Федеральное государственное образовательное бюджетное учреждение  
высшего образования  
«Финансовый университет при Правительстве Российской Федерации»

**КОЛЛЕДЖ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ**

Е.А. Колмыкова

А.В. Семенихина

ПРАКТИКУМ ПО ПРОГРАММИРОВАНИЮ

для студентов специальности

09.02.03 Программирование в компьютерных системах УП.01.01 Учебная практика

Рассмотрен и рекомендован к изданию

на заседании Методического совета колледжа

«\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

Протокол № \_\_\_\_\_\_

Преподаватель \_\_\_\_\_\_\_ /

Преподаватель \_\_\_\_\_\_\_ /

Москва 2015

**ВВЕДЕНИЕ**

Чтобы научиться программировать, не достаточно прочитать книгу, посвященную языку программирования, надо писать программы, решать конкретные задачи и как можно больше.

Учебное пособие «Программирование. Практикум» предназначено для проведения учебной практики по программированию в профессиональном модуле ПМ *01 Разработка программных модулей программного обеспечения для компьютерных систем* специальности 09.02.03 Программирование в компьютерных системах. Объем часов учебной практики составляет 80 часов.

Результатом учебной практики является освоение общих компетенций (ОК):

|  |  |
| --- | --- |
| **Код** | **Наименование результата практики** |
| 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 |
| ОК 10 | Исполнять воинскую обязанность с применением полученных профессиональных знаний |

профессиональных компетенций (ПК**):**

|  |  |  |
| --- | --- | --- |
| **Вид профессиональной деятельности** | **Код** | **Наименование результатов практики** |
| разработка программных модулей программного обеспечения для компьютерных систем | ПК 1.1 | Выполнять разработку спецификаций отдельных компонент |
| ПК 1.2 | Осуществлять разработку кода программного продукта на основе спецификаций на уровне модуля |
| ПК 1.3 | Выполнять отладку программных модулей с использованием специализированных программных средств |
| ПК 1.4 | Выполнять тестирование программных модулей |
| ПК 1.5 | Осуществлять оптимизацию программного кода модуля |
| ПК 1.6 | Разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций |

Учебное пособие состоит из двух частей. В первой части собраны задачи по основным разделам программирования. Они сгруппированы по темам и относятся практически ко всем разделам программирования: от объявления переменных и программ с линейной структурой до работы с графикой и файлами. Каждая тема (общее число тем – 21) представляет собой практическую работу продолжительностью от 2 до 8 академических часов, содержит цель работы, формируемые компетенции, методические рекомендации по решению задач или подходы к решению задач, 12 вариантов заданий и контрольные вопросы.

При создании учебного пособия авторы выдерживали принцип много-вариантности заданий, варианты выровнены по уровню сложности, все темы имеют четкую последовательность: от простого к сложному. Группы задач следуют в пособии в том порядке, в котором традиционно изучаются соответствующие разделы в курсе программирования. Поэтому перед тем как приступить к решению задач, нужно изучить соответствующую тему — прочитать раздел учебника. Если сразу решить задачу не получается, то можно посмотреть ее решение и затем еще раз попытаться решить задачу самостоятельно. Писать программу лучше сначала на бумаге, а уже затем вводить программу в компьютер. Задача считается решенной, если программа работает так, как сказано в условии задачи.

Важно, чтобы решенная задача была правильно оформлена, что предполагает использование:

* осмысленных имен переменных, констант, функций и процедур;
* отступов при записи инструкций;
* комментариев.

Правильно оформленную программу легче отлаживать, кроме того, она производит хорошее впечатление.

Практикум ориентирован на параллельное изучение языков программирования Паскаль и Си.

Во второй части приведены методические материалы и материалы справочного характера: оформление блок-схем, требования к оформлению практической работы по программированию, примеры решения задач по каждой теме, тест и контрольная работа с разобранным решением одной задачи.

Решения задач представленны в виде документированных текстов программ, изучение которых поможет студенту даже в том случае, если задача решена им самостоятельно. В условии задачи зачастую не указывается, какие данные следует определять в программе. Эти вопросы необходимо продумать самостоятельно, так как в ряде случаев подобная информация может служить подсказкой при решении задачи. Кроме того, в большинстве задач нужно предусмотреть выдачу сообщений о нестандартных ситуациях завершения программы. Кроме того, необходимо иметь в виду, что решение задачи обычно предполагает разделение ее на части и предпочтение не отдается «оптимальной программе». Решение, приведенное авторами, как правило, не является единственно возможным. Учебное пособие может быть рекомендовано для самостоятельного изучения отдельных тем.

Дополнительно для закрепления пройденного материала во второй части пособия приведен тест и контрольная работа, которые могут быть использованы для полусеместровой аттестации.

В конце книги представлены краткий перечень понятий, необходимых для решения задач, и список рекомендуемой литературы.

**ЧАСТЬ ПЕРВАЯ**

**ЗАДАЧИ ПО ОСНОВНЫМ РАЗДЕЛАМ ПРОГРАММИРОВАНИЯ**

**Тема 1. Программирование базовых алгоритмических структур: следование**

*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач с линейной структурой.

**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

***Методика выполнения работы.***

Практическая работа включает 2 задачи из различных предметных областей. Для каждой задачи необходимо выполнить следующие действия:

1. Выполнить построение математической модели задачи (формулы)

2. Составить блок-схему алгоритма

3. Написать программу, реализующую составленный алгоритм на языках программирования Паскаль, Си.

Программа должна включать:

* описание данных с типами, допустимыми для решения задачи
* ввод исходных данных с клавиатуры
* вычисления
* вывод результата на экран

4. Подготовить данные для тестирования работы программы.

5. Проверить работоспособность программы на выбранных тестах.

6. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление переменных разных типов и их основные характеристики (необходимый объем ОП);
* правила составления и выполнения арифметических выражений;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

*Вариант №1:*

1. Заданы значения сторон **a, b, c**. Вычислить площадь треугольника по формуле Герона.
2. Самолет делает две посадки для дозаправки. Сколько горючего потребуется для перелета, если при первой посадке было долито 60%, а при второй 70% до полного бака?

*Вариант №2:*

1. Вычислить среднее арифметическое и среднее геометрическое четырех чисел **a, b, c, d**.
2. На борту самолета, летящего по маршруту Москва–Лондон, находится **Х** пассажиров. Три четверти пассажиров имеют билеты второго класса, все остальные – первого. Билет первого класса стоит **В** рублей. Это в два раза больше стоимости билета второго класса.
3. Сколько денег получила авиакомпания от продажи билетов на этот рейс?

*Вариант №3:*

1. Даны два действительных числа. Найти среднее арифметическое этих чисел и среднее геометрическое их модулей.
2. Вычислить кинетическую (*E = mv2/2*) и потенциальную (*Р = mgh*) энергии тела, заданной массы **m**, которое двигается на высоте **h** со скоростью **v**.

*Вариант №4:*

1. Составьте блок-схему для нахождения количества, суммы и произведения всех натуральных чисел, делящихся на 4, на отрезке [16,64].
2. Навальный хочет оклеить обоями длинную стену в своем доме. Длина этой стены равна **А** метрам, а высота **В** метрам. Рулон обоев имеет длину **L** метров и ширину **S** метров. Сколько будут стоить обои для всей стены, сколько потребуется рулонов обоев, если стоимость одного рулона **К** рублей.

*Вариант №5:*

1. Скорость света 299792 км/с. Какое расстояние проходит свет за один час, сутки?
2. Телефонные разговоры с тремя населенными пунктами стоят **С1, С2, С3** коп/мин. Длительность разговора **t1, t2, t3** мин. соответственно. Какие суммы насчитает компьютер к оплате за каждый разговор в отдельности и вместе?

*Вариант №6:*

1. Вычислить длину окружности и площадь круга по известному диаметру.
2. Каждую неделю Саша получает сумму в размере **S** рублей на личные расходы, из них он тратит **Х**% на сладости. Остальные деньги он откладывает для покупки компьютера. Сколько месяцев потребуется Саше копить деньги, если на день рожденье бабушка с дедушкой ему подарили 3000 рублей, а мама с папой – 10000 рублей. Стоимость компьютера – 970$.

*Вариант №7:*

1. Цены на два вида товаров возросли на **Р** процентов. Вывести старые и новые цены.
2. На фабрике «Мойдодыр» при стирке 4 кг белья расходуется 250 г стирального порошка. Определить, сколько пачек стирального порошка будет израсходовано на **K** простыней (вес одной простыни – 500 г), **P** пододеяльников (вес одного пододеяльника – 650 г) и **S** скатертей (вес одной скатерти – 600 г). Вес одной пачки стирального порошка составляет 750 г.

*Вариант №8:*

1. Ввести производительности работы трех труб, которые наполняют бассейн, и время их работы. Сколько воды набрали в бассейн?
2. В банк был положен вклад в размере **N** рублей. Через год он увеличился на 4%, а еще через год – на 6%. Сколько денег будет на счету через 2 года?

*Вариант №9:*

1. Железнодорожный состав ехал **t1** часов со скоростью **V1** км/час, **t2** часов со скоростью **V2** км/час, **t3** часов со скоростью **V3** км/час. Определить длину отрезков пройденных путей с различной скоростью и полный путь
2. Петр Петрович надел новые штаны и сел на только что покрашенную табуретку. На штанах получилось квадратное пятно зеленого цвета. Длина одной стороны пятна **A** см, а площадь его в **N** тысяч раз меньше центральной площади города, в котором живет Петр Петрович. Узнай площадь этой площади.

*Вариант №10:*

1. Найти корни квадратного уравнения, заданного своими коэффициентами (*D > 0*)
2. В банк был положен вклад в размере **Х** рублей. Через год он увеличился на 3%, а еще через год – на 8%. Сколько денег будет на счету через 2 года?

*Вариант №11:*

1. Определить стоимость набора конфет, в который входят:

"Красная шапочка" - 200 г.

"Алые паруса" - 150 г.

"Чародейка" - 100 г.

если известна стоимость этих конфет за 1 кг.

1. Написать программу пересчета величины временного интервала, заданного в минутах и секундах, в величину, выраженную в часах и минутах.

*Вариант №12:*

1. Скорость лодки в стоячей воде **V** км/ч, скорость течения реки **U** км/ч (*U < V*). Время движения лодки по озеру **t1** ч, а по реке (против течения) — **t2** ч. Определить путь **S**, пройденный лодкой.
2. Коммерсант, имея стартовый капитал **k** рублей, занялся торговлей, которая ежемесячно увеличивает капитал на **p** %. Через сколько лет он накопит сумму **S**, достаточную для покупки собственного магазина?

**Тема 2. Программирование базовых алгоритмических структур: ветвление**

*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач с разветвляющейся структурой.

**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

***Методика выполнения работы.***

Практическая работа включает 5 задач из различных предметных областей. Необходимо выполнить три задачи по выбору (задача № 5 обязательна)

Для каждой задачи необходимо выполнить следующие действия:

1. Выполнить построение математической модели задачи (формулы)

2. Составить блок-схему алгоритма

3. Написать программу, реализующую составленный алгоритм на языках программирования Паскаль, Си (по выбору).

Программа должна включать:

* описание данных с типами, допустимыми для решения задачи
* ввод исходных данных с клавиатуры
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

4. Подготовить данные для тестирования работы программы таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма.

5. Проверить работоспособность программы на выбранных тестах.

6. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление переменных разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила составления и выполнения арифметических выражений;
* правила построения условных выражений;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

Для каждой задачи выбирать наиболее подходящий для неё язык программирования из изученных ранее.

Задачу №5 решать с использованием конструкции выбора.

*Вариант №1:*

1. Даны радиусы двух окружностей. Их центры совпадают. Вычислить площадь кольца между ними.
2. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа. Если все числа отрицательны и не равны между собой, то замените числа их квадратами и выведите результат на экран. Если все числа равны нулю, то сделайте их соответственно равными 5,..,20, 15 и выведите результаты. Иначе, найдите и выведите сумму этих чисел.
3. Введите три числа. Если они могут быть длинами сторон прямоугольного треугольника, выведете их в порядке возрастания и вычислите площадь полученного треугольника.
4. Пусть даны координаты трех точек на плоскости. Если они могут быть вершинами равнобедренного тупоугольного треугольника, вычислите его площадь. Выведите длины сторон и площадь в порядке возрастания значений.
5. Фигуры прямоугольник, треугольник, трапеция, круг пронумерованы цифрами от1 до 4. Введите необходимые исходные данные для каждой фигуры. Вычислите площадь геометрических фигур:

a\*b если n=1

a\*h/2 если n=2

(a+b)\*h/2 если n=3

ΠR2 если n=4

*Вариант №2:*

1. По номеру **Y** (*Y > 0*) некоторого года определить **C** - номер его столетия (учесть, что, к примеру, началом **ХХ** столетия был 1901, а не 1900 год).
2. В школу танцев принимаются юноши и девушки, имеющие рост не ниже 168 см и не выше 178 см. Их вес должен соотноситься с ростом по формуле: *ВЕС <= РОСТ – 115*. Определите будет ли принят поступающий в школу.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры два числа **A** и **B**. Если числа не равны, то замените каждое из них большим и выведите результаты на экран. Если они равны, то найдите и выведите удвоенную сумму этих чисел.
4. Введите три числа. Если они могут быть длинами сторон тупоугольного треугольника, выведете их в порядке возрастания и вычислите площадь полученного треугольника.
5. Составить блок-схему, в которой с учетом роста человека определяется его оптимальный вес и рекомендации о необходимости поправиться или похудеть (оптимальный вес вычисляется: рост - 100).

*Вариант №3:*

1. Построить алгоритм вычисления сдельной заработной платы, а так же налога и суммы выдачи на руки. Зарплата работника определяется числом обработанных деталей, умноженным на стоимость обработки. Если работник допустил брак и испортил деталь, ее стоимость вычитается из заработка. Налог исчисляется в зависимости от зарплаты, если она выше 5000 – налог 20%, если нет 13%. Может случиться, что брака окажется очень много, зарплата станет отрицательной (работник должен компенсировать нанесенный ущерб). В этом случае никакой налог не удерживается.
2. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, C**. Если A > B > C > 0, то замените все числа их квадратами и выведите результат на экран. Если они отрицательны и не равны между собой, то каждое число замените наибольшим из них и выведите результаты. Иначе, найдите и выведите их сумму.
3. Введите три числа. Если они могут быть длинами сторон остроугольного треугольника, выведите их в порядке убывания и вычислите площадь полученного треугольника.
4. Пусть даны координаты трех точек на плоскости. Если они могут быть вершинами равнобедренного остроугольного треугольника, вычислите его площадь. Выведите длины сторон и площадь в порядке возрастания значений.
5. Составить блок-схему, в которой по введенному номеру месяца выдается время года.

*Вариант №4:*

1. Приём на работу идёт на конкурсной основе. Условия приёма требуют 20 лет рабочего стажа и возраст не более 42 лет. Определите, будет ли человек принят.
2. Определить, пройдет ли кирпич с ребрами **a, b, c** в прямоугольное отверстие со сторонами **x** и **y**.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры два числа: **N** и **M**. Если оба числа больше 100, то замените каждое из них меньшим и выведите результаты на экран. Если первое число положительное, а второе отрицательное, то уменьшите их в два раза и выведите результаты. Иначе, найдите и выведите среднее арифметическое этих чисел.
4. Пусть даны три числа. Если они могут быть длинами сторон треугольника, определите его вид (разносторонний, равнобедренный, равносторонний). Вычислите длины его высот и напечатайте их в порядке убывания.
5. Составить блок-схему для задачи, проверяющей знания по предмету (например по алгоритмизации и программированию). В результате выполнения задачи на экран выводится вопрос и три варианта ответа. Пользователь должен выбрать ответ. Далее выводится сообщение «Правильно» или «Вы ошиблись».

*Вариант №5:*

1. Перераспределить значения переменных **X** и **Y** так, чтобы в **X** оказалось большее из этих значений, а в **Y** - меньшее.
2. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **a, b, c**. Если все они равны, то первое увеличьте в 10 раз, второе уменьшите в 10 раз, а третье увеличьте на 10 и выведите результаты на экран. Если равны только первое и последнее, то увеличьте все числа на 100 и выведите результаты. Иначе, найдите и выведите удвоенную сумму этих чисел.
3. Пусть даны координаты трех точек на плоскости. Если они могут быть вершинами остроугольного треугольника, вычислите его площадь.
4. Пусть даны координаты вершин прямоугольника. Определите координаты четвертой вершины.
5. Составить алгоритм, который по введенному числу К выдавал бы соответствующую ему римскую цифру.

*Вариант №6:*

1. Даны координаты верхней левой и нижней правой вершин прямоугольника. Определить, вместится ли в него круг радиуса **R**.
2. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, C**. Если *A = C*, то все числа замените их квадратами и выведите результаты на экран. Если *A > B > C*, то каждое число замените наибольшим и выведите результаты. Иначе, **A** сделайте равным в сумме **B** и **C**, **C** обнулите, **B** оставьте без изменения и выведите результаты.
3. Введите три числа. Если они могут быть длинами сторон равнобедренного треугольника, вычислите длины его высот. Выведите длину основания и длины высот в порядке возрастания.
4. Пусть даны три числа. Если они могут быть длинами сторон треугольника, определите его вид (прямоугольный, тупоугольный, остроугольный). Вычислите его периметр и площадь.
5. Составить блок-схему определения стоимости разговора по телефону с учетом скидки 20%, предоставляемой по субботам и воскресеньям.

*Вариант №7:*

1. Составьте блок-схему и программу для проверки, является ли заданное с клавиатуры число полным квадратом натурального числа.
2. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, C**. Если они не раны, то замените все числа наибольшим из них и выведите результаты на экран. Иначе, найдите и выведите сумму квадратов этих чисел.
3. Введите три числа. Если они могут быть длинами сторон разностороннего тупоугольного треугольника. Выведите их в порядке возрастания и вычислите площадь полученного треугольника.
4. Пусть даны координаты трех точек на плоскости. Если они могут быть вершинами сторон треугольника, определите его вид (разносторонний, равнобедренный, равносторонний). Вычислите длины его высот и напечатайте их в порядке убывания.
5. Составить блок-схему "расписание учебных занятий на неделю" с учетом четности недели.

*Вариант №8:*

1. Даны действительные числа **a, b, c**. Если *a <= b <= c*, то каждое число заменить наибольшим из них; если *a > b > c*, то числа оставить без изменения; в противном случае все числа заменяются их квадратами.
2. Составьте блок-схему и программу для определения, что больше – (*a – 3*) или (*2a – 10*), - в зависимости от введенного с клавиатуры значения **a**.
3. Составьте блок-схему и программу для решения следующей задачи.  
   Введите с клавиатуры три числа: **X, Y, Z**. Если числа равны, то увеличьте их в три раза и выведите результаты на экран. Если раны только два последних числа, то замените все числа на противоположные и выведите результаты. Иначе, все числа сделайте равными 222 и выведите результаты.
4. Пусть даны три числа. Если они могут быть длинами сторон треугольника, определите его вид (прямоугольный, тупоугольный, остроугольный). Найдите углы треугольника (теорема косинусов).
5. Составить блок-схему, которая реализует пользовательское меню (меню - это список, из которого необходимо сделать выбор), пункты которого содержат фамилии Ваших друзей. При выборе некоторой фамилии появляется адрес и телефон.

*Вариант №9:*

1. Даны действительные числа **a, b, c**. Если *a <= b <= c*, то каждое число заменить наименьшим из них; если *a > b > c*, то числа оставить без изменения; в противном случае все числа заменяются их кубами.
2. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры два числа: **K** и **L**. Если они положительны, то замените каждое на квадрат их суммы и выведите результаты на экран. Если все числа отрицательны, то каждое сделайте равным наименьшему и выведите результаты. Иначе, **K** приравняйте к 10, а **L** – к 15 и выведите результаты.
3. Введите три числа. Если они могут быть длинами сторон равнобедренного остроугольного треугольника, вычислите его площадь. Выведите длины сторон и площадь в порядке возрастания значений.
4. Составьте программу для определения вида треугольника (равносторонний, равнобедренный, разносторонний, прямоугольный, тупоугольный, остроугольный) если по данным трем отрезкам его можно построить.
5. Составит блок-схему, вычисляющую стоимость междугороднего телефонного разговора (цена одной минуты определяется расстоянием до города, в котором находится абонент). Исходными данными являются код города и продолжительность разговора.

*Вариант №10:*

1. Построить алгоритм вычисления «Продажа авиабилетов». Известна номинальная цена билета для каждого и трех классов. Общая стоимость заказа формируется путем умножения кол-ва заказанных взрослых билетов на цену билета, плюс число детских билетов на цену билета со скидкой на детей (40%). Кроме того при заказе на сумму более семи полных билетов положена скидка (оплачивается только 80% стоимости заказа).
2. Даны две различные прямые *y = k1\*x + b1* и *y = k2\*x + b2*. Найти точку пересечения данных прямых.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, C**. Если **A** и **C** кратны 5, а **B** – не кратно, то все числа замените их квадратами и выведите результат на экран. Если только **B** кратно 5, то увеличьте все числа на 10 и выведите результаты. Иначе, найдите и выведите среднее арифметическое этих чисел.
4. Пусть даны координаты вершин четырехугольника. Составьте программу, которая определяла бы, является ли этот четырехугольник прямоугольником.
5. Составить блок-схему и программу "Гороскоп".

*Вариант №11:*

1. Составить алгоритм вычисления стоимости основных фондов предприятия с учетом их износа и инфляции. О каждом объекте известны год и цена приобретения. Стоимость каждого объекта уменьшается за счет износа на 10% в год и увеличивается на величину инфляции (положим, средняя инфляция составляет 30% в год). Число лет инфляции в период эксплуатации объекта вычисляется с учетом того, что она возникла только с 2000 года.
2. Составьте блок-схему и программу для определения, что больше – площадь круга радиусом **a** или площадь квадрата со стороной **a**, - значение числа **a** вводится с клавиатуры (площадь круга вычисляется по формуле  
   *S = πR2*).
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры два числа: **a** и **b**. Если оба числа четные, то вычислите их сумму и выведите результат на экран. Если оба – нечетные, то найдите произведение этих чисел и выведите результат. Иначе, увеличьте все числа в пять раз и выведите результаты.
4. Введите три числа. Если они могут быть длинами сторон разностороннего остроугольного треугольника, выведите их в порядке возрастания и вычислите площадь полученного треугольника.
5. Составить блок-схему, которая после введенного с клавиатуры числа (в диапазоне от 1 до 99), обозначающего денежную единицу, дописывает слово «рубль» в правильной форме. Например, 12 рублей, 21 рубль.

*Вариант №12:*

1. Составить алгоритм стоимости автоперевозок заданного веса груза на заданное расстояние разными типами машин. Путевая скорость всех автомашин считается 5 км/ч. Стоимость перевозок состоит из зарплаты водителя и аренды автомобиля. Зарплата водителя определяется временем в пути, кроме того если автомобиль находится в пути больше 12 часов, водителю производится доплата в размере 50 руб. за каждые 12 часов пути. Исходными данными для решения задачи являются: тип автомобиля; цена часа аренды; грузоподъемность; объем заказа; вес груза; расстояние до пункта назначения.
2. Составьте блок-схему и программу для проверки, что больше: *a3* или *(2\*a)2*, - для числа **a** вводимого с клавиатуры.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, С**. Если все они четные, то найдите и выведите их суммы на экран. Если все они нечетные и *A > B > C*, то определите и выведите значения наибольшего и наименьшего из чисел. Иначе, выведите числа без изменения.
4. Введите три числа. Если они могут быть длинами сторон равнобедренного остроугольного треугольника, вычислите его площадь. Выведите длины сторон и периметр в порядке убывания значений.
5. Для целого числа **К** вывести фразу "мне **К** лет", учитывая при этом, что при некоторых значениях **К** слово "лет" надо заменить на слово "год" или "года".

*Вариант №13:*

1. Даны координаты центра круга и его радиус, а также координаты точки. Лежит ли эта точка внутри круга?
2. Составьте блок-схему и программу для проверки, делится ли заданное с клавиатуры число на 4 и 8 одновременно.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, C**. Если **A** и **C** кратны 7, а **B** – не кратно, то все числа замените их кубами и выведите результат на экран. Если только **B** кратно 7, то увеличьте все числа на 5 и выведите результаты. Иначе, найдите и выведите среднее арифметическое этих чисел.
4. Пусть даны координаты трех точек на плоскости. Если они могут быть вершинами равностороннего треугольника, вычислите его площадь и длину высоты. Выведите длины сторон, площадь и длину высоты в порядке возрастания значений.
5. Для целого числа **К** от 1 до 9 напечатать фразу "температура – **К** градусов", согласовав окончание слова "градус" с числом **К**.

*Вариант №14:*

1. Составить алгоритм расчета заработной платы с учетом прогрессивного подоходного налога и числа детей. Положим, с месячного дохода до 2000 налог не удерживается, от 2000 удерживается 12%, от 4000 - 15%, от 6000 - 20%. Налог удерживается не с всей суммы заработка, а с величины, меньшей на одну минимальную зарплату и за вычетом за каждого ребенка по min зарплате.
2. Составьте блок-схему и программу для нахождения минимального из трех чисел, введенных с клавиатуры.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры два числа: **a** и **b**. Если оба числа нечетные, то вычислите их произведение и выведите результат на экран. Если оба – четные, то найдите среднее арифметическое этих чисел и выведите результат. Иначе, увеличьте все числа в три раза и выведите результаты.
4. Введите три числа. Если они могут быть длинами сторон прямоугольного треугольника, выведете их в порядке убывания и вычислите площадь полученного треугольника.
5. Вводится падеж. Напечатать слово «Дверь» в заданном падеже.

