таблице 1представлен процесс упорядочивания элементов в массиве по возрастанию.

Таблица 1 - Процесс упорядочивания элементов в массиве по возрастанию

Номер элемента	**1**	**2**	**3**	**4**	**5**
Исходный массив	7	3	5	4	2
Первый просмотр	3	5	4	2	7
Второй просмотр	3	4	2	5	7
Третий просмотр	3	2	4	5	7
Четвертый просмотр	2	3	4	5	7

Фрагмент блок–схемы описанного алгоритма приведен на рисунке 2. Обратите внимание на то, что для перестановки элементов (блок 4) используется буферная переменная b, в которой временно хранится значение элемента, подлежащего замене.

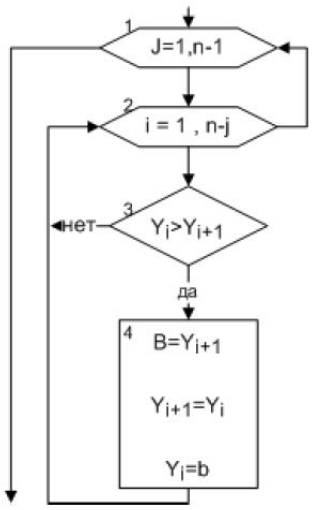


Рисунок 2. – Фрагмент блок-схемы

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Упорядочить элементы массива D размером 25 по возрастанию.

Задача 2. Упорядочить элементы массива F размером K по убыванию.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Что такое сортировка?
2. Что означает термин «упорядочение массива»?
3. В чем заключается суть метода сортировки простым выбором?
4 Перечислите отличительные особенности метода пузырьковой сортировки?
5. Как можно уменьшить количество проходов сортировки при использовании метода пузырьковой сортировки?
6. В чем заключается идея упорядочения массива методом пузырька?

Практическая работа №9 «Составление алгоритмов для работы с двухмерными массивами»

Цель работы: научиться решать задачи на составление алгоритмов в виде блок-схем с двухмерными массивами

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

Матрица – это двумерный массив, каждый элемент которого имеет два индекса: номер строки – i; номер столбца – j. Поэтому для работы с элементами матрицы необходимо использовать два цикла. Если значениями параметра первого цикла будут номера строк матрицы, то значениями параметра второго – столбцы (или наоборот). Обработка матрицы заключается в том, что вначале поочередно рассматриваются элементы первой строки (столбца), затем второй и т.д. до последней.

Рассмотрим основные операции, выполняемые над матрицами при решении задач.

Алгоритм ввода-вывода матриц.

Матрицы, как и массивы, нужно вводить (выводить) поэлементно. Блок-схема ввода элементов матрицы изображена на рисунке 1. Вывод матрицы организуется аналогично вводу.

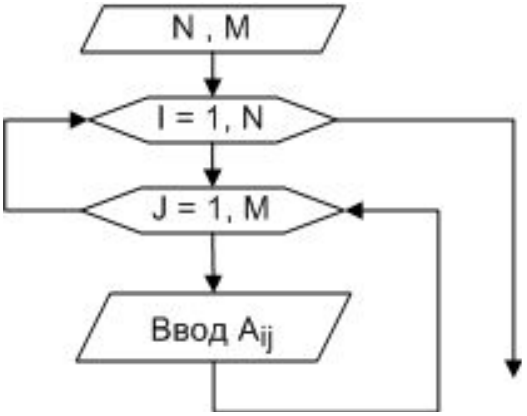


Рисунок 1 – Ввод элементов матрицы

Пример 1. Найти сумму элементов матрицы, лежащих выше главной диагонали, как показано на рисунке 2.

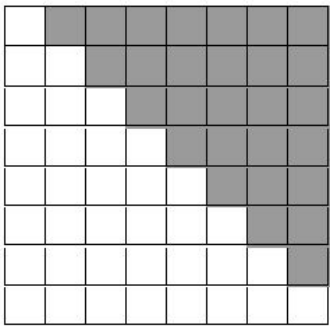


Рисунок 2. Рисунок к условию задачи

Алгоритм решения данной задачи , показанный на рисунке 3, построен следующим образом: обнуляется ячейка для накапливания суммы (переменная S). Затем с помощью двух циклов (первый по строкам, второй по столбцам) просматривается каждый элемент матрицы, но суммирование происходит только в том случае если, этот элемент находится выше главной диагонали, то есть выполняется свойство i<j.

На рисунке 4 изображен еще один вариант решения данной задачи. В нем проверка условия $i<j$ не выполняется, но, тем не менее, в нем так, же суммируются элементы матрицы, находящиеся выше главной диагонали. Для того чтобы понять, как работает этот алгоритм, вернемся к рисунку 2.

В первой строке заданной матрицы необходимо сложить все элементы, начиная со второго.

Во второй – все, начиная с третьего, в i–й строке процесс начнется с (i+1)–го элемента и так далее. Таким образом, первый цикл работает от 1 до N, а второй от i+1 до M. Предлагаем читателю самостоятельно составить программу, соответствующую описанному алгоритму.