*Вариант №15:*

1. Вводятся оценки за контрольные работы по физике и математике. Выведите на экран фразы: "МОЛОДЕЦ", если их сумма равна или больше 9, в противном случае выведите "ПОДТЯНИСЬ".
2. Составьте блок-схему и программу для определения, является ли значение функции *y = (a - b)/(a + b)* целым числом, значения чисел **a** и **b** вводятся с клавиатуры.
3. Составьте блок-схему и программу для решения следующей задачи. Введите с клавиатуры три числа: **A, B, C**. Если **A** и **C** четные, а **B** – нет, то найти среднее арифметическое четных чисел, если четное только одно число из трех – посчитать сумму всех трех чисел. Иначе, найти максимальное число. Выведите результаты.
4. Пусть даны координаты трех точек на плоскости. Если они могут быть вершинами разностороннего тупоугольного треугольника, вычислите его площадь.
5. Для целого числа **К** от 1 до 9 напечатать фразу "мы нашли **К** грибов в лесу", согласовав окончание слова "гриб" с числом **К**.

**Тема 3. Программирование базовых алгоритмических структур: цикл**

*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач с циклической структурой, построения рекурсивных формул для итерационных циклов.

**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

***Методические рекомендации по решению задач***

***Пример 1.*** Дано натуральное число **N**. Определить самую большую цифру числа и ее позицию.  
           Например для числа *N = 12345* (наибольшей является цифра 5, ее позиция - пятая слева).

Входные данные: **N** - целое число.

Выходные данные:

**max** - значение наибольшей цифры в числе,

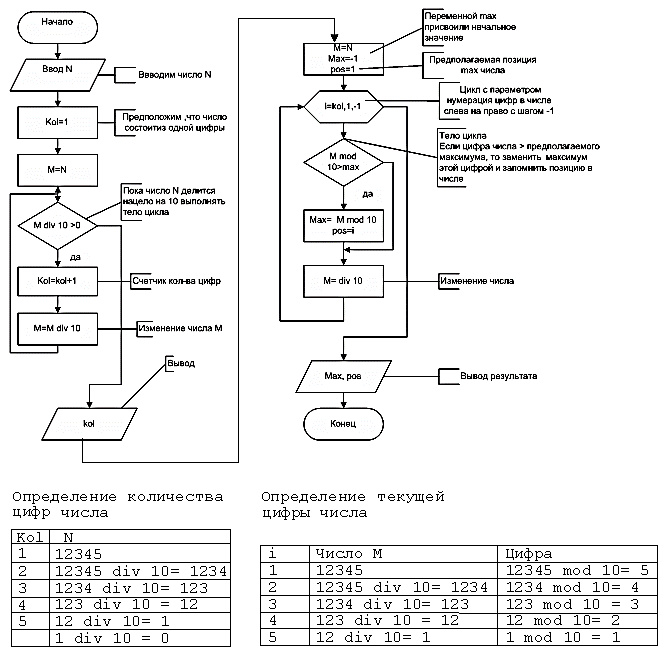
**pos** - позиция этой цифры в числе.

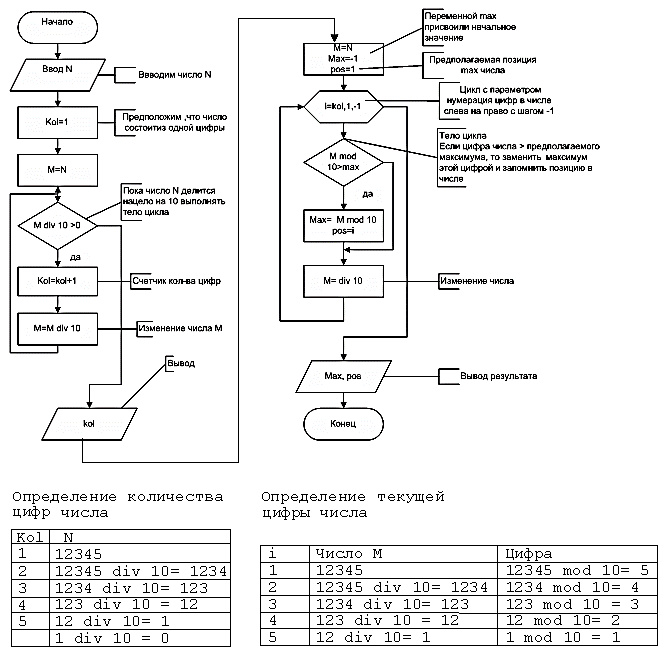
Промежуточные данные:

**i** - параметр цикла,

**kol** - количество цифр в числе,

**М** - переменная для временного хранения значения **N**.

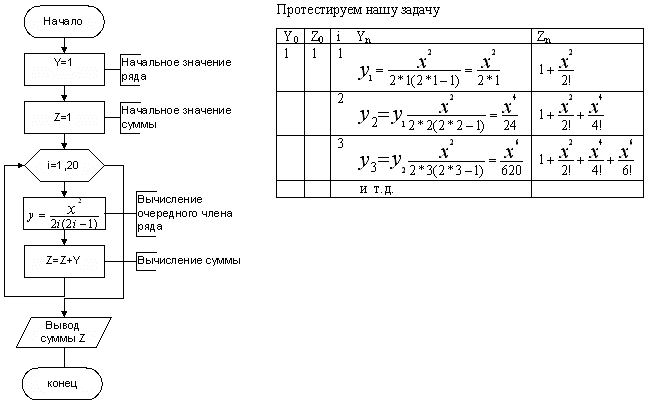
           Разобьем решение этой задачи на два этапа. Вначале найдем количество цифр в заданном числе, а затем определим наибольшую цифру и ее позицию.  
           Для определения кол-ва цифр в числе будем использовать целочисленное деление, (a div b).   
           Для определения цифр числа используется функция определения остатка от целочисленного деления – (a mod 10).  
           Проверим работу алгоритма. Введем значение *N = 12345*. Перепишем значение переменной **N** в переменную **M** т.к. нам необходимо запомнить начальное значение переменной **N**, потому что оно будет использоваться дважды (когда мы считаем кол-во цифр числа, и когда находим max значение цифр числа).  




***Пример 2.*** Вычислить сумму членов ряда

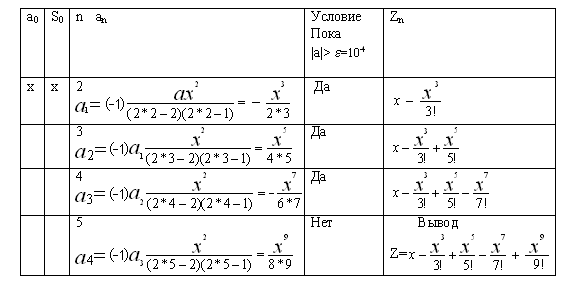
a_cycle5.bmp  
  
Для вычисления члена ряда будем использовать рекуррентную формулу

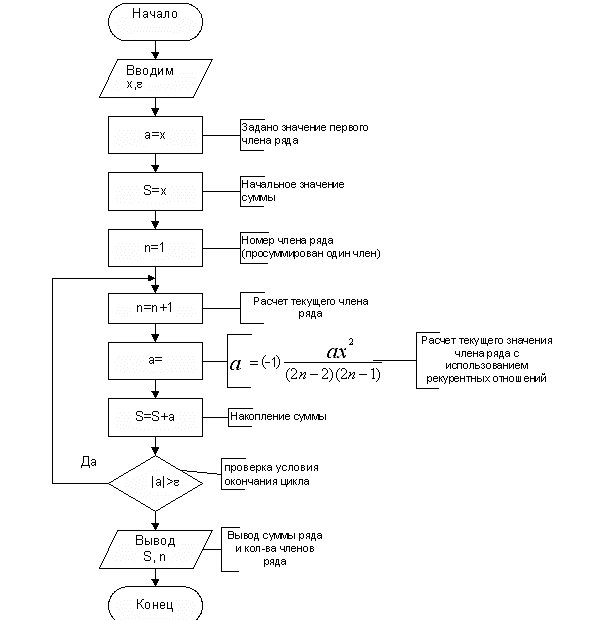
a_cycle6.bmp



***Пример 3.*** Вычислить значение суммы членов бесконечного ряда

i_cycle5.bmp  
с точностью до члена ряда, меньшего *E = 10-4* для *X = 0,1*. Определить число членов ряда, вошедших в сумму.  
Для вычисления общего члена ряда используем формулу  
i_cycle6.bmp  
используем рекуррентное соотношение, выразив **n**-й член через **(n-1)**-й  
i_cycle7.bmp  
Значение первого члена ряда вычислим до цикла с помощью присваивания *a = X*, а всех последующих членов ряда по рекуррентному соотношению  
i_cycle8.bmp  
Схема алгоритма задачи выглядит следующим образом:





***Вывод рекуррентной формулы***

При решении задач на вычисление сумм (произведений) членов конечных или бесконечных последовательностей, вычисления пределов последовательностей наибольшую трудность представляет получение рекуррентной формулы.

Далеко не во всех задачах закономерность, связывающая предыдущий и последующий члены последовательности, очевидна.

Рассмотрим последовательность

Представим ее в виде:

Общая схема получения рекуррентной формулы:

1. Первый член последовательность обычно берется из задания

2. Запишем формулу для любого члена последовательности, кроме первого

Тогда n-ый и (n+1)-ый члены последовательности принимают вид:   
http://algmet.narod.ru/theory_a4m/rec_sootn/Image170.gifhttp://algmet.narod.ru/theory_a4m/rec_sootn/Image170.gif,

3. Разделим (n+1)-ый член на n-ый.

В результате получим знаменатель последовательности, т.е. тот множитель, который связывает следующий член последовательности с предыдущим.

Следовательно, рекуррентная формула имеет вид

Правильность полученной формулы доказывается индукцией.

Проверим, действительно ли по этой формуле получаем члены последовательности, идущие подряд.

При n=1

При n=2

При n=3  и т.д.

Полученная формула действительно формирует члены последовательности.

Практическая работа включает 3 задачи из различных предметных областей. Необходимо выполнить 1-2 задачи по выбору.

Для каждой задачи необходимо выполнить следующие действия:

1. Выполнить построение математической модели задачи (рекурсивную формулу)

2. Составить блок-схему алгоритма

3. Написать программу, реализующую составленный алгоритм на языках программирования Паскаль, Си (по выбору).

Программа должна включать:

* описание данных с типами, допустимыми для решения задачи
* ввод исходных данных с клавиатуры (если необходимо по условию)
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

4. Подготовить данные для тестирования работы

5. Проверить работоспособность программы на выбранных тестах.

6. Оформить отчет по практической работе в MS WORD.

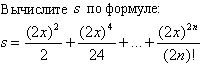
Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление переменных разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила вычисления нарастающей суммы в цикле;
* правила построения циклических операторов;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

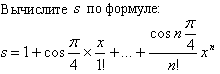
*Вариант №1:*

1. Составьте блок-схему и программу для нахождения суммы, произведения и среднего арифметического натуральных двузначных чисел, кратных 10.
2. a_cycle1.bmp
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = 1/n*

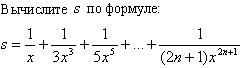
*Вариант №2:*

1. Определите и выведите на экран количество натуральных чисел, меньших 1000 и делящихся на 5 и 7 одновременно.
2. 
3. Популяция лосей каждый год сокращается вдвое. Через сколько лет их вообще не останется, если первоначально было **n** лосей.

*Вариант №3:*

1. Составьте блок-схему и программу для нахождения среднего геометрического, суммы и произведения всех натуральных трехзначных чисел на отрезке [300,900].
2. 
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = n/(n+1)*

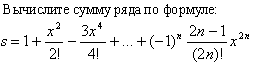
*Вариант №4:*

1. Составьте блок-схему и программу для нахождения количества, суммы и произведения всех натуральных чисел, делящихся на 4, на отрезке [16,64].
2. 
3. Процентная ставка по вкладу – 15% годовых. Первоначальный вклад составляет **n** рублей. Через сколько лет он станет больше 100000?

*Вариант №5:*

1. Составьте блок-схему и программу для нахождения среднего арифметического, геометрического и количества всех натуральных нечетных двузначных чисел.
2. a_cycle4.bmp
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = 1/(2n+1)2*

*Вариант №6:*

1. Каждая бактерия делится на две в течении одной минуты. В начальный момент имеется одна бактерия. Сколько их станет через **n** минут?
2. 
3. Царевна-лягушка съедает ежедневно на 20 комаров больше, чем в предыдущий день. Через сколько дней количество съеденных комаров превысит 100?

*Вариант №7:*

1. Сбербанк начисляет 2% годовых. Какой станет сумма вклада **S**, положенная в банк на **N** лет?
2. Среди чисел на отрезке [0..100] найдите процентное соотношение чисел кратных пяти, семи и девяти. Посчитайте, каких чисел больше.
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = 1 - 1/sin(n)*

*Вариант №8:*

1. Амеба каждые 3 часа делится на 2 клетки. Сколько клеток будет через **n** часов?
2. Среди чисел на отрезке [100..200] найдите количество и сумму чисел, меньших по абсолютному значению квадрата порядкового номера.
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = 1/(2n+5)*

*Вариант №9:*

1. Известный пират Флинт положил некоторую сумму **S** в банк под 3% годовых. Какую сумму он сможет получить через **n** лет, если инфляция составляет 2% в год.
2. Царевна Лягушка съедает ежедневно на 20 комаров больше, чем в предыдущий день и еще 2 комара. Сколько комаров она съест через месяц, если в первый день было съедено **n** комаров.
3. Бактерия делится на 2 части в течении одной минуты. Сколько минут понадобится, чтобы их стало **n**?

*Вариант №10:*

1. Общий член последовательности вычисляется по формуле: *an = sin(2n)*. Сколько членов последовательности с номерами 1, 2, 4, 8 ,..., 1024 отрицательны.
2. Найдите и выведите на экран все двузначные числа, сумма цифр которых не меняется при умножении на 2.
3. Амеба каждые 3 часа делится на 2 клетки. Через сколько часов она достигнет размера в 50 клеток?

*Вариант №11:*

1. Мой богатый дядюшка подарил мне доллар на день рождения. В каждый следующий день рождения он удваивал свой подарок и прибавлял к нему столько долларов, сколько лет мне исполнилось. Какая сумма будет подарена на каждый день рождения до совершеннолетия? Сколько всего денег будет подарено?
2. Найдите и выведите на экран все двузначные числа, сумма квадратов цифр которых делится без остатка на 17.
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = (-1)n-1/nn*

*Вариант №12:*

1. Вычислите и выведите на экран количество шаров в пирамиде из **N** слоев и количество шаров в каждом слое, если количество шаров в каждом слое равно квадрату порядкового номера слоя.
2. Среди чисел от 1 до **N** найдите и выведите на экран такие, запись которых совпадает с последними цифрами их квадрата.
3. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = 3/2n*

*Вариант №13:*

1. Билет называют счастливым, если в его номере сумма первых двух цифр равна сумме последних двух цифр. Сколько существует счастливых билетов с семизначными номерами?
2. Найдите и выведите на экран все трехзначные числа, которые без остатка делятся на 11 и имеют в своей записи хотя бы одну цифру 3.
3. Дано натуральное число **n**. Найти сумму первой и последней цифры этого числа.

*Вариант №14:*

1. Составьте блок-схему и программу для нахождения количества, суммы и произведения всех натуральных чисел, делящихся на 4, на отрезке [16,64].
2. На первых двух днях рождения у Пятачка и Кролика Вини-Пух съел всего 100 г пищи. Сколько съест Вини-Пух на **n**-ом дне рождения и сколько за все дни рождения, если на каждом следующем дне рождения он съедает столько, сколько всего на предыдущих.
3. Даны два натуральных числа **m** и **n**. Проверить, есть ли в записи числа **m** цифры, совпадающие с цифрами числа **n**.

*Вариант №15:*

1. Дано натуральное число **n**. Поменять местами первую и последнюю цифры этого числа.
2. Дан числовой ряд и некоторое число **е**. Считать сумму до тех пор, пока модуль очередного члена ряда больше или равен заданному **е**. Общий член ряда имеет вид:   
   *an = 1/2n + 1/3n*
3. Вычислить сумму ряда S = 1+X/2+X2/3+…+X16/17

**Тема 4. Обработка одномерных массивов в Паскале**

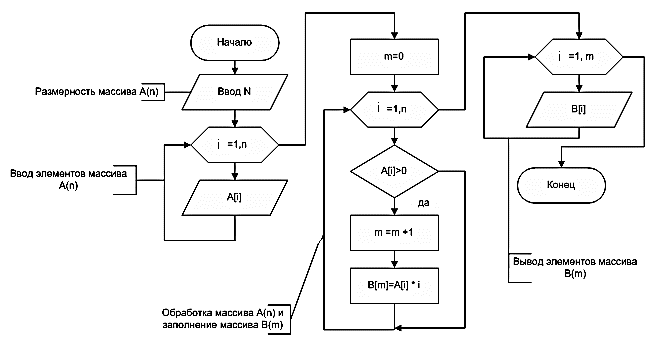
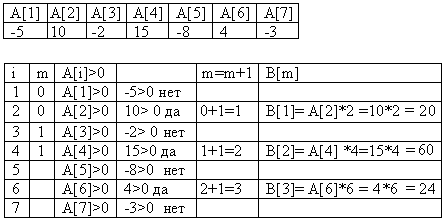
*(ориентировочное время выполнения задания — 2 часа)*

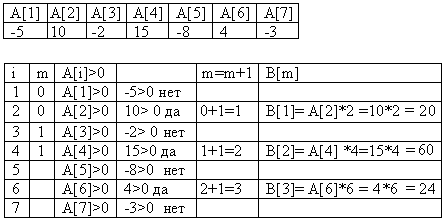
**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач обработки одномерных массивов (ввод, вывод, слияние, разбиение, поиск максимальных и минимальных значений и т.п.).

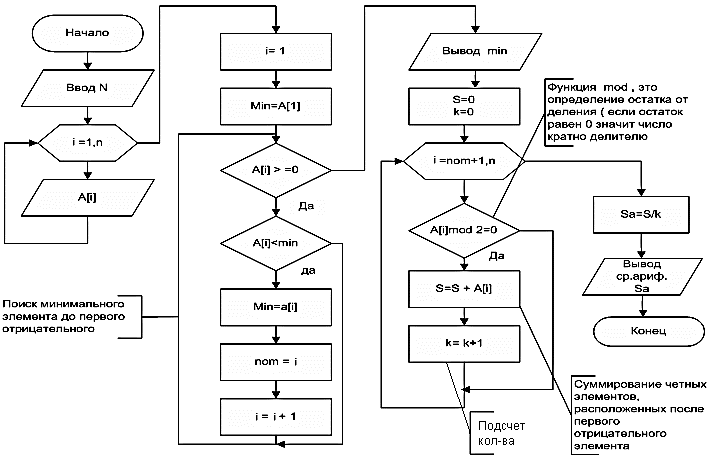
**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

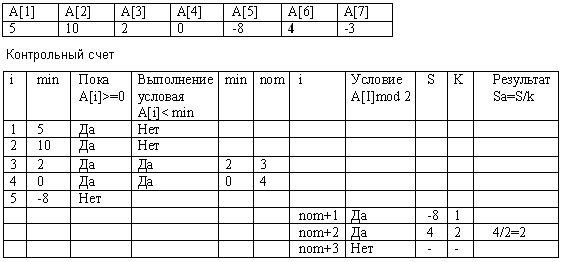
***Методические рекомендации решения задач***

***Методика решения задач***

***Пример 1.*** Дан одномерный массив **A(n)**, сформировать массив **B(m)** из положительных элементов массива   
**A(n)**. Каждый элемент массива **B(m)** вычисляется как произведение элемента массива **A(n)** на его индекс. Вывести полученный массив **B(m)** на печать.  
  
  
           После ввода элементов массива **A(n)** начинается его обработка.  
           Переменная **m** – это счетчик элементов в массиве **B** первоначально равна 0, потому что в массиве **A** может не оказаться ни одного элемента удовлетворяющего условию задачи (в нашем случае - это положительные элементы массива **A**).  
           В цикле проверяется условие задачи, т.е. если элемент массива **A[i]** положителен, тогда счетчик **m** увеличивается на единицу, и формируется массив **В(n)**. Если условие не выполняется, осуществляется переход на начало цикла.  
           Вывод результата, т.е. элементов массива **B(m)**, осуществляется в цикле, в котором переменная цикла изменяется от 1 до **m** (количество элементов записанных в массив **B**)  
           Пусть дан исходный массив **A(n)**  


  
           Таким образом мы видим, что массив **В(m)** содержит всего три элемента: *В[1] = 20; B[2] = 60; B[3] = 24*.

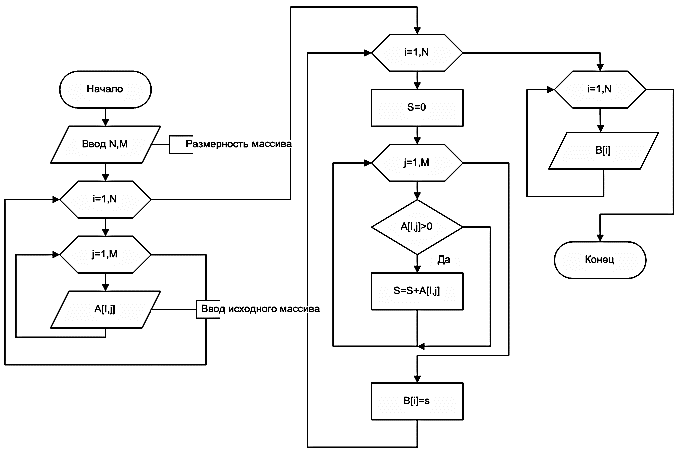
***Пример 2.*** Дан одномерный массив **A(n)**. Найти минимальный элемент до первого отрицательного элемента в массиве и среднее арифметическое четных элементов, расположенных после первого отрицательного элемента.  
  
           Для поиска минимального элемента используется цикл с предусловием.   
           Пока элемент в массиве **A** - неотрицательное число, ищется минимальный элемент.



***Пример 3 .*** В двумерном массиве найти минимальный элемент и поменять его с первым элементом в массиве.  
           Алгоритм данной задачи построен следующим образом. С клавиатуры (или случайным образом) вводятся элементы матрицы. За минимальный элемент принимаем элемент **A[1,1], k = 1** (строка) и **l = 1** (столбец) – адрес первого минимального элемента (на тот случай, если этот элемент окажется действительно минимальным).  
           Далее просматривается каждый элемент и сравнивается с минимальным элементом, как только следующий элемент оказывается меньше минимального элемента, за минимальный принимают этот элемент, и переменным **k** и **l** присваивается адрес этого элемента. Затем осуществляется перестановка минимального элемента, который стоит в **k** строке и **l** столбце с элементом **A[1,1]**.

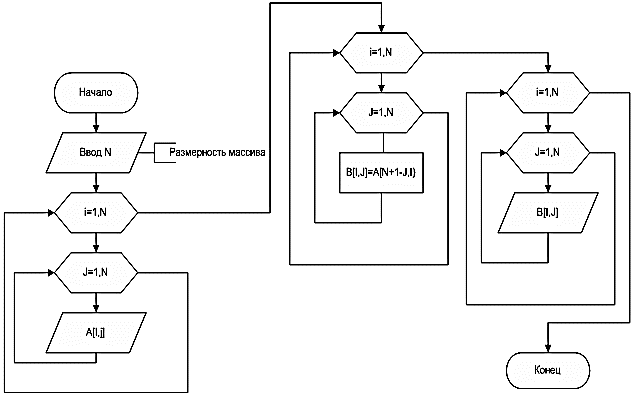
***Пример 4.*** Дан двумерный массив **A[n,m]**. Сформировать одномерный массив **B[n]** из сумм каждой строки массива **A[n,m]**.

После ввода массива идет обработка, т. е. нахождение суммы положительных элементов в каждой строке. По окончании цикла **J** (по столбцам) накопившаяся сумма записывается в массив **B**. Для того чтобы накопившаяся сумма строки не посчиталась в следующей, её необходимо обнулить (*S = 0*).



***Пример 5.*** Дана квадратная матрица **A(N,N)**, состоящая из натуральных чисел. Повернуть ее на 90 градусов по часовой стрелке и вывести результат на печать. Основная задача, которую нужно решить, состоит в определении преобразования индексов элементов матрицы. Для примера рассмотрим матрицу 3 х 3 и посмотрим, что происходит с элементами при повороте.  
  
           Если считать, что после поворота появилась другая матрица **В**, то соответствие между элементами устанавливается следующим образом  
B11 – A31  
B12 – A21  
B21 – A32  
B22 – A22 и т. д., т.е. B[I,J] – A[L,M]  
           Внимательно изучив соответствие, можно утверждать, что для элементов матрицы **N x M** справедлива следующая система уравнений:  
*I = M  
J + L = N + 1*  
отсюда правило преобразований выглядит следующим образом

*B[I,J] = A[N+1-J,I]*  
блок-схема, решающая эту задачу выглядит следующим образом:



Контрольный счет *N = 3*  
           Элементу *В[1,1] = A[N+1-J,I] = A[3+1-1,1] = A[3,1]*  
           Элементу *В[1,2] = A[N+1-J,I] = A[3+1-2,1] = A[2,1]*  
           И т.д.  
           Элементу *В[3,3] = A[N+1-J,I] = A[3+1-3,3] = A[1,3]* не трудно увидеть, что контрольный счет совпадает с табл. 1 матрица после поворота.

Практическая работа включает 3 задачи из различных предметных областей. Необходимо выполнить 1-2 задачи по выбору.

Для каждой задачи необходимо выполнить следующие действия:

1. Составить блок-схему алгоритма

2. Написать программу, реализующую составленный алгоритм на языке программирования Паскаль.

Программа должна включать:

* Описание одномерного массива типа, допустимого для решения задачи
* ввод массива с клавиатуры, с помощью генератора случайных чисел
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

3. Подготовить данные для тестирования работы

4. Проверить работоспособность программы на выбранных тестах.

5. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление массивов разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила заполнения и вывода на экран элементов массива;
* нахождение максимума и минимума среди элементов массива;
* обмен данными между элементами массива;
* слияние двух массивов в один, разбиение массива на два;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

*Вариант №1:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен предыдущему, увеличенному в 2 раза + порядковый номер элемента.
2. Задайте массив из **n** элементов. Элементы, большие 55, заменить на максимальный. Выведите на экран исходный и полученный массив.
3. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите два новых массива. Каждый элемент первого массива равен сумме соответствующего исходного элемента и его квадрата. Каждый элемент второго массива равен разности соответствующих элементов первых двух массивов. В полученном массиве найти максимальный, минимальный элементы и их номера. Выведите на экран исходные и полученные результаты.

*Вариант №2:*

1. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен сумме индекса соответствующего элемента исходного массива и его квадрата. Выведите на экран исходные и полученные результаты.
2. Результаты игр по футболу хранятся в массиве. Каждые три последовательных элемента массива содержат соответственно номер игры, количество забитых и количество пропущенных мячей. Определите и выведите на экран, в скольких играх разность забитых и пропущенных мячей больше трех.
3. Дан одномерный массив из **n** чисел. Найти сумму элементов одномерного массива до первого 0.

*Вариант №3:*

1. Задайте массив из **n** элементов и поменяйте знак всем нечетным элементам массива на противоположный. Выведите на экран исходный и измененный массивы.
2. Информация о средней суточной температуре воздуха за месяц задана в виде массива. Определите и выведите на экран, сколько раз температура была ниже среднемесячной.
3. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен кубу соответствующего элемента исходного массива. Найти максимальный элемент среди элементов с четными номерами. Выведите на экран исходные и полученные результаты.

*Вариант №4:*

1. Дан одномерный массив из **n** чисел. Вывести на экран, отсортирован ли массив по возрастанию.
2. Информация о росте учащихся одного класса задана в виде массива. Рост девочек кодируется знаком «+» (плюс), рост мальчиков – знаком «-» (минус). Определите и выведите на экран средний рост мальчиков.
3. Результаты игр по футболу хранятся в массиве. Каждые три последовательных элемента массива содержат соответственно номер игры, количество забитых и количество пропущенных мячей. Определите и выведите на экран общее количество забитых и общее количество пропущенных мячей.

*Вариант №5:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен предыдущему, уменьшенному в **n** раз (**n** - порядковый номер элемента).
2. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен среднему арифметическому элементов, стоящих до и после него в исходном массиве. Выведите на экран исходные и полученные результаты.
3. Дана последовательность из **n** чисел. Получить максимальное число из *(a(1)+a(n), a(2)+a(n-1),…)*

*Вариант №6:*

1. Задайте массив из **n** элементов. Найдите произведение элементов, имеющих нечетные индексы. Выведите на экран исходный и измененный массивы.
2. Дан одномерный массив из **n** случайных чисел в диапазоне [-10; 10]. Найти среднее арифметическое элементов до первого отрицательного.
3. Даны два массива **А(5)** и **В(5)**. Сформируйте массив **С(10)**, каждый четный по номеру элемент которого равен сумме соответствующих элементов из **А** и **В**, а нечетный – их разности:  
   *С(1) = А(1) + В(1); С(2) = А(1) - В(1); С(3) = А(2) + В(2); С(4) = А(2) - В(2)… .*  
   Выведите на экран массивы **А(5), В(5)** и **С(10)**.

*Вариант №7:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен сумме двух предыдущих элементов, умноженных на порядковый номер элемента.
2. Задайте массив из **n** элементов. Замените все нечетные элементы с четными индексами на нули. Выведите на экран исходный и измененный массивы.
3. Дано число **A** и одномерный массив из **n** элементов. Выбрать две пары чисел (**В** и **С**) из массива таким образом, чтобы **A, B, C** могли быть длинами сторон равнобедренного треугольника.

*Вариант №8:*

1. Дан одномерный массив из **n** чисел. Если число **А** встречается в массиве, найти количество элементов до числа **А** и после него.
2. Сформируйте массив, содержащий **n** элементов, с помощью чисел, введенных с клавиатуры. Получите новый массив, каждый элемент которого в **i** раз больше соответствующего элемента первого массива, где **i** – индекс элемента. Выведите на экран исходные и полученные результаты.
3. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен обратному соответствующему элементу исходного массива. Подсчитайте количество неизменившихся элементов и найдите в новом массиве максимальный по абсолютной величине отрицательный элемент. Выведите на экран исходные и полученные результаты.

*Вариант №9:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен разнице первого и предыдущего элемента, деленной на порядковый номер элемента.
2. Информация о росте учащихся одного класса хранится в массиве. Каждый нечетный по номеру элемент содержит порядковый номер учащегося по списку в журнале, а нечетный – его рост. Определите и выведите на экран количество учеников, рост которых больше 170 см.
3. Дан одномерный массив из **n** чисел. Найти минимальный элемент массива до «0» и минимальный элемент массива после нуля. Поменять их местами.

*Вариант №10:*

1. Задайте массив из **n** элементов. Замените нулем элементы массива, которые больше введенного с клавиатуры числа **А**. Выведите на экран исходный и измененный массивы.
2. Дан одномерный массив из **n** чисел. Вывести на экран, отсортирован ли массив по убыванию.
3. В заданном одномерном массиве из **n** элементов найти количество повторяющихся чисел.

*Вариант №11:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива, начиная с третьего, равен произведению двух предыдущих и его порядкового номера, первые два элемента задаются с клавиатуры.
2. Сформируйте массив, содержащий **n** элементов, с помощью чисел, введенных с клавиатуры. Получите новый массив, каждый элемент которого равен удвоенному числу соответствующего элемента исходного массива, если он больше числа **A**, введенного с клавиатуры, и не изменяется в обратном случае. Выведите на экран исходные и полученные результаты.
3. Информация о площадях и средней урожайности пшеницы хранится в массиве, где каждый элемент с четным индексом – засеянная площадь, а с нечетным – урожайность. Определите и выведите на экран количество собранной пшеницы и среднюю урожайность. Исходные данные задать случайным образом в допустимом диапазоне значений для условия задачи.

*Вариант №12:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый четный элемент массива равен целой части от деления случайного числа на текущий индекс, а нечетный равен квадрату предыдущего индекса.
2. Задайте массив из **n** элементов. Найдите разность наибольшего и наименьшего элемента. Выведите на экран исходный массив и полученную разность.
3. Информация о массе учеников первого класса хранятся в массиве. Определите и выведите на экран количество молока, требующееся на класс из расчета один стакан (200 мл) на учащегося, если известно, что учащимся, масса которых меньше 30 кг, назначают дополнительный стакан молока.

*Вариант №13:*

1. Сформируйте массив, содержащий n элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен сумме соответствующего исходного элемента и его квадрата. В полученном массиве найти максимальный элемент и его номер. Выведите на экран исходные и полученные результаты.
2. Задайте массив из **n** элементов. Все положительные элементы замените минимальными. Выведите на экран исходный и полученный массив.
3. Дан массив из **n** элементов. Сожмите массив, выбросив каждый третий его элемент, без использования дополнительного массива. Выведите на экран исходный и измененный массивы.

*Вариант №14:*

1. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен среднему арифметическому соответствующего элемента исходного массива и веденного с клавиатуры числа **А**. В полученном массиве найти максимальный элемент среди элементов с нечетными номерами. Выведите на экран исходные и полученные результаты.
2. Задайте массив из **n** элементов. Замените все его положительные элементы на максимальный отрицательный элемент. Выведите на экран исходный и полученный массив.
3. Дан массив из случайных чисел **С(10)**. Сформируйте массивы **А(5)** и **В(5)**. Каждый элемент массива **А** равен четному по номеру элементу массива **С**, уменьшенному в 2 раза, а каждый элемент массива **В** – нечетному, увеличенному в 2 раза. Выведите на экран массивы **А(5), В(5)** и **С(10)**.

*Вариант №15:*

1. Задайте массив из **n** элементов. Замените элементы кратные трем, на противоположные по знаку. Выведите на экран исходный и измененный массивы.
2. В соревновании принимают участие 30 спортсменов. Номера и результаты спортсменов хранятся в двух одномерных массивах. Определите и выведите на экран номера тех участников, которые набрали не менее 70 баллов.
3. Дан одномерный массив из **n** чисел с чередованием знаков («+», «-»). Найти среднее арифметическое элементов массива до «0» и заменить этим числом все меньшие его элементы.

**Тема 5. Обработка одномерных массивов в Си**

*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач обработки одномерных массивов (ввод, вывод, слияние, разбиение, поиск максимальных и минимальных значений и т.п.).

**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

***Методика выполнения работы.***

Практическая работа включает 3 задачи из различных предметных областей. Необходимо выполнить 1-2 задачи по выбору.

Для каждой задачи необходимо выполнить следующие действия:

1. Составить блок-схему алгоритма

2. Написать программу, реализующую составленный алгоритм на языке программирования Си.

Программа должна включать:

* Описание одномерного массива типа, допустимого для решения задачи
* ввод массива с клавиатуры, с помощью генератора случайных чисел
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

3. Подготовить данные для тестирования работы

4. Проверить работоспособность программы на выбранных тестах.

5. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление массивов разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила заполнения и вывода на экран элементов массива;
* нахождение максимума и минимума среди элементов массива;
* обмен данными между элементами массива;
* слияние двух массивов в один, разбиение массива на два;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

*Вариант №1:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен предыдущему, увеличенному в 2 раза + порядковый номер элемента.
2. Задайте массив из **n** элементов. Элементы, большие 55, заменить на максимальный. Выведите на экран исходный и полученный массив.
3. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите два новых массива. Каждый элемент первого массива равен сумме соответствующего исходного элемента и его квадрата. Каждый элемент второго массива равен разности соответствующих элементов первых двух массивов. В полученном массиве найти максимальный, минимальный элементы и их номера. Выведите на экран исходные и полученные результаты.

*Вариант №2:*

1. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен сумме индекса соответствующего элемента исходного массива и его квадрата. Выведите на экран исходные и полученные результаты.
2. Результаты игр по футболу хранятся в массиве. Каждые три последовательных элемента массива содержат соответственно номер игры, количество забитых и количество пропущенных мячей. Определите и выведите на экран, в скольких играх разность забитых и пропущенных мячей больше трех.
3. Дан одномерный массив из **n** чисел. Найти сумму элементов одномерного массива до первого 0.

*Вариант №3:*

1. Задайте массив из **n** элементов и поменяйте знак всем нечетным элементам массива на противоположный. Выведите на экран исходный и измененный массивы.
2. Информация о средней суточной температуре воздуха за месяц задана в виде массива. Определите и выведите на экран, сколько раз температура была ниже среднемесячной.
3. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен кубу соответствующего элемента исходного массива. Найти максимальный элемент среди элементов с четными номерами. Выведите на экран исходные и полученные результаты.

*Вариант №4:*

1. Дан одномерный массив из **n** чисел. Вывести на экран, отсортирован ли массив по возрастанию.
2. Информация о росте учащихся одного класса задана в виде массива. Рост девочек кодируется знаком «+» (плюс), рост мальчиков – знаком «-» (минус). Определите и выведите на экран средний рост мальчиков.
3. Результаты игр по футболу хранятся в массиве. Каждые три последовательных элемента массива содержат соответственно номер игры, количество забитых и количество пропущенных мячей. Определите и выведите на экран общее количество забитых и общее количество пропущенных мячей.

*Вариант №5:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен предыдущему, уменьшенному в **n** раз (**n** - порядковый номер элемента).
2. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен среднему арифметическому элементов, стоящих до и после него в исходном массиве. Выведите на экран исходные и полученные результаты.
3. Дана последовательность из **n** чисел. Получить максимальное число из *(a(1)+a(n), a(2)+a(n-1),…)*

*Вариант №6:*

1. Задайте массив из **n** элементов. Найдите произведение элементов, имеющих нечетные индексы. Выведите на экран исходный и измененный массивы.
2. Дан одномерный массив из **n** случайных чисел в диапазоне [-10; 10]. Найти среднее арифметическое элементов до первого отрицательного.
3. Даны два массива **А(5)** и **В(5)**. Сформируйте массив **С(10)**, каждый четный по номеру элемент которого равен сумме соответствующих элементов из **А** и **В**, а нечетный – их разности:  
   *С(1) = А(1) + В(1); С(2) = А(1) - В(1); С(3) = А(2) + В(2); С(4) = А(2) - В(2)… .*  
   Выведите на экран массивы **А(5), В(5)** и **С(10)**.

*Вариант №7:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен сумме двух предыдущих элементов, умноженных на порядковый номер элемента.
2. Задайте массив из **n** элементов. Замените все нечетные элементы с четными индексами на нули. Выведите на экран исходный и измененный массивы.
3. Дано число **A** и одномерный массив из **n** элементов. Выбрать две пары чисел (**В** и **С**) из массива таким образом, чтобы **A, B, C** могли быть длинами сторон равнобедренного треугольника.

*Вариант №8:*

1. Дан одномерный массив из **n** чисел. Если число **А** встречается в массиве, найти количество элементов до числа **А** и после него.
2. Сформируйте массив, содержащий **n** элементов, с помощью чисел, введенных с клавиатуры. Получите новый массив, каждый элемент которого в **i** раз больше соответствующего элемента первого массива, где **i** – индекс элемента. Выведите на экран исходные и полученные результаты.
3. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен обратному соответствующему элементу исходного массива. Подсчитайте количество неизменившихся элементов и найдите в новом массиве максимальный по абсолютной величине отрицательный элемент. Выведите на экран исходные и полученные результаты.

*Вариант №9:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива равен разнице первого и предыдущего элемента, деленной на порядковый номер элемента.
2. Информация о росте учащихся одного класса хранится в массиве. Каждый нечетный по номеру элемент содержит порядковый номер учащегося по списку в журнале, а нечетный – его рост. Определите и выведите на экран количество учеников, рост которых больше 170 см.
3. Дан одномерный массив из **n** чисел. Найти минимальный элемент массива до «0» и минимальный элемент массива после нуля. Поменять их местами.

*Вариант №10:*

1. Задайте массив из **n** элементов. Замените нулем элементы массива, которые больше введенного с клавиатуры числа **А**. Выведите на экран исходный и измененный массивы.
2. Дан одномерный массив из **n** чисел. Вывести на экран, отсортирован ли массив по убыванию.
3. В заданном одномерном массиве из **n** элементов найти количество повторяющихся чисел.

*Вариант №11:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый элемент массива, начиная с третьего, равен произведению двух предыдущих и его порядкового номера, первые два элемента задаются с клавиатуры.
2. Сформируйте массив, содержащий **n** элементов, с помощью чисел, введенных с клавиатуры. Получите новый массив, каждый элемент которого равен удвоенному числу соответствующего элемента исходного массива, если он больше числа **A**, введенного с клавиатуры, и не изменяется в обратном случае. Выведите на экран исходные и полученные результаты.
3. Информация о площадях и средней урожайности пшеницы хранится в массиве, где каждый элемент с четным индексом – засеянная площадь, а с нечетным – урожайность. Определите и выведите на экран количество собранной пшеницы и среднюю урожайность. Исходные данные задать случайным образом в допустимом диапазоне значений для условия задачи.

*Вариант №12:*

1. Сформируйте и выведите на экран массив из **n** элементов, где каждый четный элемент массива равен целой части от деления случайного числа на текущий индекс, а нечетный равен квадрату предыдущего индекса.
2. Задайте массив из **n** элементов. Найдите разность наибольшего и наименьшего элемента. Выведите на экран исходный массив и полученную разность.
3. Информация о массе учеников первого класса хранятся в массиве. Определите и выведите на экран количество молока, требующееся на класс из расчета один стакан (200 мл) на учащегося, если известно, что учащимся, масса которых меньше 30 кг, назначают дополнительный стакан молока.

*Вариант №13:*

1. Сформируйте массив, содержащий n элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен сумме соответствующего исходного элемента и его квадрата. В полученном массиве найти максимальный элемент и его номер. Выведите на экран исходные и полученные результаты.
2. Задайте массив из **n** элементов. Все положительные элементы замените минимальными. Выведите на экран исходный и полученный массив.
3. Дан массив из **n** элементов. Сожмите массив, выбросив каждый третий его элемент, без использования дополнительного массива. Выведите на экран исходный и измененный массивы.

*Вариант №14:*

1. Сформируйте массив, содержащий **n** элементов, с помощью датчика случайных чисел. Получите новый массив, каждый элемент которого равен среднему арифметическому соответствующего элемента исходного массива и веденного с клавиатуры числа **А**. В полученном массиве найти максимальный элемент среди элементов с нечетными номерами. Выведите на экран исходные и полученные результаты.
2. Задайте массив из **n** элементов. Замените все его положительные элементы на максимальный отрицательный элемент. Выведите на экран исходный и полученный массив.
3. Дан массив из случайных чисел **С(10)**. Сформируйте массивы **А(5)** и **В(5)**. Каждый элемент массива **А** равен четному по номеру элементу массива **С**, уменьшенному в 2 раза, а каждый элемент массива **В** – нечетному, увеличенному в 2 раза. Выведите на экран массивы **А(5), В(5)** и **С(10)**.

*Вариант №15:*

1. Задайте массив из **n** элементов. Замените элементы кратные трем, на противоположные по знаку. Выведите на экран исходный и измененный массивы.
2. В соревновании принимают участие 30 спортсменов. Номера и результаты спортсменов хранятся в двух одномерных массивах. Определите и выведите на экран номера тех участников, которые набрали не менее 70 баллов.
3. Дан одномерный массив из **n** чисел с чередованием знаков («+», «-»). Найти среднее арифметическое элементов массива до «0» и заменить этим числом все меньшие его элементы.

**Тема 6. Обработка двумерных массивов в Паскале**

*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач обработки двумерных массивов (ввод, вывод, слияние, разбиение, поиск максимальных и минимальных значений и т.п.).

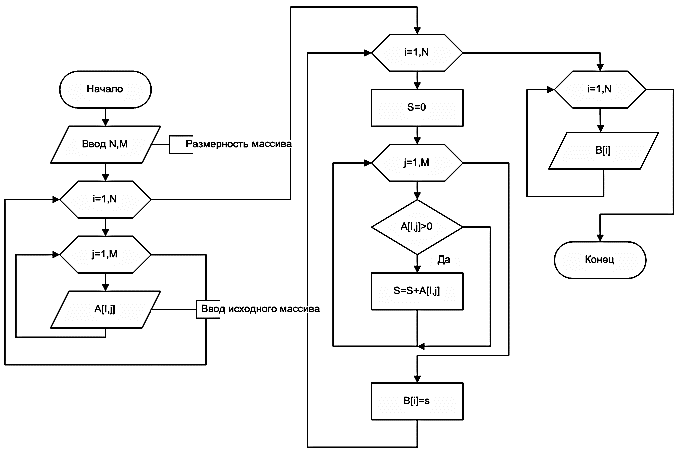
**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

***Методические рекомендации решения задач***

***Пример 1.*** В двумерном массиве найти минимальный элемент и поменять его с первым элементом в массиве.  
           Алгоритм данной задачи построен следующим образом. С клавиатуры (или случайным образом) вводятся элементы матрицы. За минимальный элемент принимаем элемент **A[1,1], k = 1** (строка) и **l = 1** (столбец) – адрес первого минимального элемента (на тот случай, если этот элемент окажется действительно минимальным).  
           Далее просматривается каждый элемент и сравнивается с минимальным элементом, как только следующий элемент оказывается меньше минимального элемента, за минимальный принимают этот элемент, и переменным **k** и **l** присваивается адрес этого элемента. Затем осуществляется перестановка минимального элемента, который стоит в **k** строке и **l** столбце с элементом **A[1,1]**.

***Пример 2.*** Дан двумерный массив **A[n,m]**. Сформировать одномерный массив **B[n]** из сумм каждой строки массива **A[n,m]**.

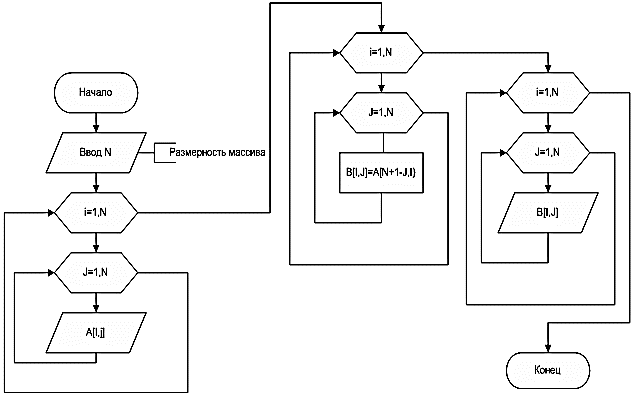
После ввода массива идет обработка, т. е. нахождение суммы положительных элементов в каждой строке. По окончании цикла **J** (по столбцам) накопившаяся сумма записывается в массив **B**. Для того чтобы накопившаяся сумма строки не посчиталась в следующей, её необходимо обнулить (*S = 0*).



***Пример 3.*** Дана квадратная матрица **A(N,N)**, состоящая из натуральных чисел. Повернуть ее на 90 градусов по часовой стрелке и вывести результат на печать. Основная задача, которую нужно решить, состоит в определении преобразования индексов элементов матрицы. Для примера рассмотрим матрицу 3 х 3 и посмотрим, что происходит с элементами при повороте.  
  
           Если считать, что после поворота появилась другая матрица **В**, то соответствие между элементами устанавливается следующим образом  
B11 – A31  
B12 – A21  
B21 – A32  
B22 – A22 и т. д., т.е. B[I,J] – A[L,M]  
           Внимательно изучив соответствие, можно утверждать, что для элементов матрицы **N x M** справедлива следующая система уравнений:  
*I = M  
J + L = N + 1*  
отсюда правило преобразований выглядит следующим образом

*B[I,J] = A[N+1-J,I]*  
блок-схема, решающая эту задачу выглядит следующим образом:

Контрольный счет *N = 3*  
           Элементу *В[1,1] = A[N+1-J,I] = A[3+1-1,1] = A[3,1]*  
           Элементу *В[1,2] = A[N+1-J,I] = A[3+1-2,1] = A[2,1]*  
           И т.д.  
           Элементу *В[3,3] = A[N+1-J,I] = A[3+1-3,3] = A[1,3]* не трудно увидеть, что контрольный счет совпадает с табл. 1 матрица после поворота.



Практическая работа включает 3 задачи из различных предметных областей. Необходимо выполнить 1-2 задачи по выбору.

Для каждой задачи необходимо выполнить следующие действия:

1. Составить блок-схему алгоритма

2. Написать программу, реализующую составленный алгоритм на языке программирования Паскаль.

Программа должна включать:

* Описание двумерного массива типа, допустимого для решения задачи
* ввод массива с клавиатуры, с помощью генератора случайных чисел
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

3. Подготовить данные для тестирования работы

4. Проверить работоспособность программы на выбранных тестах.

5. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление массивов разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила заполнения и вывода на экран элементов массива (построчно);
* нахождение максимума и минимума среди элементов массива;
* обмен данными между строками (столбцами) массива;
* преобразование двумерного массива в одномерный, создание двумерного массива из одномерного;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

*Вариант №1:*

1. Заполнить случайными числами матрицу **n\*m**. Вывести на печать вторую строку массива.
2. Заполнить матрицу **m\*n** с клавиатуры. Найти сумму, количество и произведение нечетных элементов массива.
3. Составить программу формирования и вывода на экран одномерного массива, состоящего из максимальных элементов столбцов двумерного массива, заданного с помощью датчика случайных чисел.

*Вариант №2:*

1. Заполнить матрицу **m\*n** с клавиатуры. Найти количество, сумму и среднее арифметическое положительных элементов массива.
2. Дана матрица с размерностью **m\*n,** заполненная случайными целыми числами. Поменять местами максимальный и минимальный элементы.
3. Составить программу формирования и вывода на экран одномерного массива, состоящего из минимальных элементов строк двумерного массива, заданного с помощью датчика случайных чисел.