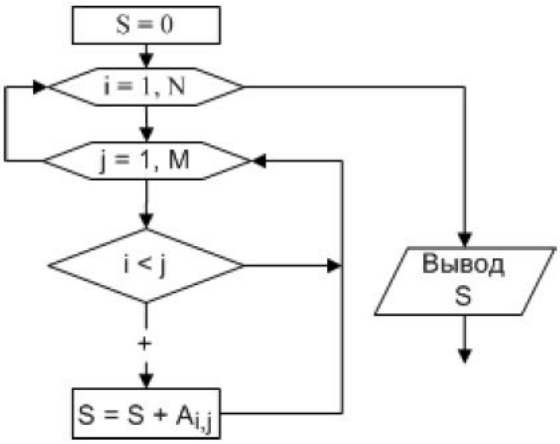


Рисунок 3. Блок-схема примера алгоритма 1

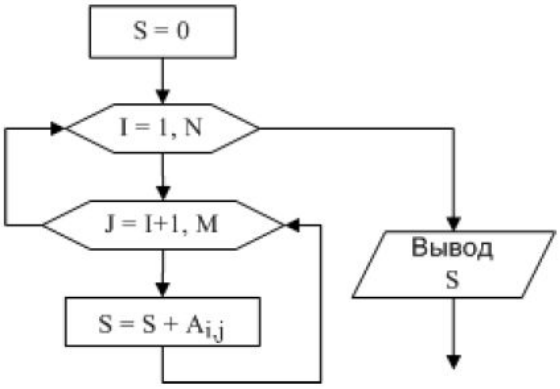


Рисунок 4. Блок-схема примера алгоритма 2

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Вычислить среднее арифметическое, сумму, разность и произведение значений элементов матрицы L.

Задача 2. Найти сумму чисел кратных 5- ти в матрице T целых чисел.

Задача 3. Найти количество положительных и отрицательных элементов матрицы W.

Задача 4. Вывести на экран дисплея элементы матрицы равные 1-му и найти их количество.

Задача 5. В двухмерном массиве, состоящем из NxM вещественных элементов, вычислить сумму отрицательных элементов.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.

3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Дайте определение матрице.
2. Как задается размерность двухмерного массива?
3. Что такое «индекс матрицы»?
4. Как происходит обращение к элементам двухмерного массива?
5. Какие данные могут выступать в качестве индексов и элементов матрицы?
6. В чем состоит особенность организации цикла при обработке матрицы?
7. Как осуществляется доступ к каждому элементу матрицы?

Практическая работа №10 «Алгоритмы поиска и сортировка данных в матрице»

Цель работы:

Оснащение: OS Windows, MS Office, Visio.

Формируемые компетенции: ОК-1 ОК-2 ОК-3 ОК-4 ОК-5 ОК-7 ОК-8 ОК-9 ПК-1.1

Теоретическая часть

Сортировка данных – это обработка информации, в результате которой ее элементы (записи) располагаются в определенной последовательности в зависимости от значения некоторых признаков этой информации.

Сортировка данных позволяет сократить во много раз продолжительность решения задач, которые связаны с обработкой больших массивов информации. Когда элементы отсортированы, как в телефонном справочнике, их проще найти, обновить, исключить, легче отыскать, какие элементы пропущены.

Смысл любой сортировки заключается в перестановке элементов последовательности в определенном заданном порядке. Упорядочение осуществляется в процессе многократного просмотра исходного массива.

В зависимости от того, где выполняется сортировка, во внутренней оперативной памяти компьютера или на внешних носителях данных, различают методы внутренней и внешней сортировки.

Не существует алгоритма сортировки универсально наилучшего в любой ситуации. Имеется много наилучших способов, но только в случаях, когда известно, что́ сортируется, на каком компьютере и с какой целью. Эффективность алгоритма будет зависеть от множества факторов:

- сколько элементов участвует в сортировке;
- в какой степени элементы уже отсортированы;
- какой диапазон значений и распределение сортируемых элементов;

• предполагается ли, что элементы будут периодически исключаться или дополняться;

• можно ли сравнивать элементы параллельно.

Метод сортировки является устойчивым, если относительный порядок элементов с равными значениями не меняется после упорядочения.

Для оценки алгоритмов сортировки обычно используют функциональную зависимость времени от количества N сортируемых элементов.

При разработке алгоритмов рекомендуется выводить на печать исходные данные с соответствующими пояснениями, что позволяет повысить наглядность решения задачи.

Сортировка данных в матрицах производится аналогично сортировки в одномерныз массивах.

Практическая работа

1. Решить задачи по этапам.
2. Разработать блок-схемы в программе Microsoft Visio

Задача 1. Произведите сортировку двумерного массива по убыванию методом перебора.

Задача 2. Произведите упорядочивание элементов каждой строки матрицы по возрастанию любым методом.

Содержание отчета

Отчет должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение, скриншоты.
4. Вывод по работе.