*Вариант №3:*

1. Заполнить случайными числами матрицу **n\*m**. Вывести на печать последний столбец массива.
2. Заполнить матрицу **m\*n** с клавиатуры. Найти максимальный элемент в матрице.
3. Составить программу формирования и вывода на экран одномерного массива, состоящего из максимальных элементов строк двумерного массива, заданного с помощью датчика случайных чисел.

*Вариант №4:*

1. Дана квадратная матрица с размерностью **m\* m,** заполненная случайными числами в диапазоне [0,9]. Найти произведение элементов массива.
2. Заполнить матрицу **m\*n** с клавиатуры. Найти среднее арифметическое чисел, оба индекса которых четные.
3. Дана матрица **m\*n**. Добавить предпоследнюю строку, каждый элемент которой равен сумме элементов соответствующего столбца.

*Вариант №5:*

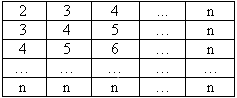
1. Дана матрица с размерностью **m\*n,** заполненная случайными числами в диапазоне [0,9]. Получить новую матрицу путем деления всех элементов на наибольший по модулю элемент.
2. Заполнить матрицу **m\*n** с клавиатуры. Найти количество нулевых элементов массива, стоящих на четных и нечетных местах.
3. В матрице **m\*n,** заполненной с клавиатуры,  поменять местами первый и последний столбцы.

*Вариант №6:*

1. Заполнить случайными числами матрицу **n\*m**. Вывести на печать предпоследний столбец массива.
2. Заполнить матрицу **m\*n** с клавиатуры. Найти минимальный элемент главной диагонали.
3. Заполнить случайными числами квадратную матрицу. Найти скалярное произведение строки с наибольшим элементом матрицы и столбца с наименьшим.

*Вариант №7:*

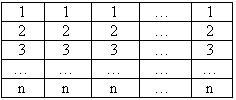
1. Заполнить матрицу **m\*n** с клавиатуры. Найти произведение, сумму и количество отрицательных элементов массива.
2. Сформировать матрицу по заданному образцу:



1. Дана матрица **m\*n**. Добавить предпоследний столбец, каждый элемент которого равен произведению элементов соответствующей строки.

*Вариант №8:*

1. Заполнить матрицу **m\*n** с клавиатуры. Найти минимальный элемент в матрице.
2. Сформировать матрицу по заданному образцу:



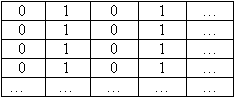
1. Дана матрица **m\*n**, найти минимальный элемент из максимальных в каждом столбце.

*Вариант №9:*

1. Заполнить случайными числами матрицу **n\*m**. Вывести на печать последнюю строку массива.
2. Напишите программу формирования массива размером **5х5** с помощью датчика случайных чисел. Найдите и выведите на экран минимальное значение элементов для каждого столбца, и для исходного массива.
3. В матрице **m\*n** поменять местами максимальный и минимальный элемент в каждой строке.

*Вариант №10:*

1. Заполнить матрицу **m\*n** с клавиатуры. Найти количество, произведение и среднее арифметическое четных элементов массива.
2. Сформировать матрицу по заданному образцу:



1. Дана матрица **m\*n**. Добавить предпоследнюю строку, каждый элемент которой равен произведению элементов соответствующего столбца.

**Тема 7. Обработка двумерных массивов в Си**

*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач обработки двумерных массивов (ввод, вывод, слияние, разбиение, поиск максимальных и минимальных значений и т.п.).

**Задачи:** формирование компетенций ОК 1- ОК 5, ПК 1.1- 1.6

***Методика выполнения работы.***

Практическая работа включает 3 задачи из различных предметных областей. Необходимо выполнить 1-2 задачи по выбору.

Для каждой задачи необходимо выполнить следующие действия:

1. Составить блок-схему алгоритма

2. Написать программу, реализующую составленный алгоритм на языке программирования Си.

Программа должна включать:

* Описание двумерного массива типа, допустимого для решения задачи
* ввод массива с клавиатуры, с помощью генератора случайных чисел
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

3. Подготовить данные для тестирования работы

4. Проверить работоспособность программы на выбранных тестах.

5. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление массивов разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила заполнения и вывода на экран элементов массива (построчно);
* нахождение максимума и минимума среди элементов массива;
* обмен данными между строками (столбцами) массива;
* преобразование двумерного массива в одномерный, создание двумерного массива из одномерного;
* правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

*Варианты работ см. Практическая работа № 6*

**Тема 8. Сортировка методом «пузырька»**

*№ 9 – сортировка выбором*

*№ 10 – сортировка вставкой*

*№ 11 – сортировка методом Хоара*

*Все программы выполнять на Си. Сортировку оформить в отдельную функцию. Сравнить скорость сортировки различных методов.*

*(ориентировочное время выполнения каждого задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач сортировки одномерных массивов различными методами.

**Задачи:** формирование компетенций ОК 1- ОК 10, ПК 1.1- 1.6

***Методика выполнения работы.***

Перед выполнением задания изучить особенности каждого метода сортировки.

Для каждой задачи необходимо выполнить следующие действия:

1. Составить блок-схему алгоритма

2. Написать программу, реализующую составленный алгоритм на языке программирования Си.

Программа должна включать:

* Описание двумерного массива типа, допустимого для решения задачи
* ввод массива с клавиатуры, с помощью генератора случайных чисел
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

3. Подготовить данные для тестирования работы

4. Проверить работоспособность программы на выбранных тестах.

5. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление массивов разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила заполнения и вывода на экран элементов массива (построчно);
* формирование одномерного массива из элементов двумерного;
* сортировки одномерного массива различными методами;

правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90

***Вариант 1***

Дана прямоугольная матрица. Сформировать из положительных элементов матрицы, расположенных в первой и последней строках, новый одномерный массив. Полученный массив сортировать в порядке убывания элементов.

Вывести:

1.двумерный массив,

2.одномерный массив до сортировки,

3. одномерный массив после сортировки.

***Вариант 2***

Задан двумерный массив. Получить одномерный массив, состоящий из минимальных элементов строк двумерного массива. полученный массив сортировать по убыванию элементов.

Вывести:

1.двумерный массив,

2.одномерный массив до сортировки,

3. одномерный массив после сортировки.

***Вариант 3***

Дана последовательность чисел А размерности N. Выбрать среди них числа больше заданного числа в последовательность В и расположить их по возрастанию. Найти номер первого и последнего членов полученной последовательности в исходной последовательности. Вывести исходную и полученную последовательности и найденные номера.

***Вариант 4***

Определить массив из 50 вещественных чисел: x[i] = 10\*Cos( i/10), i= 1, 2, . . . 50. Отсортировать массив по убыванию значений элементов. Вывести массивы до и после сортировки.

***Вариант 5***

Определить массив из 70 вещественных чисел: x[i]= i \* Sin( i/20), i= 1, 2, . . . 70. Отсортировать массив по возрастанию значений элементов.  Вывести массивы до и после сортировки.

***Вариант 6***

Ввести массив x1,x2,...,x20. Элементы, на нечетных местах, расположить в порядке возрастания, а на нечетных в порядке убывания. Вывести массивы до и после сортировки.

***Вариант 7***

Ввести массив a1,a2,...,a15 в диапазоне [-10; 10]. Требуется упорядочить его по возрастанию абсолютных значений элементов. Вывести массивы до и после сортировки.

***Вариант 8***

Ввести массив x1,x2,...,x20 в диапазоне [-10; 10]. Требуется расположить отрицательные элементы в порядке убывания. Вывести массивы до и после сортировки.

***Вариант 9***

Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить каждую строку по возрастанию элементов. Вывести исходную матрицу и результат сортировки.

***Вариант 10***

Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить все элементы матрицы таким образом, чтобы при чтении матрицы по строкам ее элементы образовывали отсортированный по убыванию массив. Вывести массивы до и после сортировки.

***Вариант 11***

Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить столбцы по возрастанию последних элементов столбцов. Вывести массивы до и после сортировки.

***Вариант 12***

Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить все элементы матрицы таким образом, чтобы при чтении матрицы по столбцам ее элементы образовывали отсортированный по возрастанию массив. Вывести массивы до и после сортировки.

***Вариант 13***

Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить все элементы матрицы таким образом, чтобы при чтении матрицы по столбцам ее элементы образовывали отсортированный по убыванию массив. Вывести массивы до и после сортировки.

***Вариант 14***

Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Разместить все отрицательные элементы по возрастанию в верхнюю область матрицы (заполняя ими матрицу по строкам), а неотрицательные – по убыванию в нижнюю область. Вывести массивы до и после сортировки.

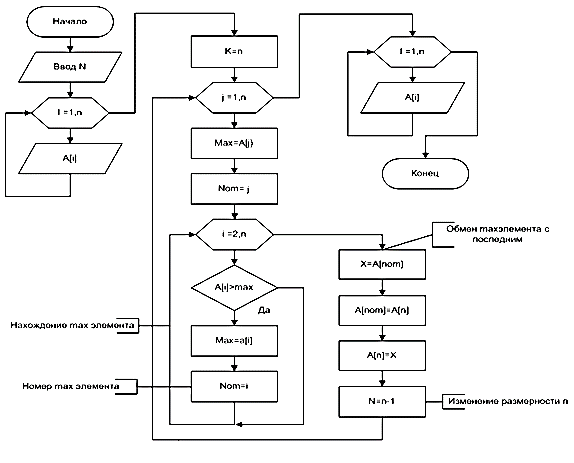
***Вариант 15***

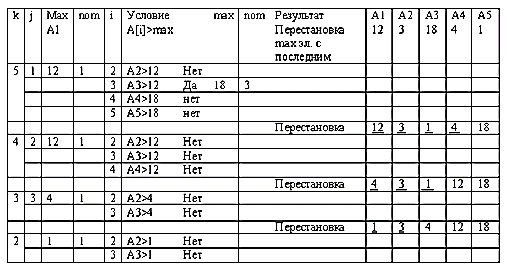
Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Разместить все максимальные элементы по убыванию в левую область матрицы (заполняя ими матрицу по столбцам), а остальные – в правую область. Вывести массивы до и после сортировки.

**Тема 9. Сортировка выбором**

***Методика выполнения работы.***

Сортировка выбором:  
           Найдем в массиве самый большой элемент и поменяем его местами с последним элементом. Повторим алгоритм поиска максимального элемента, уменьшив количество просматриваемых элементов на единицу, и поменяем его местами с предпоследним элементом. Операцию поиска проводят до полного упорядочивания элементов в массиве. Так как, происходит изменение переменной n, то в начале алгоритма ее значение необходимо сохранить (для упорядочивания по убыванию необходимо перемещать минимальный элемент).





**Тема 10. Сортировка вставкой**

***Методика выполнения работы.***

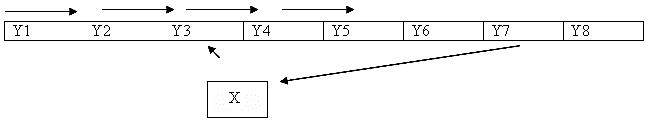
 Сортировка вставкой:  
           Прежде чем приступить к составлению блок схемы рассмотрим следующий пример.

Пусть известно, что в массиве из восьми элементов первые шесть упорядочены, а седьмой элемент нужно поставить между вторым и четвертым.

Сохраним его во вспомогательной переменной Х, а на его место запишем шестой.

Далее пятый элемент переместим на место шестого, четвертый - на место пятого, а третий на место четвертого, тем самым выполнив сдвиг элементов на одну позицию вправо.

Записав содержимое вспомогательной переменной в третью позицию, достигнем требуемого результата.

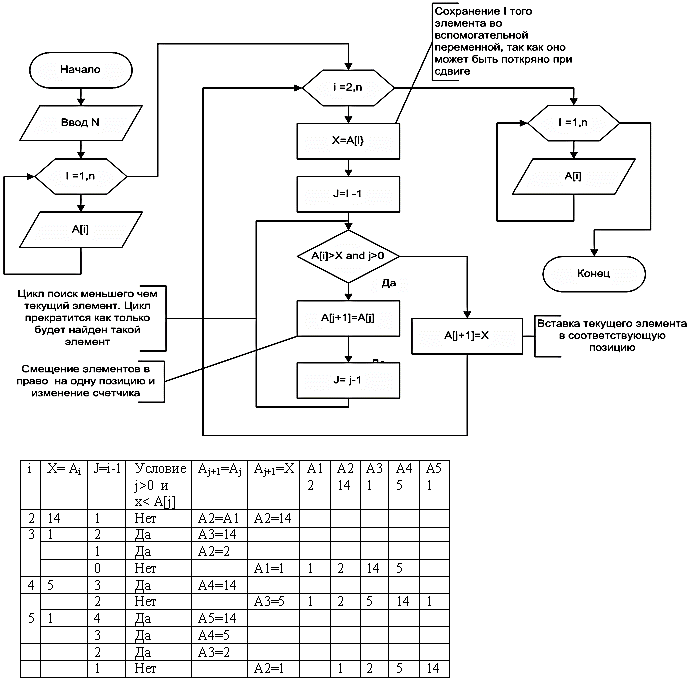


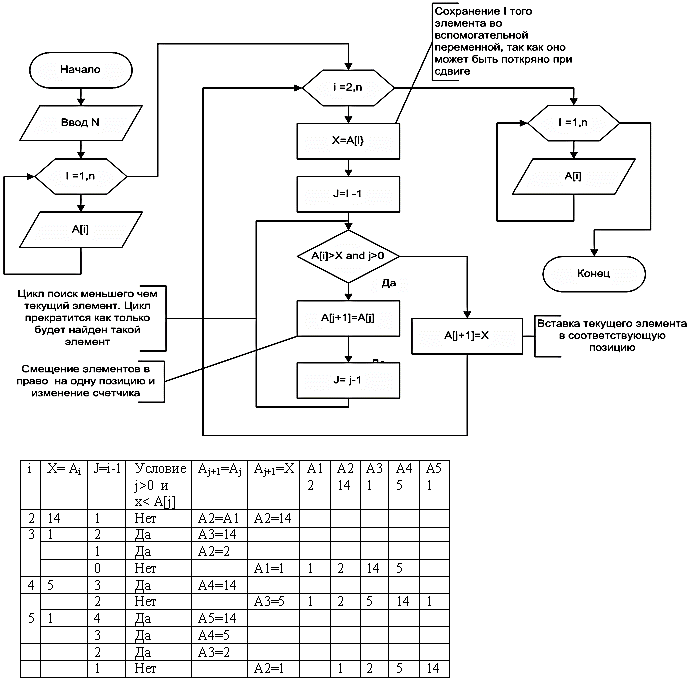
Составим блок схему алгоритма, учитывая, что описанные действия, возможно, придется выполнить неоднократно.

Организуем цикл для просмотра всех элементов массива, начиная со второго.

Сохраним значение текущего I- того элемента во вспомогательной переменной Х, так как оно может быть потеряно при сдвиге элементов, и присвоим переменной j значение предыдущего (I -1) элемента массива.

Далее движемся по массиву влево в поисках элемента, меньшего, чем текущий, и пока он не найден, сдвигаем элементы вправо на одну позицию.

Для этого организуем цикл, который прекратится, как только будет найден элемент меньше текущего. Если такого элемента в массиве не найдется и переменная j станет равной 0, это будет означать, что достигнута левая граница массива, и текущий элемент необходимо установить в первую позицию. 



**Тема 11. Метод быстрой сортировки**

***Методика выполнения работы.***

           Метод быстрой сортировки был впервые описан Ч.А.Р. Хоаром в 1962 году.

*Быстрая сортировка* – это общее название ряда алгоритмов, которые отражают различные подходы к получению критичного параметра, влияющего на производительность метода.

При общем рассмотрении алгоритма быстрой сортировки, отметим, что этот метод основывается на последовательном разделении сортируемого набора данных на блоки меньшего размера таким образом, что между значениями разных блоков обеспечивается отношение упорядоченности (для любой пары блоков все значения одного из этих блоков не превышают значений другого блока).

*Опорным (ведущим) элементом* называется некоторый элемент массива, который выбирается определенный образом. С точки зрения корректности алгоритма выбор опорного элемента безразличен. С точки зрения повышения *эффективности алгоритма* выбираться должна *медиана*, но без дополнительных сведений о сортируемых данных ее обычно невозможно получить. Необходимо выбирать постоянно один и тот же элемент (например, средний или последний по положению) или выбирать элемент со случайно выбранным индексом.

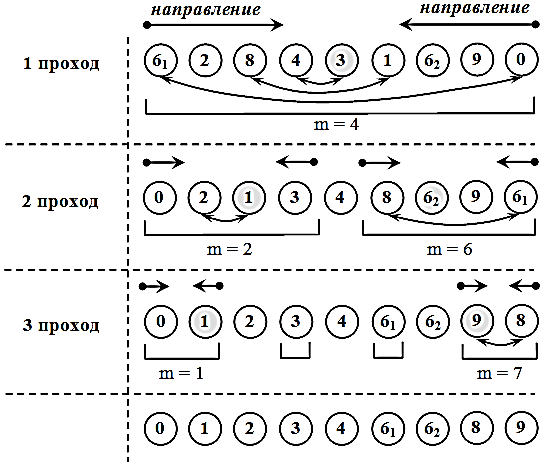
Алгоритм быстрой сортировки Хоара

Пусть дан массив x[n] размерности n.

Шаг 1. Выбирается опорный элемент массива.

Шаг 2. Массив разбивается на два – левый и правый – относительно опорного элемента. Реорганизуем массив таким образом, чтобы все элементы, меньшие опорного элемента, оказались слева от него, а все элементы, большие опорного – справа от него.

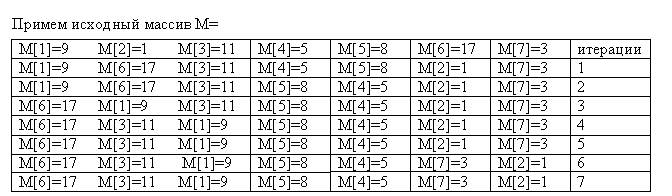
Шаг 3. Далее повторяется шаг 2 для каждого из двух вновь образованных массивов. Каждый раз при повторении преобразования очередная часть массива разбивается на два меньших и т. д., пока не получится массив из двух элементов.



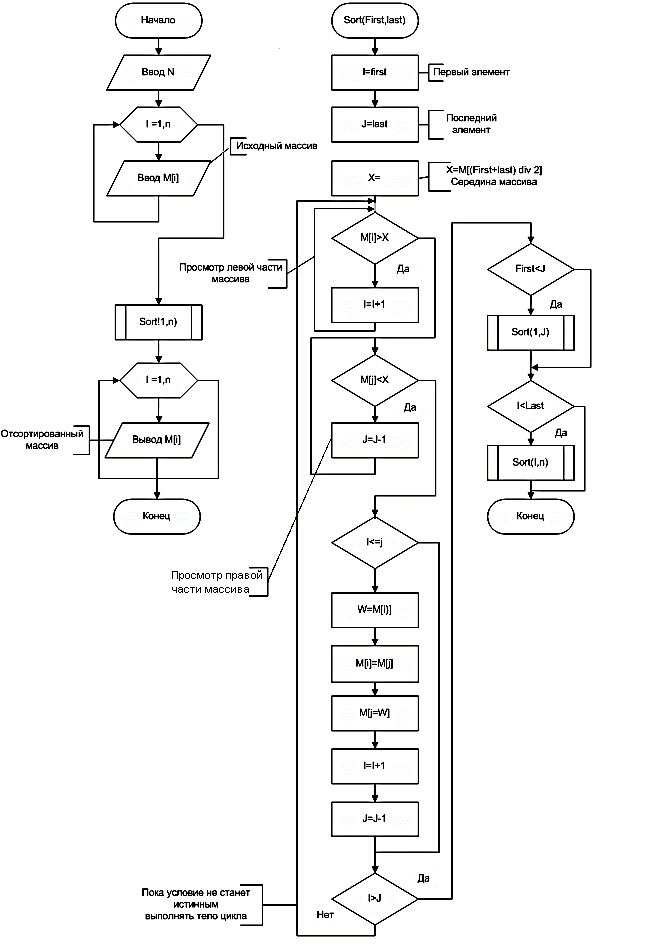
Рассмотрим пример, реализующий этот метод.

Процедура сортировки Sort имеет входные параметры First (левая граница массива – первый элемент), Last (правая граница массива - последний элемент)

X= M[(First+Last) div 2] - определяем середину массива.

           После первого вызова процедуры Sort исходный массив делится на две части.  
           Далее выполняется обмен по приведенному правилу. В результате этого получим массив из двух частей.  
           Далее рекурсивно левая часть в свою очередь дробится на две части, и вызывается подпрограмма для сортировки левой части.  
           После того как левая часть массива отсортирована, опять рекурсивно вызывается подпрограмма, в которой определяется середина данной части массива, и выполняется обмен элементов. 

 Итерация 1: Делим массив на две части. В левой части находим самый маленький элемент, в правой самый большой элемент по отношению к элементу M[x]. И меняем их местами M[2] и M[6]  
 Итерация : меняем его с элементом M[x] M[4] и M[5]  
 Итерация 3: возврат в процедуру sort(1,4) т.е. сортируется левая часть с 1 по 4 элемент M[2] и M[3]  
 Итерация 4: в левой части элементы упорядочены  
 Итерация 5: возврат в процедуру sort(4,7) т.е. сортируется правая часть с 4 по 7 элемент M[6] и M[7] и т.д.



**Тема 12. Рекурсивные алгоритмы**

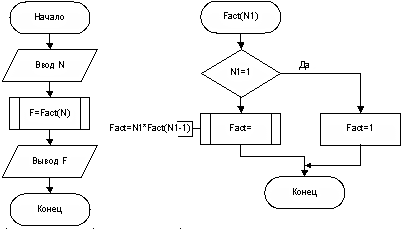
*(ориентировочное время выполнения задания — 2 часа)*

**Цель работы:** Получение практических навыков по составлению рекурсивных алгоритмов и программ.

**Задачи:** формирование компетенций ОК 1- ОК 10, ПК 1.1- 1.6

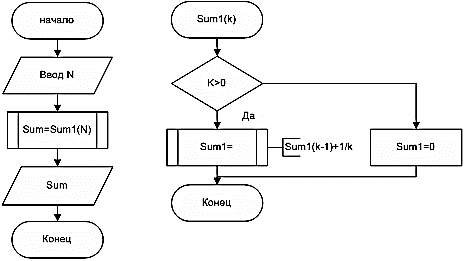
***Методика решения задач***

***Рекурсия*** - это такой способ организации вычислительного процесса, при котором процедура или функция в ходе выполнения составляющих ее операторов обращается сама к себе.  
           В основной блок-схеме блоком, содержащим выражение F = Fact(N) вызывается функция Fact с параметром – N (фактический параметр). В подпрограмме-функции вычисления факториала проверяется условие N1 = 1, Если оно выполняется, то функция Fact принимает значение равное 1. На этом выполнение программы функции заканчивается. Если условие не соблюдается, то выполняется вычисление произведения N1\*Fact(N1-1)  
(N1 - формальный параметр). Вычисление произведения носит рекурсивный характер, так как при этом осуществляется вызов функции Fact(N1-1), значение которой вычисляется функцией Fact и так до тех пор пока значение формального параметра N1 не станет равным 1.

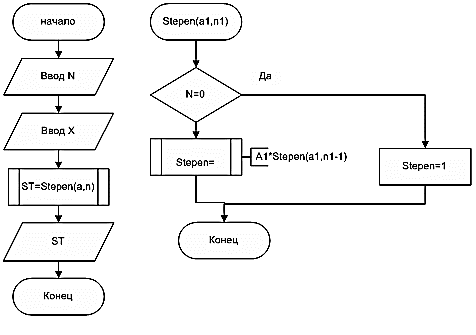


***Пример 1.*** Вычислить сумму ряда *S = 1/1 + 1/2 + 1/3 + … 1/N*

 В теле функции помещен блок *Sum1(k-1) + 1/k*, который означает, что блок **Sum1**  обращается к самому себе с новым значением.



***Пример 2.*** Классическим примером рекурсивной функции может быть возведение числа в некоторую степень, например, вычислить значение функции *Y = XN*



Практическая работа включает 2 задачи.

Для каждой задачи необходимо выполнить следующие действия:

1. Составить рекуррентную формулу

2. Составить блок-схему рекурсивного алгоритма

3. Написать программу, реализующую составленный алгоритм на языке программирования Си, выделив вычисление по рекуррентной формуле в отдельную функцию.

Программа должна включать:

* Описание данных типа, допустимого для решения задачи
* Ввод начальных значений (по необходимости) с клавиатуры
* Обработку данных в соответствии с заданным вариантом
* Вывод результата на экран

3. Подготовить данные для тестирования работы

4. Проверить работоспособность программы на выбранных тестах.

5. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования", «теория алгоритмов»:

* Понятие рекурсии;
* Правила составления рекуррентной формулы;
* Описание и вызов функции;
* Правила оформления блок-схемы алгоритма в соответствии с ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД, ГОСТ 19.701-90*.*

*Для построения рекуррентных формул воспользоваться разложениями:*

*http://www.mathprofi.ru/g/razlozhenie_funkcij_v_stepennye_ryady_clip_image119.gif*

*http://www.mathprofi.ru/g/razlozhenie_funkcij_v_stepennye_ryady_clip_image085.gif*

*http://www.mathprofi.ru/g/razlozhenie_funkcij_v_stepennye_ryady_clip_image286.gif*

*http://www.mathprofi.ru/g/razlozhenie_funkcij_v_stepennye_ryady_clip_image015.gif*

разложение тангенса в ряд Маклорена

разложение натурального логарифма в степенной ряд

*Вариант 1.*

1. S = ln(1+α)
2. Найти сумму цифр заданного натурального числа

*Вариант 2.*

1. S = eα cos(α)
2. Найти количество цифр в заданном натуральном числе

*Вариант 3.*

1. S = α m + cos(α)
2. Описать функцию C(m, n), где 0≤m≤n, для вычисления биномиального коэффициента по следующим формулам: при 0<m<n

*Вариант 4.*

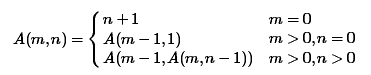
1. S = α! - cos(α)
2. Вычислить наибольний общий делитель двух натуральных чисел

*Вариант 5.*

1. S = x! - sin(x)
2. Найти число, которое образуется при записи цифр заданного натурального числа в обратном порядке.

*Вариант 6.*

1. S = 1! + 2! + ... + n!
2. Функция Аккермана *A*(*m*,*n*) определена следующим образом:



Даны два целых неотрицательных числа *m* и *n*, каждое в отдельной строке. Выведите *A*(*m*,*n*).

*Вариант 7.*

1. S = x1 + x2 + ... + xn
2. Дана монотонная последовательность, в которой каждое натуральное число k встречается ровно k раз: 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, ...

По данному натуральному n выведите первые n членов этой последовательности. Попробуйте обойтись только одним циклом for.

*Вариант 8.*

1, 2. Числа Фибоначчи задаются следующими соотношениями: *f0=f1=1; fn=fn-1+fn-2, n>1*. Напишите рекурсивную и нерекурсивную функции, вычисляющие n-e число Фибоначчи, и сравните скорость их работы. Объясните результаты сравнения.

*Вариант 9.*

1. Напишите рекурсивную функцию, вычисляющую сумму целых чисел n и m, в которой из арифметических операций используется только прибавление и вычитание единицы.

2. S = ex sin(x)

*Вариант 10.*

1. S = xm + sin(x)

2. Напишите функцию, которая вычисляет и печатает все сочетания из n первых натуральных чисел по k чисел.

*Вариант 11.*

1. S = xm + tg(x)

2. Дано n различных натуральных чисел. Определить все возможные перестановки этих чисел.

*Вариант 12.*

1. S = ex tg(x)

2. Разработать функцию для вычисления определителя матрицы порядка n (1

*Вариант 13.*

1. S = x! + sin(x)

2. Найти n-ю производную функции , построив для рекуррентное соотношение.

**Тема 13. Обработка матриц с небольшим количеством строк или столбцов**

*(ориентировочное время выполнения задания — 4 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ на Паскале для решения задач с использованием одномерных и двумерных (матриц) массивов.

**Задачи:** формирование компетенций ОК 1- ОК 10, ПК 1.1- 1.6

***Методика выполнения работы.***

Согласно предложенному варианту задания, разработать схему алгоритма и

программу для ввода исходных данных, обработки данных и вывода результатов.

Программа должна включать:

1) объявление массивов вещественных данных;

2) открытие файлов для чтения (ввода) данных и для записи (вывода) данных

3) с помощью отдельных процедур с параметрами выполнить

* ввод исходных данных (учесть три возможности ввода данных: с клавиатуры, случайным образом, из файла);
* вывод массива исходных данных (для улучшения качества просмотра предпочтение отдать табличному виду и формату с фиксированной точкой);
* обработку данных в соответствии с заданным вариантом;
* вывод результатов выполнения программы (в процессе отладки программы на экран; после отладки — в файл);

4) для управления работой программой разработать структуру меню для вызова каждой процедуры (формирование меню осуществляется средствами модуля CRT);

5) для тестирования программы сформировать исходные данные таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса предмета «Основы алгоритмизации и программирования»:

* создание простого меню средствами модуля CRT;
* объявление массивов вещественных данных и основные характеристики массивов данных (в какой последовательности элементы массивов располагаются в памяти и какой объем ОП они занимают);
* работа с текстовыми или типизированными файлами (открытие файлов для чтения/записи ASSIGN, RESET, REWRITE; чтение (запись из файла/в файл READ, WRITE; закрытие файлов CLOSE).

*Вариант 1*

1. Найти сумму и число тех элементов заданного массива Хl, Х2,…,Хn, которые попадают на заданный отрезок.

2. Для матрицы из 3 столбцов и 7 строк отпечатать номера тех строк, в которых третий элемент больше суммы двух других элементов строки, и число строк такого рода.

*Вариант 2*

1. Подсчитать по отдельности суммы (Cl и С2) и количества (Ml и М2) отрицательных и положительных элементов заданного одномерного массива.
2. Для матрицы из 2 строк и 8 столбцов отпечатать номер каждого столбца, сумма элементов которого меньше нуля, и число таких столбцов.

*Вариант 3*

1. Выделяя из заданных элементов X1,X2,...,Хn положительные элементы, для которых к тому же справедливо равенство sinXi≤0, найти число и произведение такого рода элементов
2. Для матрицы из 3 строк и 6 столбцов отпечатать номера тех столбцов, в которых первый элемент меньше второго, а второй - меньше третьего, и число таких столбцов.

*Вариант 4*

1. Выделяя из заданных элементов X1,X2,...,Хn положительные элементы, для которых к тому же справедливо равенство sinXi≥0, найти число и произведение такого рода элементов
2. Для матрицы из 2 столбцов и 10 строк отпечатать номер каждой строки, оба элемента который имеют нулевые значения, и число таких строк.

*Вариант 5*

1. Для заданного массива X1,X2,...,Хn найти среднее арифметическое CX

элементов, имеющих четные номера, и при том положительных, а для заданного массива Y1, Y2,...,Yn найти среднее арифметическое CY элементов, имеющих нечетные номера, и притом отрицательных.

2. Для матрицы из 3 столбцов и 6 строк отпечатать номер каждой строки, в которой второй элемент меньше среднего арифметического элементов этой строки, и число таких строк.

*Вариант 6*

1. При заданных элементах X1,X2,...,Хn найти по отдельности суммы Cl,С2 и количества Ml,М2 элементов, значение которых соответственно больше W и меньше -W.
2. Дана матрица из 2 строк и 10 столбцов. Первый элемент каждого столбца представляет абсциссу, а второй - ординату из 10 точек плоскости XOY. Отпечатать номера тех столбцов, которыми представлены точки первой четверти плоскости, а также количество таких столбцов.

*Вариант 7*

1. При заданных X1,X2,...,Хn и Y1, Y2,...,Yn проверяя на равенство элементы пар (Х1,Y1),(X2, Y2),..., (Хn, Уn) подсчитать число случаев равенства элементов пары; одновременно найти среднее арифметическое элементов X1,X2,...,Хn
2. Для матрицы из 3 строк и 7 столбцов отпечатать номера тех столбцов, сумма элементов которых превышает заданную величину, и число таких столбцов.

*Вариант 8*

1. При заданных X1,X2,...,Хn и Y1, Y2,...,Yn проверяя на равенство элементы пар (Х1,Y1),(X2, Y2),..., (Хn, Уn) подсчитать число случаев неравенства элементов пары; одновременно найти среднее арифметическое элементов Y1, Y2,...,Yn
2. Для матрицы из 3 столбов и 6 строк отпечатать номер каждой строки, в   
   которой не менее 2 элементов имеют нулевое значение, и число таких строк.

*Вариант 9*

1. При заданной величине А и заданных элементах X1,X2,...,Хn, Y1, Y2,...,Yn определить число произведений вид ХiYi, удовлетворяющих условию ХiYi<А и сумму таких произведений.
2. Для матрицы из 2 строк и 10 столбцов отпечатать номер каждого столбца, знаки элементов которого не совпадает, и число таких столбцов.

*Вариант 10*

1. Найти среднее арифметическое тех элементов одномерного массива Х, значение которых не превышают Xi, включая и сам элемент Xi. Найти также среднее арифметическое всех элементов данного массива.
2. Для матрицы из 3 строк и 7 столбцов отпечатать номер каждого столбца, в котором значение его наибольшего элемента оказалось меньше заданной величины, и число таких элементов.

*Вариант 11*

1. При заданных абсциссах X1,X2,...,Хn и ординатах Y1, Y2,...,Yn точек плоскости ХОУ, подсчитать, сколько из них находится в пределах круга заданного радиуса R с центром в начале координат, а также среднее арифметическое расстояний от начала координат для всех заданных точек.
2. Для матрицы из 2 столбцов и 10 строк отпечатать номер каждой строки, элементы которой имеют совпадающие значения, и число таких строк.

*Вариант 12*

1. Найти сумму и число тех элементов заданного массива X1,X2,...,Хn каждый из которых во - первых, больше элемента с тем же номером из другого заданного массива Y1, Y2,...,Yn а, во-вторых, положителен.
2. Для матрицы из 2 столбцов и 9 строк отпечатать номер каждой строки, квадрат первого элемента которой меньше абсолютного значения второго элемента строки, и число таких строк.

**Тема 14. Обработка массивов переменной длины.**

*(ориентировочное время выполнения задания — 4 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач с использованием одномерных и двумерных (матриц) массивов переменной длины.

***Методика выполнения работы.***

Дана матрица, состоящая из n строк и n столбцов или одномерные массивы из n элементов каждый. Согласно предложенному варианту задания, разработать схему алгоритма и программу на Си для получения и вывода указанных в условии результатов и самих массивов, если изменялись какие-либо их элементы.

Алгоритм любой задачи может быть составлен с использованием **единственного двукратного цикла.**

Программа должна включать;

1) объявление массивов вещественных данных;

2) открытие файлов для чтения (ввода) данных и для записи (вывода) данных;

3) с помощью отдельных функций с параметрами выполнить:

ввод исходных данных (учесть три возможности ввода данных: с клавиатуры, случайным образом, из файла);

вывод массива исходных данных (для улучшения качества просмотра предпочтение отдать табличному виду и формату с фиксированной точкой);

обработку данных в соответствии с заданным вариантом;

вывод результатов выполнения программы (в процессе отладки программы - на экран; после отладки — в файл);

4) для управления работой программой разработать структуру меню для вызова каждой функции;

5) для тестирования программы сформировать исходные данные таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма

6) оформить отчет в MS Word

Для выполнения работы необходимо знание следующих теоретических вопросов из курса предмета «Основы программирования»

* строение функции и ее вызов;
* глобальные и локальные переменные;
* формальные и фактические параметры;
* типы формальных параметров;
* основные средства и приемы отладки программ, использующих функции.

*Вариант 1*

1. Найти среднее арифметическое неотрицательных элементов матрицы, а также подсчитать, сколько таких элементов в каждой отдельно взятой строке матрицы.
2. Если ни один из столбцов матрицы А, не содержит два и более равных нулю элемента найти сумму элементов матрицы, лежащих на главной диагонали и выше ее.

*Вариант 2*

1. Изменить все строки матрицы, в которых отрицателен элемент главной диагонали: к каждому элементу i-й строки прибавляется элемент Тi из заданного массива Т1,Т2,...,Тn. Подсчитать число измененных строк матрицы.
2. Если сумма двух первых строк матрицы А меньше суммы элементов двух последних ее строк, изменить матрицу А, прибавив к элементам каждой строки заданные элементы Х1, X2, ..., Х5

*Вариант 3*

1. Найти среднее арифметическое тех элементов матрицы, каждый из которых больше находящегося с ним в одной строке элемента главной диагонали, и сумму элементов главной диагонали.

2. Если наибольший элемент матрицы А находится выше главной диагонали, найти сумму элементов матрицы, лежащих ниже главной диагонали.

*Вариант 4*

1. Найти среднее арифметическое элементов матрицы и сумму элементов тех строк матрицы, в которых отрицателен элемент главной диагонали.
2. Найти среднее арифметическое матрицы А, и, если матрица А не содержит ни одного отрицательного элемента, изменить элементы матрицы путем вычитания из них среднего арифметического.

*Вариант 5*

1. Получить массив Х1,Х2,...,Хn по правилу Хi=l, если каждый элемент i-ro столбца, кроме первого и последнего элементов, меньше полусуммы двух соседних элементов (предыдущего и последующего), иначе Хi=0.
2. Если разность максимального и минимального элемента матрицы А превышает заданную величину Р, заменить в матрице А все отрицательные элементы нулями, а положительные единицами.

*Вариант 6*

1. Получить массив C1,C2,...,Сn по правилу: Сi=0, если все элементы i-го столбца матрицы равны 0, иначе Сi =1. Найти также сумму всех элементов матрицы.
2. Если среднее арифметическое элементов С1, С2, ..., С7 больше минимального элемента матрицы, уменьшить на величину последнего каждый из элементов С1, C2,..., С7

*Вариант 7*

1. Изменить матрицу, заменив каждый отрицательный элемент, лежащий выше главной диагонали, его абсолютной величиной. Найти также сумму элементов главной диагонали.
2. Если все элементы главной диагонали матрицы А отрицательны, разделить все элементы матрицы на максимальный по абсолютной величине элемент матрицы.

*Вариант 8*

1. Получить массив Х1,Х2,...,Хn и найти произведение элементов матрицы;

элемент Хi, представляет среднее арифметическое положительных элементов в i-й строке матрицы, или равен 0, если положительных элементов не обнаружено.

2. Если среднее арифметическое матрицы А положительно, задать элементам С1, C2, ..., Сi; значения тех элементов матрицы А, которые больше этого среднего арифметического.

*Вариант 9*

1. Изменить матрицу, прибавляя к каждому отрицательному элементу значение предыдущего элемента той же строки. Если отрицателен первый элемент строки, прибавить к нему значение последнего элемента строки.
2. Дана матрица Р с двумя строками и десятью столбцами, каждым столбцом которой задана абсцисса и ордината одной из десяти точек плоскости. Если нет ни одной пары точек, расстояние между которыми меньше заданной величины R, заменить на нуль в матрице Р все отрицательные абсциссы точек, увеличив ординаты этих точек на R

*Вариант 10*

1. Найти сумму элементов матрицы и подсчитать число ее строк, в пределах каждой из которых элементы упорядочены по возрастанию: Аi1<Аi2...<Аik.
2. Даны две последовательности: C1, C2,..., C10; P1, P2,..., P10. Если наибольший элемент первой последовательности меньше наименьшего элемента второй последовательности, задать значения элементам Х1, Х2, ..., Х10 новой последовательности значения C1, С2, ..., C10, а элементам X11, X12, ..., X20 значения Р1, P2, ., Р10.

*Вариант 11*

1. Найти среднее арифметическое отрицательных элементов матрицы, лежащей ниже главной диагонали, и среднее арифметическое всех элементов главной диагонали.
2. Даны две последовательности; С1, C2, ..., С7, Р1, P2, ..., Р7. Если каждый элемент первой последовательности меньше суммы элементов второй, найти при каких значениях i, j максимально значение выражения Сi/ (Pj +Ci2 )

*Вариант 12*

1. Дан массив C1,С2,...,Сn. Изменить матрицу, увеличив каждый элемент, который меньше элемента главной диагонали, находящегося с ним в одной строке, на Сi где i — номер строки. Подсчитать общее число измененных элементов.
2. Если исходная последовательность С1, С2, ...,С10 не содержит ни одного элемента, значение которого совпадает со значением какого-либо элемента последовательности P1, P2 … Р10, задать значения элементам Х1, X2,..., Х7 по правилу Xi=maх(Сi+ Рi).

**Тема 15 . Сложное сочетание данных разной структуры.**

*(ориентировочное время выполнения задания — 4 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач вычисления значений элементов одномерного массива по заданной формуле.

***Методика выполнения работы.***

Дана переменная В и матрица А, состоящая из m строк и m столбцов. Элементы С образуют одномерный массив из m элементов.

Требуется составить алгоритм и программу вычисления значений элементов одномерного массива Х1,Х2,..., X m (когда вычисляется Хi, Р равно Х i -1, для первого элемента массива Р=1).

S — сумма элементов главной диагонали матрицы А, вычисляемая в алгоритме.

Составить программу на Паскале и на Си.

Программа должна включать:

1) объявление массивов вещественных данных;

2) открытие файлов для чтения (ввода) данных и для записи (вывода) данных;

3) с помощью отдельных процедур (функций) с параметрами выполнить:

4) ввод исходных данных (учесть три возможности ввода данных: с клавиатуры, случайным образом, из файла);

5) вывод массива исходных данных (для улучшения качества просмотра предпочтение отдать табличному виду и формату с фиксированной точкой);

6) обработку данных в соответствии с заданным вариантом;

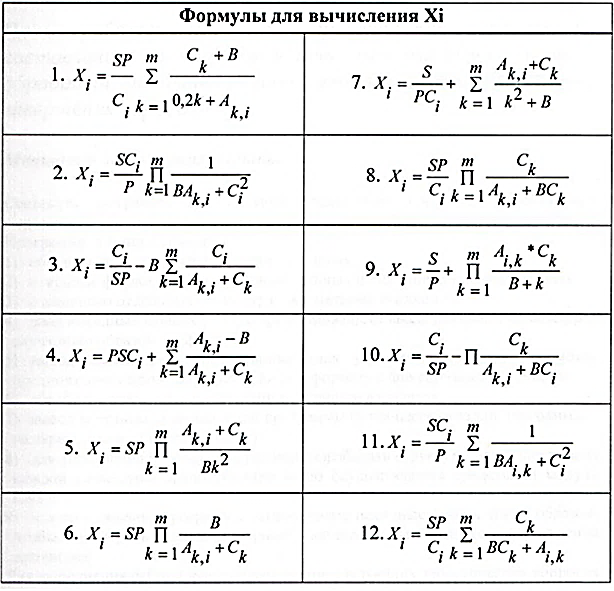
7) вывод результатов выполнения программы (в процессе отладки программы на экран; после отладки — в файл);

8) для управления работой программой разработать структуру меню для вызова каждой процедуры (функции) (в Паскале формирование меню осуществляется средствами модуля CRT);

9) для тестирования программы сформировать исходные данные таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса предмета "Основы алгоритмизации и программирования":

* объявление переменных разных типов и их основные характеристики (необходимый объем ОП);
* правила выполнения арифметических выражений;
* правила построения и выполнения сложных циклов;
* правила обработки исключительных ситуаций (деление на ноль).



**Тема 16. Сложное условие завершения цикла при работе с матрицами.**

*(ориентировочное время выполнения задания —* 4 *часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ для решения задач обработки матрицы с использованием сложного условия завершения обработки.

***Методика выполнения работы.***

Составить программу на двух языках программирования, предусматривая завершение обработки матрицы при выполнении указанного в задании условия.

Оформить отчет по практической работе в MS Word.

Программа должна включать:

1) объявление массивов вещественных данных;

2) открытие файлов для чтения (ввода) данных и для записи (вывода) данных;

3) с помощью отдельных функций с параметрами выполнить:

4) ввод исходных данных (учесть три возможности ввода данных: с клавиатуры, случайным образом, из файла);

5) вывод массива исходных данных (для улучшения качества просмотра предпочтение отдать табличному виду и формату с фиксированной точкой);

6) обработку данных в соответствии с заданным вариантом;

7) вывод результатов выполнения программы (в процессе отладки программы - на экран; после отладки — в файл);

8) для управления работой программой разработать структуру меню для вызова каждой функции;

9) для тестирования программы сформировать исходные данные таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса предмета «Основы программирования»:

* объявление переменных разных типов и их основные характеристики (необходимый объем ОП);
* правила выполнения арифметических выражений;
* правила построения и выполнения сложных циклов;
* правила обработки исключительных ситуаций (деление на ноль).

*Вариант 1*

Дана матрица А с 2 строками и 10 столбцами. В цикле, начиная с 1-го столбца матрицы, проверять знак произведения двух элементов каждого столбца и печатать значение произведения при положительном его знаке, заканчивая цикл при выявлении отрицательного знака произведения элементов какого-либо столбца.

*Вариант 2*

Дана матрица А с 2 строками и 10 столбцами. Печатать суммы элементов каждого   
столбца, начиная с первого, пока не встретится столбец с равными элементами.

*Вариант 3*

Дана матрица А с 10 строками и 2 столбцами. Накапливать в цикле сумму разностей первого и второго элемента каждой строки, начиная с разности элементов 1-й строки, и печатать значения нарастающей суммы, заканчивая цикл, как только значение суммы превысит заданную величину Т.

*Вариант 4*

Дана матрица А с 2 строками и 10 столбцами. Первый элемент каждого столбца представляет абсциссу, а второй — ординату одной из 10 точек а плоскости XOY.   
Номер столбца рассматривать как номер точки. Начиная с первой точки, в цикле   
печатать расстояния точек от начала координат, пока эти расстояния меньше   
заданной величины С, при превышении которой цикл заканчивается.

*Вариант 5*

Дана матрица А с 2 строками и 10 столбцами. Первый элемент каждого столбца рассматривается как диаметр внутренней, а второй — как диаметр внешней окружности кольца. В цикле печатать площади колец, начиная с кольца, представленного первым столбцом, заканчивая цикл, если (по ошибке) второй элемент какого-либо столбца оказывается меньше первого.

*Вариант 6*

Дана матрица А с 2 строками и 10 столбцами. Вычислить отношение (Т) элементов 1-го столбца, а затем, начиная со второго столбца, в цикле осуществить проверку, пропорциональны ли элементы 1-й и 2-й строк, для чего вычислять отношения элементов каждого столбца и сравнивать их с Т, заканчивая цикл, если обнаружится, что какое-либо из них не равно Т. По ходу проверки печатать значения элементов.

*Вариант 7*

Дана матрица А с 3 столбцами и 7 строками. Содержание каждой ее строки трактуется как длины сторон одного из 7 треугольников. Начиная с 1-й строки, в цикле рассматривать элементы отдельных строк и, пока элементы в строках соразмерны в плане построения треугольника с данными сторонами (любой из них меньше суммы двух других), печать площади треугольников, иначе цикл оканчивать.

*Вариант 8*

Дана матрица А с 2 столбцами и 10 строками Первый элемент каждой строки представляет абсциссу, а второй — ординату одной из 10 заданных точек. Начиная с точки, заданной 1-й строкой, в цикле суммировать по отдельности абсциссы и ординаты точек и печатать последовательные значения сумм, заканчивая цикл при выявлении точки, у которой какая-либо из координат отрицательны,

*Вариант 9*

Дана матрица А с 2 столбцами и 10 строками. Каждая строка задает параметры

кольца: 1-й элемент строки — внутренний радиус; 2-й элемент — внешний радиус кольца. В цикле осуществить проверку возможности последовательного вложения колец в порядке их задания строками матрицы, начиная с 1-й строки. Печатать по ходу проверки внешние радиусы колец, заканчивая цикл, если внешний радиус очередного кольца оказался больше внутреннего радиуса предыдущего (вложение оказалось невозможным).

*Вариант 10*

Дана матрица А с 2 строками и 10 столбцами. Первый элемент каждого столбца представляет абсциссу, а второй — ординату одной из 10 заданных точек. Осуществить в цикле проверку факта расположения всех 10 точек на прямой, проходящей через начало координат (в этом случае имеет место пропорциональная зависимость абсцисс и ординат точек), заканчивая цикл, если указанный факт не подтверждается. В процессе проверки печать номера рассматриваемых столбцов.

*Вариант 11*

Дана матрица А с 2 строками и 10 столбцами, которая содержит данные о движении поезда на отдельных участках дистанции: 1-й элемент каждого столбца представляет скорость, а 2-й — длину соответствующего участка (считать скорость на участке постоянной). Начиная с 1-го столбца, вычислять в цикле отрезки времени, за которые будут пройдены участки, и печатать их нарастающую сумму, прекращая цикл, если выявляется отрицательное значение элемента в каком-либо столбце (ошибка в данных).

*Вариант 12*

Дана матрица А с 2 столбцами и 10 строками. Первый элемент каждой строки представляет показание одного из 10 датчиков, а второй — «критическое» значение соответствующего параметра, измеряемого датчиком. Начиная с 1-й строки матрицы, в цикле печатать показания датчиков, прекращая вывод показаний и заканчивая цикл, если зафиксировано повышение критического значения параметра.

**Тема 17. Разработка алгоритмов и программ с выбором структуры данных.**

*(ориентировочная время выполнения задания — 4 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ на языке Си с выбором структуры данных.

***Методика выполнения работы.***

Составить три варианта алгоритма решения задачи с одним циклом. Исходные данные каждой задачи, названные в условии заданной матрицей А, должны быть представлены в первом варианте алгоритма — двумерным массивом, во втором — совокупностью одномерных массивов (2-3 массива), в третьем двумя или тремя простыми переменными, значения которых следует изменять перед очередным выполнением тела цикла. Провести сравнение полученных вариантов. Программа должна включать:

1) объявление массивов вещественных данных;

2) открытие файлов для чтения (ввода) данных и для записи (вывода) данных;

3) с помощью отдельных функций с параметрами выполнить:

4) ввод исходных данных (учесть три возможности ввода данных: с клавиатуры, случайным образом, из файла);

5) вывод массива исходных данных (для улучшения качества просмотра предпочтение отдать табличному виду и формату с фиксированной точкой);

6) обработку данных в соответствии с заданным вариантом;

7) вывод результатов выполнения программы (в процессе отладки программы - на экран; после отладки — в файл);

8) для управления работой программой разработать структуру меню для вызова каждой функции;

9) для тестирования программы сформировать исходные данные таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма.