Контрольные вопросы

1. Может ли массив содержать один элемент?
2. Может ли массив не содержать ни одного элемента?
3. Можно ли во время выполнения программы изменить размер массива (количество элементов в нем)?
4. Могут ли элементами массива быть числа: 1, 1.41, 1.78, 2?
5. Верно ли, что тип элементов массива может быть любым?
6. Может ли типом индекса массива быть тип integer?
7. Может ли типом индекса массива быть тип real?

СПИСОК ОСНОВНЫХ ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ

1. Федеральный Государственный образовательного стандарта среднего профессионального образования по специальности 09.02.03 Программирование в компьютерных системах. Утв.приказом Министерства образования и науки РФ от 28 июля 2014 г. N 804. - URL: http://base.garant.ru/70731880/ (дата обращения: 01.09.2018).

2. Правила оформления и требования к содержанию курсовых проектов (работ) и выпускных квалификационные работ. [Текст] : дата введ. 2015-05-19 / ДГТУ. - Изд. офиц. - Ростов : ДГТУ, 2015. - 83 с. - (Система менеджмента качества. Учебно-методическое обеспечение).

3. Постановление Государственного комитета СССР по стандартам от 18 декабря 1978 г. N 3350 «ГОСТ 19.401-78. Единая система программной документации. Текст программы. Требования к содержанию и оформлению» (ред. январь 2010 г.) с Изменением N 1, утвержденным в марте 1983 г. (ИУС 7-83) http://docs.cntd.ru/document/gost-19-401-78-espd (дата обращения: 11.09.2018)

4. Постановление Государственного комитета СССР по стандартам от 11 декабря 1979 г. N 4753 «ГОСТ 19.404-79 ЕСПД. Пояснительная записка. Требования к содержанию и оформлению» (ред. январь 2010 г.) http://docs.cntd.ru/document/1200007671 (дата обращения: 11.09.2018)

5. Постановление Государственного комитета СССР по стандартам от 18 декабря 1978 г. N 3350 «ГОСТ 19.402-78 ЕСПД. Описание программы» (ред. январь 2010 г.) с Изменением N 1, утвержденным в сентябре 1981 г. (ИУС 11-81) http://docs.cntd.ru/document/1200007652 (дата обращения: 11.09.2018)

6. Постановление Государственного комитета СССР по управлению качеством продукции и стандартам от 26.12.90 N 3294 «ГОСТ 19.701 ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения» (ред. январь 2010 г.) http://docs.cntd.ru/document/9041994 (дата обращения: 11.09.2018)

7. Постановление Государственного комитета СССР по стандартам от 18 декабря 1978 г. N 3351 «ГОСТ 19.201-78 ЕСПД. Техническое задание. Требования к содержанию и оформлению» (ред. январь 2010 г.) с Изменением N 1, утвержденным в июне 1981 г. (ИУС 9-81) http://docs.cntd.ru/document/1200007648 (дата обращения: 11.09.2018)

8. Постановление Государственного комитета СССР по стандартам от 12 января 1979 г. N 74 «ГОСТ 19.505–79 ЕСПД. Руководство оператора. Требования к содержанию и оформлению» (ред. январь 2010 г.) с Изменением N 1, утвержденным сентябре 1981 г. (ИУС 11-81) http://docs.cntd.ru/document/gost-19-505-79-espd (дата обращения: 11.09.2018)

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Основная литература

1. Теория алгоритмов: учеб. пособие для подготовки обучающихся специальности 09.02.03 Программирование в компьютерных системах очной и заочной форм обучения, Шахты: ИСОиП (филиал) ДГТУ в г. Шахты, 2019

Перечень ресурсов информационно-телекоммуникационной сети "Интернет"

1. Основы алгоритмизации и программирования (среда PascalABC.NET) : учеб. пособие / И.Г. Фризен. (Среднее профессиональное образование). http://znanium.com/bookread2.php?book=559358 (основная литература)

2. Математическая логика и теория алгоритмов: учебник / А.В. Пруцков, Л.Л. Волкова. — М.: КУРС: ИНФРА-М, 2017. — 152 с. http://znanium.com/bookread2.php?book=559358 (дополнительная литература)

3. Математическая логика и теория алгоритмов: учебник / А.В. Пруцков, Л.Л. Волкова. — М.: КУРС: ИНФРА-М, 2017. — 152 с. http://znanium.com/bookread2.php?book=559358 (дополнительная литература)

ПРИЛОЖЕНИЕ А
(обязательное)
ФОРМА ТИТУЛЬНОГО ЛИСТА

Форма титульного листа

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНСТИТУТ СФЕРЫ ОБСЛУЖИВАНИЯ И ПРЕДПРИНИМАТЕЛЬСТВА (ФИЛИАЛ) ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» В Г. ШАХТЫ РОСТОВСКОЙ ОБЛАСТИ (ИСОиП (филиал) ДГТУ в г. Шахты)

КОЛЛЕДЖ ЭКОНОМИКИ И СЕРВИСА

Журнал практических работ

по дисциплине "Теория алгоритмов".

Выполнил _____ Иванов И.И. группа КВ 9-217
(подпись) (инициалы, фамилия, группа)

Проверил _____ преподаватель И.А. Топоркова.
(подпись) ученая степень, зание, инициалы, фамилия)