Для выполнения работы необходимо знание следующих теоретических вопросов изкурса предмета «Основы программирования»:

объявление переменных разных типов и их основные характеристики (необходимый объем ОП);

правила выполнения арифметических выражений;

правила построения и выполнения сложных циклов;

правила обработки исключительных ситуаций.

*Вариант 1*

Дана матрица А с 2 столбцами и 50 строками. 1-й элемент каждой строки представляет действительную, а 2-й — мнимую часть одного из 50 комплексных чисел.Определить номера тех строк, которые представляют комплексные числа с модулем, большим 1.

*Вариант 2*

Дана матрица А с 3 строками и 25 столбцами. 1-й элемент каждого столбца представляет массу, а 2-й и 3-й — абсциссу и ординату одной из 25 материальных точек. Определить координаты тех точек, масса которых отличается от "эталонной" массы не более, чем на 10 %.

*Вариант 3*

Дана матрица А с 2 столбцами и 30 строками. Определить номера тех строк, у   
которых знак произведения их элементов отрицателен.

*Вариант 4*

Дана матрица А с 2 строками и 70 столбцами. Определить номера тех столбцов, у которых сумма их элементов отлична от нуля.

*Вариант 5*

Задана матрица А с 2 столбцами и 20 строками. 1-й элемент каждой строки представляет абсциссу, а 2-й — ординату одной из 20 точекна плоскости. Определить координаты тех точек, которые расположены в 1-й четверти координатной плоскости.

*Вариант 6*

Задана матрица А с 2 строками и 35 столбцами. Определить номера тех столбцов, элементы которых имеют совпадающие значения.

*Вариант 7*

Задана матрица А с 3 столбцами и 25 строками. Каждая строка матрицы содержит длины сторон одного из 25 треугольников. Определить номера тех строк, которые содержат длины сторон, не допускающие построение треугольника.

*Вариант 8*

Задана матрица А с 2 строками и 20 столбцами, 1-й элемент каждого столбца представляет диаметр внутренней, а 2-й — диаметр внешней окружности одного из20 колец. Определить внешние диаметры тех колец, площадь которых больше площади прямоугольника с заданными сторонами Р и Т.

*Вариант 9*

Задана матрица А с 2 столбцами и 45 строками. Определить номера тех строк, оба элемента которых имеют нулевые значения.

*Вариант 10*

Задана матрицаА с 2 строками и 60 столбцами. Определите элементы первой

строки, которые больше находящихся с ними в одном столбце элементов второй строки.

*Вариант 11*

Задана матрица А с 3 столбцами и 80 строками. Определить номера тех строк, знаки элементов которых не совпадают.

*Вариант 12*

Задана матрица А с 2 строками и 70 столбцами. 1-й элемент каждого столбца

представляют абсциссу, а 2-й — ординату одной из 70 точек на плоскости.

Определить среднее значение расстояний от начала координат тех точек, которые лежат в левой полуплоскости.

**Тема 18. Обработка символьной информации.**

*(ориентировочное время выполнения задания — 4 часа)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ обработки символьной информации.

*Методика выполнения работы.*

Дан текст. Текст разбит на слова. Предполагается, что слово — это группа символов, не содержащая пробелов и отделенная пробелами от других слов.

Способ описания текста: массив символов, строка, текстовый файл.

Составить программу на языке Си для обработки текстовой информации по поставленному условию, которая должна включать:

1) объявление необходимых переменных и массивов;

2) открытие файлов для чтения (ввода) данных и для записи (вывода) данных;

3) с помощью отдельных подпрограмм с параметрами выполнить:

4) ввод исходных данных (учесть две возможности ввода данных: с клавиатуры и из файла);

5) вывод исходных данных для просмотра;

6) обработку данных в соответствии с заданным вариантом;

7) вывод результатов выполнения программы (в процессе отладки программы - на экран; после отладки — в файл);

8) для управления работой программой разработать структуру меню для вызова каждой функции;

9) для тестирования программы сформировать исходные данные таким образом, чтобы проверить каждый вариант альтернативы каждого разветвления алгоритма.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса предмета «Основы программирования»:

объявление переменных разных типов и их основные характеристики (необходимый объем ОП);

правила выполнения выражений, включающих строковые данные;

правила построения и выполнения сложных циклов;

правила обработки исключительных ситуаций.

*Вариант 1*

1. В заданном тексте вычислить количество слов и распечатать их по одному в строке.
2. Дана непустая последовательность строчных латинских букв, за которой следует точка. Построить и напечатать в алфавитном порядке множество, состоящее из всех букв, встретившихся в последовательности.

*Вариант 2*

1. В заданном тексте подсчитать количество вхождений каждого из заданного символа.
2. Дана непустая последовательность строчных букв английского алфавита, организованная в слова, разделенные пробелом. За последним словом следует точка. Составить и напечатать множество, состоящее из согласных букв, входящих более чем в одно слово.

*Вариант 3*

1. В заданном тексте определить целое число, состоящее из наибольшего количества цифр.
2. Дана непустая последовательность заглавных латинских букв, за которой следует точка. Построить и напечатать в алфавитном порядке множество, состоящее из букв А..F и Х..Z, встретившихся в последовательности.

*Вариант 4*

1. Определить, содержит ли данный текст символы, отличные от букв и пробела.
2. Дана непустая последовательность символов, за которой следует пробел. Построить и напечатать множество, состоящее из цифр 4..8 и знаков арифметических действий, встретившихся в последовательности.

*Вариант 5*

1. В заданном тексте подсчитать количество предложений, в которые входит заданное слово.
2. Дана непустая последовательность заглавных латинских букв как слова, разделенные пробелом. За последним словом следует точка. Составить и напечатать в порядке, обратном алфавитному, множество, состоящее из всех согласных букв, входящих в третье по счету слово.

*Вариант 6*

1. Распечатать все однокоренные слова в заданном тексте (корень задается).
2. Дана непустая последовательность символов, за которой следует пробел. Построить и напечатать множество, состоящее из знаков препинания (.,;:!?), встретившихся в последовательности.

*Вариант 7*

1. Выбрать из заданного текста слова заданной длины и напечатать их по одному в каждой строке.
2. Дана непустая последовательность цифр и строчных латинских букв, за которой следует точка. Определить, каких букв — гласных (а, е, i, о, u) или согласных больше в этом множестве.

*Вариант 8*

1. Заменить все строчные буквы латинского алфавита в исходной строке текста на соответствующие большие (заглавные) буквы.
2. Дана непустая последовательность строчных латинских букв в виде слов, разделенных запятыми. За последним словом следует точка. Составить и напечатать в порядке, обратном алфавитному, множество, состоящее из гласных букв, которые входят в каждое слово.

*Вариант 9*

1. В тексте в каждой строке осуществить перестановку символов таким   
   образом, чтобы первый символ стал последним, второй предпоследним и т.д.
2. Дана непустая последовательность строчных латинских букв, за которой следует точка. Построить и напечатать множество, состоящее из первых вхождений букв в последовательность, сохраняя их взаимный порядок.

*Вариант 10*

1. В заданном тексте определить слово с наибольшим вхождением заданного символа.
2. Дана непустая последовательность строчных латинских букв, за которой следует точка. Составить и напечатать в алфавитном порядке множество, состоящее из букв, входящих в последовательность не менее двух раз.

*Вариант 11*

1. Изменить текст, удалив из него все повторные вхождения слов.
2. Дана непустая последовательность строчных и заглавных латинских букв, организованная в слова, разделанные пробелом или точкой. За последним словом — восклицательный знак. Построить множество букв последовательности. Определить правильность написания (с заглавной ли буквы) "предложений" — слов, следующих за точкой.

*Вариант 12*

1. В заданном тексте каждую запятую прижать к предыдущему слову и отделить от следующего пробелом.
2. Дан непустой текст из заглавных латинских букв, за которым следует точка. Определить, упорядочены ли эти буквы по алфавиту.

**Тема 19. Программирование баз данных с использованием комбинированного типа данных.**

*(ориентировочное время выполнения задания — 8 часов)*

**Цель работы:** Получение практических навыков по составлению алгоритмов и программ обработки структурированных данных на языке программирования Паскаль.

***Методика выполнения работы.***

Разработать программу для обработки данных типизированного файла.   
Программа должна включать процедуры:

1) ввода данных массива записей и создание типизированного файла;

2) вывод шапки таблицы;

3) вывод данных одной записи;

4) дополнение файла новыми записями;

5) просмотр данных типизированного файла,

6) модификация ряда записей;

7) поиск записей файла с выводом найденных записей или сообщения о неуспешном поиске (поиск осуществить по одному поисковому признаку и по любому сочетанию заданных поисковых признаков);

8) удаление записей из типизированного файла (после поиска по заданию преподавателя).

Исходные данные для создания типизированного файла берутся из текстового файла (Soz.dat) или вводятся с клавиатуры.

Тестирование процедур поиска, корректировки и удаления производить с помощью данных, расположенных в текстовом файле (Poiskl.dat, Poisk2.dat, Коr.dat, Udal.dat). При тестировании проверить:

1) правильность обработки записей, расположенных в файле: первой, последней и в середине;

2) правильность ответа из процедур при попытке:

а) обработки записи, не существующей в файле;

б) при отсутствии поискового признака.

***Теоретическая часть.***

Типизированный файл - последовательность элементов одного типа.

Описание файлового типа имеет синтаксис:

file of < тип элементов>;

Примеры описания переменных файлового типа:

Type  
  Student = record  
      Name, SerName : string;  
      YearOld : byte;  
      Sessia : array [1..10] of byte;  
    end;  
Var  
  VFile1 : file of char;  
  VFile2 : file of Student;  
  VFile3 : file of string;

Удаление записей из типизированного файла в зависимости от задачи и от числа удаляемых записей при одном обращении кпроцедуре удаления можно осуществлять несколькими способами.

1) Удаление записей с пересылкой данных после каждого удаления.

В процессе удаления для каждой удаляемой записи производится поиск удаляемой записи в основном типизированном файле и пересылка остающихся записей в рабочий (типизированный файл), а затем обратная пересылка неудаленных записей (в случае успешного поиска удаляемой записи) из рабочего файла в основной типизированный файл.

2) Удаление записей с пересылкой данных и с переименованием файла после удаления каждой записи. В этом способе также используются два файла, основной и рабочий. В процессе удаления по каждой удаляемой записи производится вначале поиск удаляемой записи и пересылка неудаляемых записей из основного типизированного файла в рабочий, затем исходный файл удаляется, а рабочий файл получает имя исходного.

3) Удаление записей с пометкой удаляемых, однократной пересылкой данных и с переименованием файла. В этом способе также используются два файла, основной и рабочий. В   
процессе удаления производится:

- пометка каждой удаляемой записи (значению поля, по которому производится поиск удаляемой записи, присваивают О или ' ');

- пересылка всех неудаляемых записей из исходного файла в рабочий;

- удаление исходного файла;

- переименование рабочего файла.

4) Удаление со сдвигом записей и с усечением файла.

В этом способе используется только один типизированный файл. В процессе удаления по каждой удаляемой записи производится:

- поиск удаляемой записи;

- сдвиг всех записей, расположенных после удаляемой записи, к началу файла.

После удаления всех требуемых записей производится усечение файла.

5) Удаление с перемещением записей и с усечением файла.

В этом способе используется только один типизированный файл. В процессе удаления по каждой удаляемой записи производится:

- поиск удаляемой записи;

- в случае успешного поиска — копирование последней записи файла на место удаляемой, и уменьшение количества остающихся записей файла на единицу.

После удаления всех требуемых записей производится усечение файла.

Процедуры и функции работы с файлами:

***Erase(F)****- уничтожение внешнего файла.*

***Rename(F,NewName)****- переименование внешнего файла.*

***IOResult****- результат последней операции ввода-вывода.*

**Truncate**(<имя файловой переменной>) *- удаляет все компоненты в файле, начиная с того, над которым находиться указатель.*

Для выполнения работы необходимо знание следующих теоретических вопросов   
из курса предмета «Основы программирования»:

описание записей и массивов записей;

обращение к элементам записей и элементам массивов записей;

объявление текстовых и типизированных файлов;

процедуры для открытия и закрытия файлов;

процедуры и функции для работы с типизированными файлами (Erase, ReName, IoResult, Truncate).

*Вариант 1*

Багаж авиапассажира характеризуется фамилией пассажира, номером рейса, датой вылета, количеством вещей и общим весом вещей. Вывести фамилии пассажиров, багаж которых состоит из одной вещи весом не менее 30 кг, и вычислить средний вес багажа всех пассажиров.

*Вариант 2*

Расписание экзаменов содержит следующую информацию: номер студенческой группы, дата и время экзамена, номер аудитории, наименование дисциплины и фамилия преподавателя. Вывести даты, по которым занят конкретный преподаватель.

*Вариант 3*

Сведения об автомобиле состоят из его марки, номера, фамилии владельца и отметки о прохождении техосмотра. Вывести фамилии владельцев и номера автомобилей определенной марки, не прошедших техосмотр.

*Вариант 4*

В расписании поездов указано: номер поезда, пункт отправления, пункт назначения, время отправления, время в пути, цена билета. Вывести информацшо о поездах, следующих из Москвы в Санкт-Петербург, время отправления которых с 7.00 до 9.00.

*Вариант 5*

В анкетах сотрудников некоторого учреждения имеются следующие сведения: фамилия, имя, отчество, год рождения, должность, стаж, образование (мастер, инженер, рабочий и т.д.). Вывести фамилии, имена и отчества мастеров с высшим образованием и вычислить их средний стаж.

*Вариант 6*

Имеются следующие сведения о книгах: фамилии и инициалы автора, название, год издания Вывести сведения о всех книгах, изданных с 1990 г., в названии которых содержится слово «информатика».

*Вариант 7*

Сведения состоят из фамилий сотрудников, должности, стажа работы и их телефонов. Вывести фамилии сотрудников, номера телефонов которых начинаются с цифр 361,362 или 273.

*Вариант 8*

Сведения о компьютерах некоторого учреждения состоят из марки компьютера, даты приобретения, номера комнаты, где находится компьютер, и фамилии ответственного лица. Вывести сведения о компьютерах PentiumIV, приобретенных в 2000 году.

*Вариант 9*

Имеются сведении об экспортируемых товарах: наименование, страна-экспортер, срок поставки и количество товара. Вывести страны, в которые должен быть поставлен данный товар до 1 июля.

*Вариант 10*

Сведения о заказах некоторого предприятия бытового обслуживания состоит из номера заказа, фамилии заказчика и даты (число, месяц, год) его исполнения. Вывести все весенние заказы.

*Вариант 11*

О детях, поступающих в спортивную секцию, собраны следующие сведения: фамилии, имя, год рождения, вес, рост. Вывести сведения о детях от 10 до 12 лет, рост которых не менее 155 см, а вес не более 45 кг.

*Вариант 12*

Имеются сведения о районных библиотеках города: номер, тип (детская, взрослая), район, адрес, часы работы. Вывести адреса и часы работы всех детских библиотек данного района.

**Тема 20. Построение графика функции**

*(ориентировочное время выполнения задания – 4 часа)*

**Цель работы:** Получение практических навыков решения задач с использованием графического видеорежима.

***Методика выполнения работы***

Множество графических процедур и функций среды программирования Pascal собраны в модуле Graph. Для подключения библиотеки графических функций и процедур необходимо подключить модуль к вашей программе строкой

Uses graph;

Взаимодействие программы и видеосистемы в графических режимах обеспечивают драйверы.

Для инициализации графического режима используется процедура:

**InitGraph** (var Driver, Mode: integer; Path:string);

Где Driver – переменная типа integer, определяющая тип графического драйвера; Mode – переменная того же типа, задающая режим работы графического адаптера; Path – выражение типа string, содержащее путь доступа к файлу драйвера.

Пример фрагмента программы, где инициализируется графический режим:

Program primer;

Uses graph;

Var

D,m: integer; {переменные для установки драйвера и режима работы}

Begin

D:=9;

M:=2;

InitGraph(d,m, ‘здесь нужно указать путь к драйверу EGAVGA.BGI’}

……..

Наиболее простой способ выбора графического драйвера и режима – автоматический (detect).

Program primer;

Uses graph;

Var

D,m: integer; {переменные для установки драйвера и режима работы}

Begin

D:=detect;

InitGraph(d,m, ‘здесь нужно указать путь к драйверу EGAVGA.BGI’);

(Для работы в среде PascalABC требуется подключить модуль GraphABC. Используемые в нем функции могут отличаться от Turbo Pascal.)

**Завершение работы графического режима.** Завершает работу адаптера в графическом режиме и восстанавливает текстовый режим работы экрана процедура **CloseGraph**.

Любая программа, использующая графический режим, будет иметь одну и ту же структуру:

1. определение графического драйвера;
2. установка графического режима;
3. инициализация графического режима;
4. построения;
5. закрытие графического режима.

Цвет выводимых в графическом режиме на экран линий и символов можно задать процедурой

**SetColor** (color: word);

аргумент которой – целое число от 0 до 15 или имя одной из приведенных выше констант.

Установка цвета действует на те линии и тексты, которые выводятся после ее вызова, но не меняет цвет линий и символов, выведенных на экран ранее. Таким образом, процедуру SetColor следует вызывать каждый раз перед выбором нового цвета. Если цвет не установлен, то используется белый цвет.

**Установка цвета фона**

Чтобы установить цвет фона для всего экрана, используется процедура:

**SetBkColor** (color: word);

Если процедура установки цвета фона не вызвана, экран будет черным.

**Установка указателя вывода**

*Процедура* **MoveTo** (x,y: integer) перемещает указатель в точку с координатами x,y.

*Процедура* **MoveRel** (dx,dy: integer) перемещает указатель на dx, dy пикселей относительно последнего положения.

Функции **GetX** и **GetY** возвращают координаты x, y указателя вывода.

**Установка точки**

*Процедура* **PutPixel** (x,y: integer; color: word) устанавливает точку с координатами (x,y) и закрашивает ее указанным цветом color.

*Функция* **GetPixel** (x,y: integer): word возвращает значение цвета, в который окрашена точка с координатами (x,y).

**Рисование линий**

*Процедура* **Line** (x1,y1,x2,y2: integer) вычерчивает линию между двумя точками экрана с координатами (x1, y1) и (x2, y2).

*Процедура* **LineTo** (x,y: integer) вычерчивает линию от последнего положения указателя до точки с координатами (x, y).

**Построение графиков в декартовой системе координат**

Построение расчетов желательно сопровождать визуализацией результата на экране монитора. Это позволяет избежать ошибок, контролировать результаты расчета и оптимизировать алгоритм. Графики функций строятся обычно в декартовой системе координат (ХоY), Функция может быть задана в явном виде типа Y=F(X), в неявном: F(X, Y)=0, в параметрическом: X=Fx( t), Y=Fy( t).

*Построение графика функции Y=F(X) с масштабированием по осям координат.*

При построении графиков функций на экране монитора необходимо преобразовывать расчетные координаты в графические с соблюдением определенных пропорций, а также предусмотреть возможность масштабирования графика по осям координат. Для этого необходимо создать процедуры, обеспечивающие универсальность программирования графических изображений. Ниже приводится алгоритм построения графиков в правой системе координат, расположенной в заданной области экрана, с возможностью автоматического масштабирования.

Пусть задана непрерывная функция F(X) в диапазоне изменения аргумента Y=[A,В].Требуется построить по N точкам график функции Y=F(X) в прямоугольной области экрана.

Алгоритм построения графика функции Y=F(X).

1) Определяем массивы значений аргумента и функции:   
X[i], Y[i]=F( X[i] ), где i= 1...N.

При равномерном разбиении интервала [А..В] массивы можно задавать операторами:

Dx:= (В-А) / (N-1); ( шаг разбиения по Х )

for i:= l to N do begin

X[i]:= А + round( Dx\*(i-1));

Y[i]:= F( X[i] ) end;

2) Определяем наибольшее (Y max) и наименьшее (Y min) значения функции в   
заданном интервале изменения аргумента.

Y max:= Y[1];

Y min:= Y[1];

for i:=1 to N do begin

IF Y\_max < Y[i] THEN Y\_max;= Y[i];

IF Y\_min > Y[i] THEN Y\_min:= Y[i] end;

В случае явного задания функции для аргумента Х наибольшее значение Х max:= В и наименьшее X min:= А. Эти значения необходимо определить для полного размещения графика в расчетной области.

3). Определяем коэффициенты масштабирования при построении графика в   
заданной области экрана:

Kx:=(right- left) /(Х\_mах-Х\_min);

Ку:=(down- up) /(Y\_max- Y\_min);

Если Х и Y имеют одинаковую размерность или оба безразмерны, то появится искажение естественной формы **кривой** вследствие разного масштабирования по осям координат (растяжение или сжатие по одной из осей). Для вывода графика без искажения формы кривой следует переназначить координаты области экрана так, чтобы получить Ку=Кх.

*Пример операторов для автомасштабирования:*

Пусть заданы нижняя, верхняя и левая границы области построения графика: down, up, left. Необходимо найти значение right при условиях: Ку=Кх и right<=GetMaxX. Если условие ограничения графика по длине экрана не выполняется, то полагается right:=GetMaxX, и значение uр корректируется (уменьшается).

Ку:=(down-up)/(Y\_max- Y\_min);

Кх:= Ку;

right:= left+ Round(Кх\*(X\_max — Х\_min));

If right> GetMaxX then begin

right:= GetMaxX;

Кх:= (right-left) /(X\_max – Х\_min);

Ку:=кх;

up:= down - Round( Ку\*(Y\_max — Y\_min));

end;

4). Определяем координаты точек для построения графика в системе координат экрана;

XG[i]:= left +Round(Кх • (Х[i] –X\_min));

YG[i]:= down - Round(Ку • ( Y[i] –Y\_min));

Здесь Кх (X[i] –X\_min) и Ку (У[i] –Y\_min) — смещения координат точек по соответствующим осям относительно границ области left и down. Разные знаки перед смешениями по осям Х и Y получаются из-за необходимости "переворота" оси У, которая в координатах монитора направлена сверху вниз.

5). Строим график в виде последовательных отрезков:

moveto( ХО[1], YG[1] );   
for i := 2 to N 4 dо lineto(XG[i], YG[i]);

6). Строим оси координат, предварительно задав начало координат X0, Y0 (обычно Х0 =0, Y0=0 ) и вычислив:

XG0:left +round(Кх\* (X0-Х\_min));

YG0:= down - round( Ку **•** (Y0 - Y\_min));

Оси координат целесообразно строить лишь с случае их попадания в   
соответствующие диапазоны Х\_min...X\_max, Y\_min.. Y\_max.

if Х\_min\*Х\_mах<=0 then line (XG0, up, XG0,down);

if Y\_min \*Y\_max<=0 then line (left, YG0, right, YG0);

Доработать методику для совместного построения нескольких графиков несложно. Необходимо для всех функций определить дискретные координаты Y[i] и найти наибольшее и наименьшее значения для всех функций в заданных диапазонах изменения аргументов.

Наибольшим значениям присвоить Y\_max и X\_max, а наименьшим – Y\_min и Х\_min. Затем определить область для построения графика и координаты точек кривых. Кривые для разных функций желательно рисовать разным цветом с указанием вида функции.

Варианты задания.

Построить графики функций, заданных по одному из вариантов в таблице 1:

1. Y1 (х) и Y2( x)в разных областях экрана;
2. a) Y1 (х) и Y2( x)в родной области экрана;

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Функция Y1(x) | Диапазон х | Функция Y2(x) |
|  | 2Sin(x-5) | -5…5 | Sin(x2) |
|  | Cos(x2-4x-1) | -1…10 | Sin(x2 -4x-1) |
|  | (x2-1)/(x4+1) | -1.5 …1.5 | -x5+2x3-1 |
|  | X(x-3)(x+1) | -3…3 | x(x-3)(x-1) |
|  | ex+e-x | -3…3 | ex+e-2x |
|  | x2Sin(1/x) | 0.1 …3 | x3Sin(1/x2) |
|  | xSin2 (1/x) | 0.1 …5 | (x+1)Sin2 (1/x) |
|  | Sqrt(x2+2)Sin(x) | -10 …10 | (x+1)Sin(x) |
|  | Cos (x-1)+Abs(x) | -5 …5 | Sin(x)-Abs (x+1) |
|  | (x+3)/(x-2) | 1 … 20 | 3Sin(x) |
|  | x3/(x+1)2 | -10…5 | Cos(x2) |
|  | xCos(x-2) | -4… 4 | (x-5)/x2 |
|  | Sin(x2-2) | -2…10 | (x-1)Sin2(3/x) |
|  | ex-3x | -5…5 | Sqrt(x2+4)-3 |
|  | ex Abs(x+1) | -10… 10 | 2x5+x3-4 |

**Тема 21. Динамические структуры данных**

*(ориентировочное время выполнения задания – 4 часа)*

**Цель работы:** Получение практических навыков решения задач с использованием динамических структур данных – стека, очереди, линейных списков.

***Методика выполнения работы***

Перед выполнением задания изучить особенности каждого способа описания и работы со структурами данных.

Для каждой задачи необходимо выполнить следующие действия:

1. Написать программу, реализующую составленный алгоритм на языке программирования Си.

Программа должна включать:

* Описание соответствующего типа данных, допустимого для решения задачи
* ввод данных
* обработку данных в соответствии с заданным вариантом
* вывод результата на экран

2. Подготовить данные для тестирования работы

3. Проверить работоспособность программы на выбранных тестах.

4. Оформить отчет по практической работе в MS WORD.

Для выполнения работы необходимо знание следующих теоретических вопросов из курса дисциплины "Основы программирования":

* объявление динамических структур данных разных типов и их основные характеристики (необходимый объем ОП, допустимые операции);
* правила работы с указателями, допустимые операции;

**Стеком** называется упорядоченный набор элементов, в котором размещение

новых элементов и удаление существующих производится только с одного его

конца, называемого вершиной стека. При представлении стека в статической

области памяти для него выделяется память, как для вектора.

В дескрипторе этого вектора кроме обычных для вектора параметров должен находиться также указатель стека – адрес вершины стека. Указатель стека может указывать либо на первый свободный элемент стека, либо на последний записанный в стек элемент.

При работе со стеками разрешается пользоваться следующими операциями:

добавление элемента в стек, удаление элемента из стека, определение

пустоты и переполнения стека.

Стек должен быть описан следующим образом:

struct STACK

{

int data [ST\_SIZE];

int top;

};

Элементы стека хранятся в массиве размером ST\_SIZE. Переменная top

является указателем на вершину стека.

Для реализации операции Push (добавление) необходимо выполнить следующие

операции:

1) получить память под новый элемент (указательpt\_new);

2) занести значение элемента в поле данных;

3) привязать новый указатель next к старому top;

4) привязать указатель стека top к указателю pt\_new.

Функция, реализующая эту операцию, может выглядеть так:

LIST Push( LIST top, int new\_el )

{

LIST pt\_new;

pt\_new = (LIST) calloc (1, sizeof(struct LINK));

pt\_new->data = new\_el;

pt\_new->next = top;

top = pt\_new;

return top;

}

Для получения элемента из стека мы должны реализовать следующие

действия:

1) прочитать значение элемента;

2) переместить указатель стека top на следующий элемент;

3) освободить память.

LIST Pop(LIST top, int \*top\_el)

{

LIST pt\_buf;

pt\_buf=top;

\*top\_el = pt\_buf ->data;

free (pt\_buf);

return top;

}

**Очередью** называется упорядоченный набор элементов, которые могут удаляться

с одного ее конца (называемого началом очереди) и помещаться в другой конец

этого набора (называемого концом очереди).

Очередь мы будем описывать так:

struct QUEUE

{

int data [QUE\_SIZE];

int head;

int tail;

};

Как и ранее, мы предполагаем, что данные хранятся в массиве, размер

которого равен значению переменной QUE\_SIZE. Точно также переменная head

указывает на голову очереди, а переменная tail – на первую свободную ячейку в

ее хвосте.

Как мы уже знаем, над очередью выполняются следующие операции:

• создать очередь;

• добавить элемент в очередь;

• изъять элемент из очереди;

• очередь пуста(да/нет);

• очередь полная(да/нет);

• определение текущей длины очереди.

/\* Постановка элемента в очередь\*/

LIST AddQueEL( LIST que, int new\_el )

{

LIST pt\_new;

pt\_new = (LIST) calloc (1, sizeof(struct LINK));

pt\_new->data = new\_el;

pt\_new->next = que;

que = pt\_new;

return que;

}

/\* Получение элемента из очереди\*/

int GetQueEl( LIST que, int \*head\_el )

{

LIST pt\_buf, pt\_pred;

/\* Сохранение адреса конца очереди. \*/

pt\_buf = que;

if( pt\_buf != NULL ) {

while(pt\_buf->next != NULL) {

/\* Цикл для продвижения к началу очереди так, что pt\_buf получает адрес

начала очереди. \*/

pt\_pred = pt\_buf;

pt\_buf = pt\_buf ->next;

}

\*head\_el = pt\_buf ->data;

/\* Получение значения элемента, стоящего в начале очереди. Это значение

потом будет напечатано. \*/

if (que == pt\_buf) {

/\* Если адреса начала и конца очереди совпадут, значит очередь пуста. \*/

que = NULL;

return FALSE;

}

/\* Последний элемент очереди удаляется из нее и память, отведенная под

этот элемент освобождается. \*/

elsept\_pred -> next = NULL;

free (pt\_buf);

returnTRUE;

}

}

**Связанный список** – это структура данных, в произвольно выбранное место

которой данные могут как включаться, так и изыматься. Для обеспечения такой

гибкости, в каждый элемент добавляется указатель на следующий элемент списка.

Структуру списка можно задать так:

struct LINK

{

int data;

struct LINK \*next;

};

В этом описании значения элементов списка хранятся в поле data, а поле next

является указателем на следующий элемент списка.

Можно создать циклический список, который не имеет первого и последнего

элементов.

Элементы двунаправленного связанного списка содержат три поля: поле data,

содержащее информацию, хранимую в элементе, и левое и правое поля,

содержащие указатели на соседние элементы. Такой список можно реализовать

следующим образом:

struct LINK

{

int data;

struct LINK \*right;

struct LINK \*left;

};

Здесь right и left – правый и левый указатели на соседние элементы

соответственно.

В начало каждого списка можно поместить фиктивный узел, называемый

ведущим (заголовком) списка. Поле элемента заглавного узла игнорируется, но

ссылка узла сохраняется в качестве указателя узла, содержащего первый элемент

списка.

*Вариант 1*

Разработайте следующие функции для обслуживания стека:

• добавление элемента в стек;

• печать стека;

• создание стека;

• извлечение(удаление) элемента из стека.

*Вариант 2*

Разработайте комплекс функций для обслуживания стека:

• создание стека;

• добавление элемента в стек;

• печать стека;

• упорядочивание элементов в стеке по полю данных.

*Вариант 3*

Напишите программу, обрабатывающую очередь пассажиров, желающих купить авиабилет. Элементы очереди содержат следующую информацию: фамилия пассажира, пункт прибытия, дата вылета. Информация об имеющихся билетах хранится в списке, элементы которого содержат номер рейса, пункт прибытия, дату, число непроданных билетов. Пассажир, обеспеченный билетом, удаляется из очереди, остальные остаются там. Содержимое списка по мере обеспечения пассажиров билетами корректируется.

*Вариант 4*

Напишите программу, формирующую очередь целых чисел и затем удаляющую из нее все числа, меньшие заданного числа (использовать можно только функции добавления и изъятия элементов).

*Вариант 5*

Разработайте комплекс функций для обслуживания однонаправленного списка, содержащего элементы с ключами. Поле ключа заполняется уникальными значениями. Правило формирования ключа выберите самостоятельно. Это может быть номер элемента в списке или номер элемента при его создании.

Необходимо разработать следующие функции:

• создание списка;

• добавление элемента после элемента с заданным ключом;

• вывод содержимого поля ключа и поля данных.

*Вариант 6*

Разработайте функции для обслуживания однонаправленного списка, содержащего элементы без ключей:

• создание списка;

• добавление элемента после элемента с заданным номером;

• вывод содержимого полей данных.

*Вариант 7*

Разработайте функции для обслуживания однонаправленного списка, содержащего элементы без ключей:

• создание списка;

• вывод содержимого полей данных;

• удаление элемента с заданным значением.

*Вариант 8*

Дан список. Создайте функцию, которая перемещает наибольший элемент данного списка так, чтобы он стал последним узлом.

*Вариант 9*

Напишите программу, которая удаляет узлы в четных позициях связного списка.

*Вариант 10*

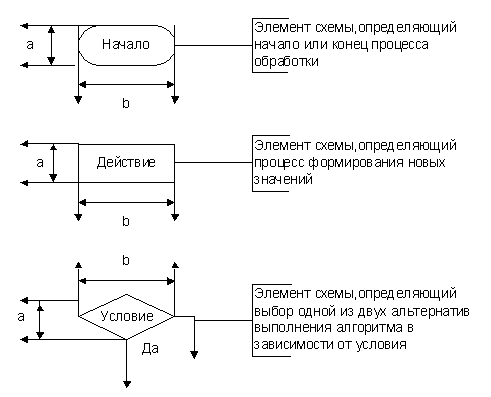
В кассах аэропорта 4 окна. В каждом окне на обслуживание пассажира уходит некоторое (различное для каждого пассажира) время. Вновь входящий пассажир становится в самую короткую очередь. При двух и более одинаковых по длине очередях ( в том числе и пустых) он становится в ближайшую (то есть имеющую наименьший номер) очередь. Для каждого пассажира известно время его прихода в кассы и время на его обслуживание.

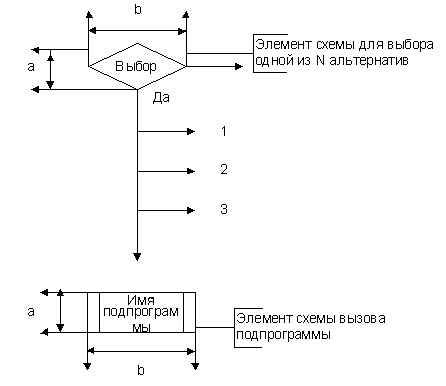
Напишите программу, моделирующую деятельность касс. Определите при помощи этой модели среднее время на обслуживание одного пассажира при заданном входном потоке пассажиров.

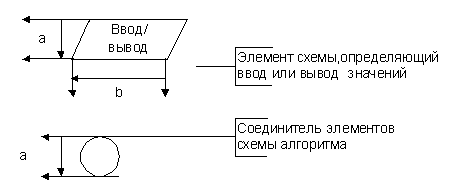
**ЧАСТЬ ВТОРАЯ**

**МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ**

**2.1 Оформление блок-схем**

  Оформление программ должно соответствовать определенным требованиям. В настоящее время действует единая система программной документации (ЕСПД), которая устанавливает правила разработки, оформления программ и программной документации. В ЕСПД определены и правила оформления блок-схем алгоритмов (ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД).  
*Основные блоки схем алгоритмов (элементы схем определены ГОСТ 19.701-90)*  
 





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 Размер ‘a’ (в миллиметрах) должен выбираться, исходя из формулы: a = 10 + 5\*n, где n = 1, 2, 3, …  
           Размер ‘b’ определяется соотношениями: b = 1.5\*a или b = 2\*a.  
           В учебном процессе рекомендуется использование блок-схем, в соотношении a = 20мм, b = 40мм.  
           Операции обработки данных и носители информации изображаются на схеме соответствующими блоками. Большая часть блоков по построению условно вписана в прямоугольник со сторонами а и b.

Минимальное значение а равно 10 мм, увеличение а производится на число, кратное 5 мм. Размер b=1,5 мм. Для отдельных блоков допускается соотношение между а и b, равное 1:2. В пределах одной схемы рекомендуется изображать блоки одинаковых размеров. Все блоки нумеруются.

           Линии, соединяющие блоки и указывающие последовательность связей между ними, должны проводиться параллельно линиям рамки. Стрелка в конце линии может не ставиться, блок может иметь в качестве продолжения одни 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.  
           Если при обрыве линии продолжение схемы находится на этом же листе, то на одном и другом конце линии изображается специальный символ ***соединитель*** — окружность диаметром 0,5 мм. Внутри парных окружностей указывается один и тот же идентификатор. В качестве идентификатора, как правило, используется порядковый номер блока, к которому направлена соединительная линия. Если схема занимает более одного листа, то в случае разрыва линии вместо окружности используется *межстраничный соединитель*. Внутри каждого соединителя указывается адрес — откуда и куда направлена соединительная линия. Адрес записывается в две строки: в первой указывается номер листа, во второй — порядковый номер блока.

**2.2 Порядок оформления практической работы по программированию**

Каждая тема из первой части учебного пособия должна быть оформлена в виде отчета по практической работе, который содержит:

1. Титульный лист (по образцу)
2. Текст задания
3. Блок-схема алгоритма (на бланке)
4. Текст программы на языке программирования
5. Таблица тестовых данных
6. Скриншот экрана выполнения программы для заданных входных данных

Оформленную работу необходимо сохранить в файле с именем:

***Группа\_ Фамилия\_№ работы.docx***

ПРИМЕР ОФОРМЛЕНИЯ ТИТУЛЬНОГО ЛИСТА

Федеральное государственное образовательное бюджетное учреждение

высшего образования «Финансовый университет при Правительстве Российской Федерации»

**Колледж информатики и программирования**

**Специальность 09.02.03 Программирование в компьютерных системах**

**ПМ.01 Разработка программных модулей программного обеспечения для компьютерных систем**

**УП.01 Учебная практика по программированию**

ПРАКТИЧЕСКАЯ РАБОТА №

Вариант №

Выполнил:

студент гр. \_\_\_\_\_\_\_\_\_\_\_\_

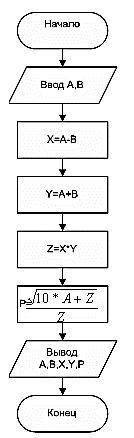
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ф.И.О.

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ф.И.О.

Москва, 2016

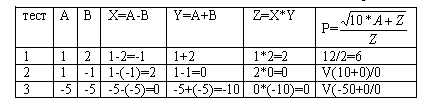
ПРИМЕРЫ ОФОРМЛЕНИЯ ЗАДАЧ

***Пример 1.*** Определить, что будет выведено на печать в результате выполнения следующего алгоритма. Провести тестирование алгоритма для значений **А** и **В**, приведённых в таблице. 

1.  Начало;   
2.  Ввод (А,В);   
3.  X:= А - В;   
4.  Y:= A + B;   
5.  Z:= Y\*X;   
6.  P:= ((10\*A + Z)½)/Z;   
7.  Вывод(А, В, Х, Y, Z, P);   
8.  Конец.

 Таблица значений, заполненная в ходе тестирования алгоритма

задачи 1, будет иметь вид:



Программы на языках программирования Паскаль и С:

**Program** P1;

**Var** a, b: integer; {исходные данные}

x, y, z: integer;p: real; {вычисляемые переменные}

**begin**

writeln('Введите значение А');

readln(a);

writeln('Введите значение B');

readln(b);

x:=a-b;

y:=a+b;

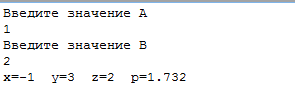
z:=a\*b;

p:=sqrt (10\*a + z)/z;

writeln ('x=', x,' y=', y,' z=', z,' p=',p:5:3);{вывод результатов}

**end**.

*Вид экрана:*

**

#include "stdafx.h"

#include "stdio.h"

#include "math.h"

#include "conio.h"

int main()

{

int a,b; //исходные данные

int x, y, z; // вычисляемые переменные

double p;

puts ("Vvedite A");

scanf("%i",&a);

puts ("Vvedite B");

scanf("%i",&b);

x=a-b;

y=a+b;

z=a\*b;

p=sqrtf(10\*a+z)/z;

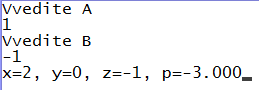
printf("x=%d, y=%d, z=%d, p=%5.3lf",x,y,z,p); // вывод результата

getch();

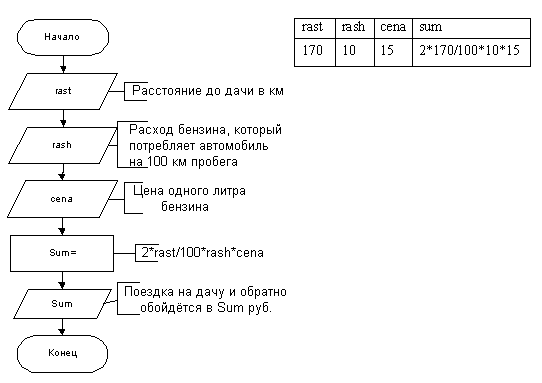
return 0;

}

*Вид экрана*:

**

***Пример 2.*** Нарисовать блок схему вычисления стоимости поездки на автомобиле на дачу (туда и обратно). Исходными данными являются: расстояние до дачи в км; кол-во бензина; которое потребляет автомобиль на 100 км пробега; цена одного литра бензина.



**Program P2;**

**Var** rast, rash, cena: integer;  
sum:real;

**Begin**

Writeln(‘ введите значение rast’);

Readln(rast);

Writeln(‘ введите значение rash’);

Readln( rash);

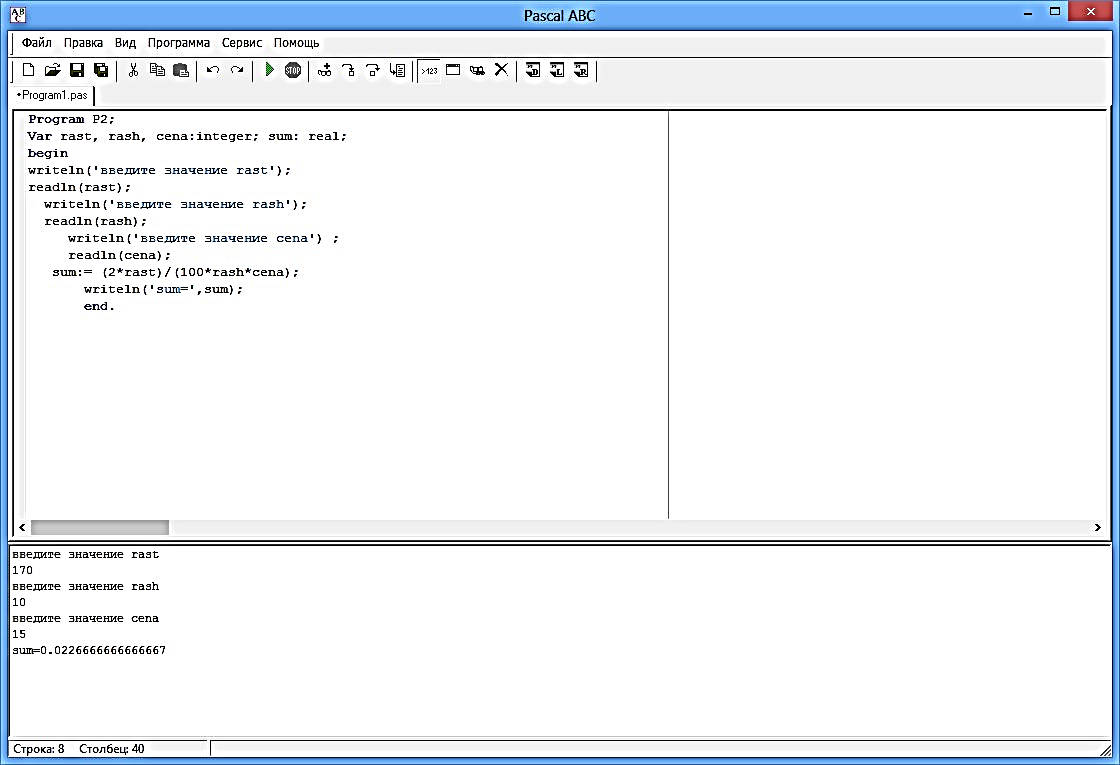
Writeln(‘ введите значение cena’);

Readln(cena);

Sum:=(2\*rast)/(100\*rash\*cena);

Writeln(‘Сумма=’,sum);

End.

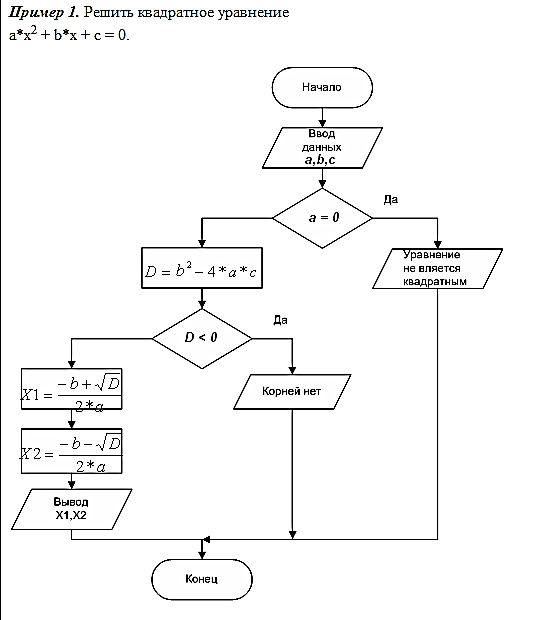


**2.3 Примеры решения задач**

**Тема 1-2. Программирование базовых алгоритмических структур: следование, ветвление**

ПРИМЕР 1 Найти корни квадратного уравнения

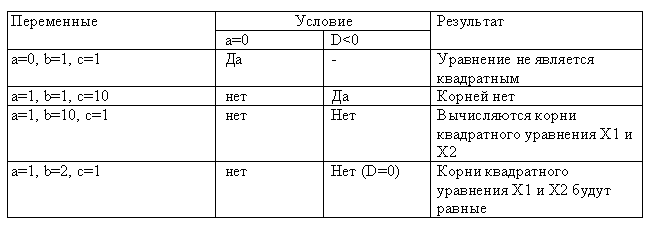
*Ax2+bx+c=0*



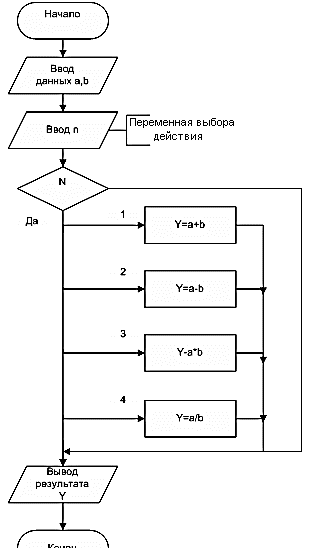
ПРИМЕР 1

Выполнение алгоритма осуществляется следующим образом. После ввода значений коэффициентов **a, b, c** производится проверка значения **a**.

При **а = 0** уравнение становится линейным - об этом сообщается пользователю, и выполнение алгоритма прекращается. В противном случае, когда **а** отлично от нуля производится вычисление дискриминанта **D**. Если *D < 0*, то выдается сообщение о том, что уравнение не имеет корней, в противном случае  
(*D > 0*) производится вычисление корней **Х1** и **Х2**.  
           Проверим работу алгоритма для нескольких наборов **a, b, c** (наборы значений подбираются таким образом, чтобы можно было проверить все ветви алгоритма).



ПРИМЕР 2***.*** Составить алгоритм работы примитивного калькулятора, который выполняет действия «+», «-», «\*», «/» двух переменных.

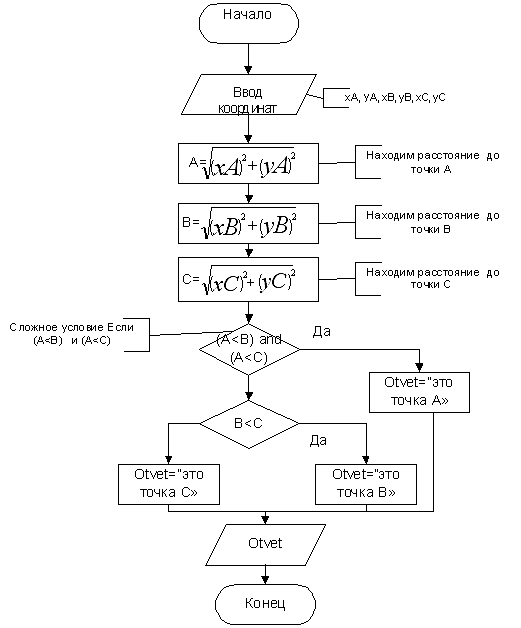


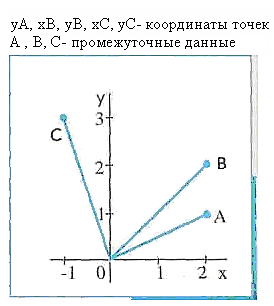
Данная блок-схема реализовывает алгоритм ветвления - выбор, в котором значение переменной **N** осуществляется выбор действия.

N = 1 - сложение N = 3 - умножение;

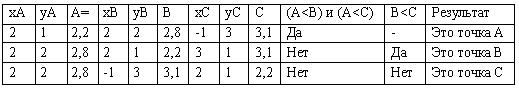
N = 2 - вычитание N = 4 - деление.

ПРИМЕР ***3.*** Определить ближайшую к началу координат точку плоскости из трех заданных.





Для проверки работы алгоритма будут использованы три набора тестовых данных:



ПРИМЕР 4 . Определить, пройдёт ли кирпич с рёбрами a,b,c в прямоугольное отверстие со сторонами x и y.

program P2;

var x,y,a,b,c:real;

begin

writeln('Введите размер отверстия x');

readln(x);

writeln('Введите размер отверстия y');

readln(y);

writeln('Введите ребро a');

readln(a);

writeln('Введите ребро b');

readln(b);

writeln('Введите ребро c');

readln(c);

if (x>a) and (y>a) and (x>b) and (y>b) or

(x>a) and (y>a) and (x>c) and (y>c) or

(x>b) and (y>b) and (x>c) and (y>c) then

writeln('кирпич проходит в отверстие')

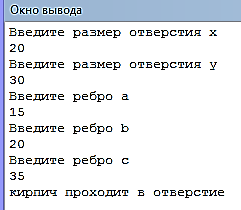
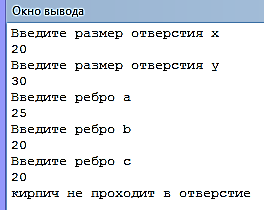
else

writeln('кирпич не проходит в отверстие');

readln;

end.

Результаты работы программы:

Примечание: Наиболее распространенная ошибка при решении данной задачи – сравнение площадей отверстия и грани кирпича.

*ПРИМЕР 5.* Составить блок-схему и программу, в которой с учетом роста человека определяется его оптимальный вес и рекомендации о необходимости поправиться или похудеть (оптимальный вес вычисляется по индексу массы тела).

#include "stdafx.h"

#include <stdio.h>

#include <conio.h>

#include "locale.h"

#include "math.h"

void main()

{

setlocale(LC\_ALL,"Russian");

int v,r; //вес, рост

int d; //переменная выбора

float ind; //индекс массы тела

printf("Введите свой \nвес: ");

scanf("%d",&v);

printf("рост: ");

scanf("%d",&r);

ind=(float)v/pow((float)r/100,2);

printf(" индекс массы тела:%5.2f",ind);

if(ind>25) d=1;

else if(ind<18) d=2;

else d=3;

switch(d)

{

case 1:printf("\nСбросьте вес"); break;

case 2:printf("\nНаберите вес"); break;

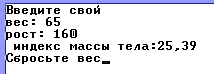
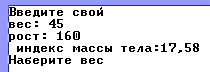
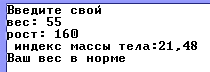
case 3:printf("\nВаш вес в норме"); break;

}

getch();

}

Результат работы программы:

**Тема 3. Программирование базовых алгоритмических структур: цикл**

*ПРИМЕР 1* Составьте блок-схему и программу для нахождения суммы и произведения всех натуральных чисел на отрезке [30,90]

program chisla;

var n,i:integer;

sum:longint; proizv:real;

begin

sum:=0; proizv:=1;

for i:=30 to 90 do begin

sum:=sum+i;

proizv:=proizv\*i;

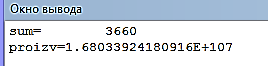
end;

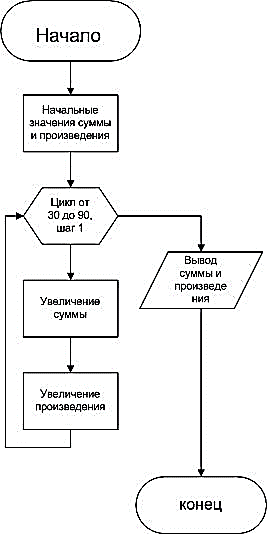
writeln('sum=',sum:12); writeln('proizv=',proizv);

readln;

end.

Результаты работы программы:



блок-схема. 

*ПРИМЕР 2*. Вычислить по формуле сумму первых 10 членов ряда:

Для решения задачи выведем рекуррентную формулу:

#include "stdafx.h"

#include <stdio.h>

#include <conio.h>

#include "locale.h"

void main()

{

setlocale(LC\_ALL,"Russian");

int x,n=10; //исходные данные

int i; //переменная цикла

float a, s; //член ряда, сумма ряда

printf("Введите x: ");

scanf("%d",&x);

a=(float)x\*x/6; //первый член ряда

s=a; //начальное значение суммы

for(i=2;i<11;i++)

{a=a\*x\*(i-1)/(i+2);

printf("a(%d)=%f\n",i,a); //отладочная печать для контроля слагаемых

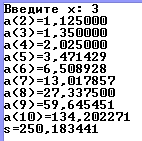
s+=a; //наращивание суммы

}

printf("s=%f\n",s);

getch();

}

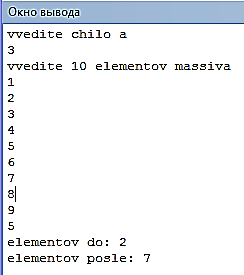
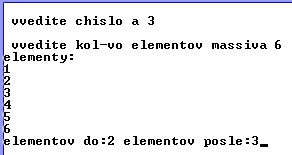


**Тема 4-5. Обработка одномерных массивов в Паскале, С**

ПРИМЕР 1 Дан одномерный массив из n чисел. Если число А встречается в массиве, найти количество элементов до первого вхождения числа A.

|  |  |
| --- | --- |
| **program** p41;  **const** n=10;  **var** b:**array**[1..n] **of** integer;  i,a,d,p:integer;  **begin**  writeln('vvedite chilo a');  readln(a);  writeln('vvedite ',n,' elementov massiva');  **for** i:=1 **to** n **do** read(b[i]);  **for** i:=1 **to** n **do**  **if** (b[i]=a) **then begin**  d:=i-1;  p:=n-i;**break**;  **end**;  writeln('elementov do: ',d);  writeln('elementov posle: ',p);  readln;  **end**. | #include <stdio.h>  #include <conio.h>  void main(void)  {  int b[10];  int temp,i,a,d,p,n;  printf("\n vvedite chislo a ");  scanf("%d",&a);  printf("\n vvedite kol-vo elementov massiva ");  scanf("%d",&n);  printf("elementy:\n");  for (i=0;i<n;i++)  {  scanf("%d ",&temp);  b[i]=temp;  }  for (i=0;i<n;i++)  if (b[i]==a)  {d=i;p=n-i;break;}  printf("elementov do:%d elementov posle:%d",d, p);  getch();  } |

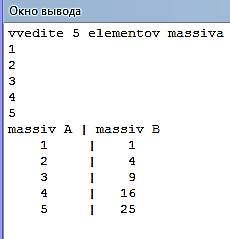
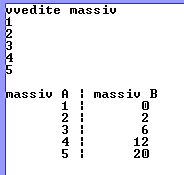
Результаты:

*ПРИМЕР 2*. Сформируйте массив, содержащий n элементов, с помощью чисел введенных с клавиатуры. Получите новый массив, каждый элемент которого в i раз больше соответствующего элемента первого массива. Выведите на экран исходные и полученные данные.

|  |  |
| --- | --- |
| **program** p42;  **const** n=5;  **var** a,b:**array**[1..n] **of** integer;  i:integer;  **begin**  writeln('vvedite ',n,' elementov massiva');  **for** i:=1 **to** n **do** read(a[i]);  writeln('massiv A | massiv B');  **for** i:=1 **to** n **do begin**  b[i]:=a[i]\*i;  writeln(a[i]:5,' |',b[i]:5);  **end**;  readln;  **end**. | #include "stdafx.h"  #include <stdio.h>  #include <conio.h>  int main()  {  const int n=5;  int a[n],b[n];  int i;  printf("vvedite massiv\n");  for (i=0;i<n;i++)  {  scanf("%d ",&a[i]);  }  printf("massiv A | massiv B\n");  for (i=0;i<n;i++)  {  b[i]=a[i]\*i;  printf("%7d | %7d\n ",a[i],b[i]);  }  getch();  return 0;  } |

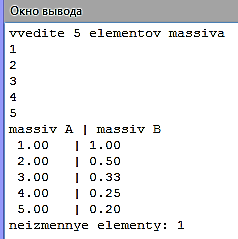
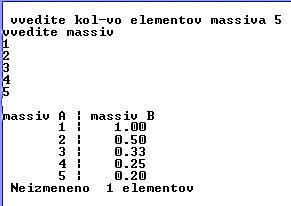
Результаты:

*ПРИМЕР 3*. Сформируйте массив, содержащий n элементов. Получите новый массив, каждый элемент которого равен обратному соответствующему элементу исходного массива. Подсчитайте количество неизменившихся элементов. Выведите на экран исходные и полученные результаты.

|  |  |
| --- | --- |
| **program** p43;  **const** n=5;  **var** a,b:**array**[1..n] **of** real;  i,c:integer;  **begin**  c:=0;  writeln('vvedite ',n,' elementov massiva');  **for** i:=1 **to** n **do** read(a[i]);  writeln('massiv A | massiv B');  **for** i:=1 **to** n **do begin**  b[i]:=1/a[i];  writeln(a[i]:5:2,' |',b[i]:5:2);  **if** (a[i]=b[i]) **then** c:=c+1;  **end**;  writeln('neizmennye elementy: ',c);  readln;  **end**. | #include "stdafx.h"  #include <stdio.h>  #include <conio.h>  void main(void)  {  int a[10];  float b[10];  int i,n,c;  c=0;  printf("\n vvedite kol-vo elementov massiva");  scanf("%d",&n);  printf("vvedite massiv\n");  for (i=0;i<n;i++)  {  scanf("%d ",&a[i]);  }  printf("massiv A | massiv B\n");  for (i=0;i<n;i++)  {  b[i]=1/(float)a[i];  printf("%7d | %7.2f\n ",a[i],b[i]);  if (a[i]==b[i]) c++;  }  printf("Neizmeneno %d elementov",c);  getch();  } |

Результаты:

**Тема 6-7. Двумерные массивы в Паскале, С**

Составьте программу формирования и вывода на экран одномерного массива, состоящего из максимальных элементов столбцов двумерного массива, заданного с помощью датчика случайных чисел.

|  |  |
| --- | --- |
| Pascal:  program task\_1;  const n=20;  var  a:array[1..n] of integer;  b:array[1..n,1..n] of integer;  i,k,r,j:integer;  begin  randomize;  writeln('Zadaite razmeri massiva (max 20x20)');  readln(k,r);  writeln(2-mernii massiv - ');  for i:=1 to k do begin  writeln ;  for j:=1 to r do begin  b[i,j]:=random(101);  write(b[i,j],' ')  end;  end;  writeln;  writeln('Polychennii massiv -');  for j:=1 to k do begin  for i:=1 to r do begin  if b[i,j]>a[j] then a[j]:=b[i,j] ;  if i=r then write(a[j],' ')  end;  end;  readln;  end. | C:  #include "stdafx.h"  #include <stdio.h>  #include <conio.h>  #include "stdlib.h"  void main(void)  {const int n=5;  int a[n][n];  int b[n];  int i,j;  printf("vvedite massiv\n");  for (i=0;i<n;i++)  {  for (j=0;j<n;j++)  {  a[i][j]=1+rand()%100;  printf("%3d", a[i][j]);  }  printf("\n");  }  for (j=0;j<n;j++)  {b[j]=a[0][j];  for (i=1;i<n;i++)  {  if (a[i][j]>b[j]) b[j]=a[i][j];  }  }  printf("\n\nmassiv B\n");  for (i=0;i<n;i++)  printf("%3d ",b[i]);  getch();  } |
| Результат: |  |

Задание 2

Сформировать матрицу по образцу.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 … | 1 | 1 |
| 1 | 0 | 0 … | 0 | 1 |
| … | … | … | … | … |
| 1 | 0 | 0 … | 0 | 1 |
| 1 | 1 | 1 … | 1 | 1 |

|  |  |
| --- | --- |
| Pascal: | С: |
| program task\_3;  const n=20;  var  a:array[1..n,1..n] of integer;  i,j,q,w:integer;  begin  writeln(‘Zadaite razmeri massiva (max 20x20)’);  readln(q,w);  writeln(‘ matrcia –‘);  for i:=1 to q do begin  writeln;  for j:=1 to w do begin  if (i=1) or (j=1) or (i=q) or (j=w) then a[i,j]:=1  else  a[i,j]:=0;  write(a[i,j],’ ‘)  end;  end;  readln;  end. | #include "stdafx.h"  #include <stdio.h>  #include <conio.h>  #include "stdlib.h"  void main(void)  {const int n=10, m=8;  int a[n][m];  int i,j;  printf("massiv\n");  for (i=0;i<n;i++)  {  for (j=0;j<m;j++)  {  if((i==0)||(i==n-1)||(j==0)||(j==m-1)) a[i][j]=1;  else a[i][j]=0;  printf("%3d", a[i][j]);  }  printf("\n");  }  getch();  } |
| Результат: |  |

**Темы 8-11. Алгоритмы сортировки: метод "Пузырька", сортировка выбором, сортировка вставкой, быстрая сортировки: метод Хоара**

*ПРИМЕР 1.* Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить столбцы по возрастанию последних элементов столбцов. Вывести массивы до и после сортировки.

Метод сортировки «пузырьком»

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a[10][10],i,j,k,p;

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

a[i][j] = rand()%25; printf("%3d",a[i][j]);

}

printf("\n");

}

printf("\n");

for(k=0;k<10;k++)

{

for(j=0;j<10-k-1;j++)

{

if(a[9][j]>a[9][j+1])

{

for(i=0;i<10;i++)

{

p = a[i][j];

a[i][j] = a[i][j+1];

a[i][j+1] = p;

}

}

}

}

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

printf("%3d",a[i][j]);

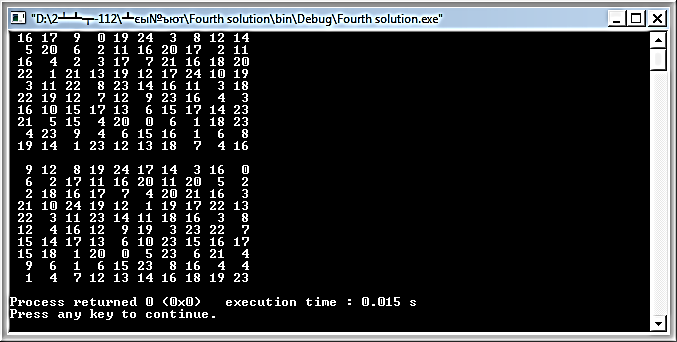
}

printf("\n");

}

return 0;

}



Метод сортировки выбором

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a[10][10],i,j,k,min,temp;

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

a[i][j] = rand()%25; printf("%3d",a[i][j]);

}

printf("\n");

}

printf("\n");

{

for(i=0;i<9;i++)

{

min = i;

for(j=i+1;j<10;j++)

{

if(a[9][j]<a[9][min]) min = j;

}

for(k=0;k<10;k++)

{

temp = a[k][i];

a[k][i] = a[k][min];

a[k][min] = temp;

}

}

}

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

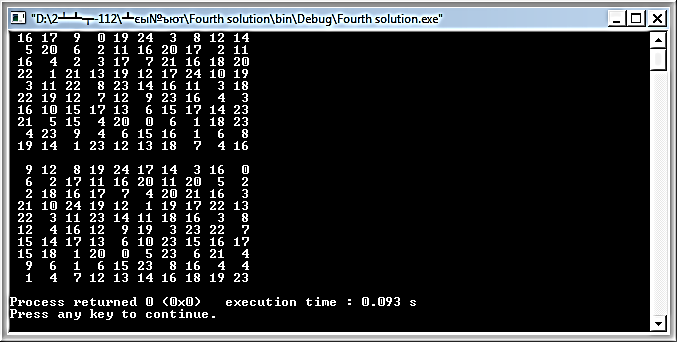
printf("%3d",a[i][j]);

}

printf("\n");

}

return 0;

}

Метод сортировки вставкой

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a[10][10],i,j,k,p,l;

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

a[i][j] = rand()%25; printf("%3d",a[i][j]);

}

printf("\n");

}

printf("\n");

for(i=1;i<10;i++)

{

p = a[9][i];

j = i - 1;

while(p<a[9][j]&&j >= 0)

{

a[9][j+1] = a[9][j];

for(k=0;k<9;k++)

{

l = a[k][j+1];

a[k][j+1] = a[k][j];

a[k][j] = l;

}

j = j - 1;

}

a[9][j+1] = p;

}

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

printf("%3d",a[i][j]);

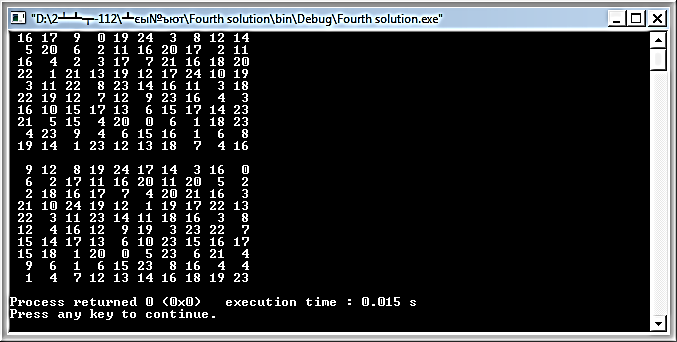
}

printf("\n");

}

return 0;

}



Метод сортировки Хоара

#include <stdio.h>

#include <stdlib.h>

void quicksort (int list[], int low, int high)

{

int pivot, i, j,temp;

if (low < high)

{

pivot = low;

i = low;

j = high;

while(i < j)

{

while((list[i] <= list[pivot]) && (i <= high))

{

i++;

}

while((list[j] > list[pivot]) && (j >= low))

{

j--;

}

if(i<j)

{

temp = list[i];

list[i] = list [j];

list[j] = temp;

}

}

temp = list[j];

list[j] = list[pivot];

list[pivot] = temp;

quicksort(list,low,j-1);

quicksort(list,j+1,high);

}

}

int main()

{

int a[10][10],i,j;

for(i=0;i<10;i++)

{

for(j=0;j<10;j++)

{

a[i][j] = rand()%25; printf("%5d",a[i][j]);

}

printf("\n");

}

printf("\n");

quicksort(a[9],0,9);

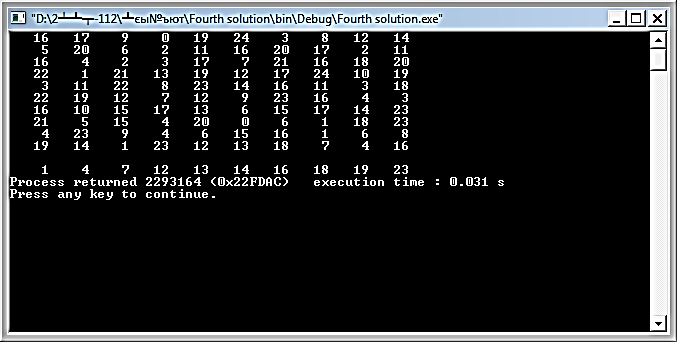
for(i=0;i<10;i++)

{

printf("%5d",a[9][i]);

}

}



Вывод: самые подходящие для выполнения данной задачи методы – метод пузырька и метод вставки в силу их быстроты (оба они показали наименьшее время равное 0,015 секунды) и удобства модернизации их алгоритмов под данную задачу. Метод выбора также удобно модифицировать, но он работает почти в два раза медленнее (0,031 секунды), а метод Хоара одновременно самый медленный (0,091 секунды) и самый сложномодифицируемый (потребуется несколько дополнительных циклов для выполнения задачи, в то время как при сортировке методом пузырька и методом вставки достаточно добавить перемещение требуемых элементов столбца во вложенный цикл алгоритма)

*ПРИМЕР 2*. Разработать программу, которая вводит целочисленную матрицу из *n* строк и *m* столбцов. Упорядочить все элементы матрицы таким образом, чтобы при чтении матрицы по строкам ее элементы образовывали отсортированный по убыванию массив. Вывести массивы до и после сортировки.

1. Сортировка методом «пузырька»

Алгоритм сортировки: При первом проходе вдоль массива, начиная проход "снизу"- с первого элемента, берется первый элемент и *поочередно* сравнивается с последующими. При этом:

* если встречается более "легкий" (с меньшим значением) элемент, то они меняются местами;
* при встрече с более "тяжелым" элементом, последний становится "*эталоном*" для сравнения, и все следующие сравниваются с ним.

В результате наибольший элемент оказывается в самом «верху» - в конце массива.

Если необходимо сортировать по убыванию, обмен происходит, если встречается более «тяжелый» элемент, тогда в конце массива окажется наименьший элемент.

#include "stdafx.h"

#include <stdio.h>

#include <conio.h>

#include "stdlib.h"

#include <time.h>

int p,i,j;

const int n=3,m=5;

void sort\_p(int b[m\*n]) //алгоритм сортировки

{

printf("\n\n");

for(i=0;i<n\*m-1;i++)

for(j=n\*m-1;j>i;j--)

if(b[j]>b[j-1])

{

p=b[j];

b[j]=b[j-1];

b[j-1]=p;

}

}

void main()

{

srand(time(NULL));

int b[n\*m],a[n][m],k=0;

for(i=0;i<n;i++) // заполнение массива

{

for(j=0;j<m;j++)

{

a[i][j]=rand()%99+1;

printf("%5d",a[i][j]);

}

printf("\n");

}

printf("\n"); //формирование одномерного массива из двумерного

for(i=0;i<n;i++)

for(j=0;j<m;j++){

b[k]=a[i][j];k++;

}

k=0;

sort\_p(b); //вызов функции сортировки

for(i=0;i<n;i++) //формирование двумерного массива из одномерного

for(j=0;j<m;j++)

{

a[i][j]=b[k];k++;

}

for(i=0;i<n;i++)

{

for(j=0;j<m;j++)

printf("%5d",a[i][j]);

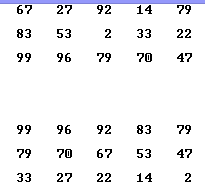
printf("\n");

}

getch();

}

Результат:



2. Сортировка выбором

Алгоритм:

**нц для i от 1 до N-1***выбрать среди***M[i+1]**,. . ., **M[N]***наименьший элемент и поменять его местами с***M[i]  
кц**

Функция сортировки:

void sort\_v(int b[m\*n])

{

for(i=0;i<n\*m-1;i++)

{

for(j=i+1;j<m\*n;j++)

if(b[j]>b[i])

{

p=b[j];

b[j]=b[i];

b[i]=p;

}

}

}

3. Сортировка простыми вставками

Алгоритм:

**нц для j от 2 до N***переместить***M[j]***на позицию***i <= j***такую, что***M[j] < M[k]***для***i<= k < j***и  
         либо***M[j] >= M[i-1]***, либо***i=1  
кц**

Функция сортировки:

void sort\_vs(int b[n\*m])

{

int x;

for(i=1;i<n\*m;i++)

{

x=b[i];

j=i-1;

while((b[j]<x)&&(j>-1))

{

b[j+1]=b[j];

j=j-1;

}

b[j+1]=x;

}

}

4. Сортировка Хоара (быстрая)

Алгоритм:

После выбора опорного элемента, слева от него остаются все элементы меньше его, а справа большие. И далее происходит рекурсивный вызов функции сортировки для каждого подмассива. Идея в том, чтобы сортировку обеих частей провести в отдельных поток.

Функция сортировки:

int sort\_x(int b[n\*m],int last)

{

int t;

for (int i=0; i<last; i++)

{

if (b[i] < b[last])

{

t=b[i];

b[i] = b[last];

b[last] = t;

}

}

if (last > 1)

{

last--;

sort\_x(b,last);

}

else return 0;

}

**Тема 12. Рекурсивные алгоритмы**

*ПРИМЕР 1*. Вычислить значение функции y=x-1/x! при помощи рекурсивной функции.

#include "stdafx.h"

#include <stdio.h>

#include <conio.h>

float fact(int x)

{

if((x==0)||(x==1))return 1;

else return x\*fact(x-1);

}

void main()

{

int x;

printf("Vvedite chislo: ");

scanf("%d",&x);

printf("Funkciya ravna: %.4f",x-1/fact(x));

getch();

}



**Тема 13. Разработка и программирование алгоритмов сложной структуры: обработка матриц с небольшим количеством строк или столбцов**

.

*ПРИМЕР 1*. При заданных элементах x1,x2,…,xn. Найти по отдельности суммы С1 и С2 элементов, значение которых, соответственно больше и меньше W.

**program** zadacha;

**const** n=10;

**type** s=**array**[1..n] **of** integer;

**var** i,v,k:integer;

c1,c2:integer;

a:s;

w:real;

**procedure** klav(m:integer; **var** a:s); //процедура ввода с клавиатуры

**var** i:integer;

**begin**

writeln('vvedite massiv');

**for** i:=1 **to** m **do begin**

write('a[',i,']= ');

read(a[i]);

**end**;

writeln;

**end**;

**procedure** rand(m:integer; **var** a:s); //процедура ввода случайных чисел

**var** i:integer;

**begin**

randomize;

**for** i:=1 **to** m **do**

a[i]:=random(10);

**end**;

**procedure** fil(m:integer;**var** a:s); //процедура ввода из файла

**var** i:integer; ft:text; t:integer;

**begin**

assign(ft,'c:\cc\massiv1.txt');

i:=1; reset(ft);

**while not** eof(ft) **do begin**

read(ft,t);

a[i]:=t;

i:=i+1; **end**; close(ft);

**end**;

**procedure** ecran(a:s); //процедура вывода на экран

**var** i,m:integer;

**begin**

m:=n;

**for** i:=1 **to** m **do begin**

write('a[',i,'] ');

**end**; writeln;

**for** i:=1 **to** m **do begin**

write(a[i]:3,' ');

**end**; writeln; readln;

**end**;

**procedure** vfile(a:s); //процедура вывода в файл

**var** ft:text; t:integer;

i,m:integer;

**begin** m:=n;

assign(ft,'c:\cc\massiv2.txt');

rewrite(ft);

**for** i:=1 **to** m **do begin**

t:=a[i];

write(ft,t);

**end**;

close(ft);

**end**;

**begin**

v:=0;

writeln('1:vvod s klavy');

writeln('2:sluchainye chisla');

writeln('3:iz faila');

read(v);

**case** v **of**

1: klav(n,a);

2: rand(n,a);

3: fil(n,a); **end**;

writeln('vibirite sposop vivoda');

writeln('1:na ekran');

writeln('2:v file');

read(k);

**case** k **of**

1:ecran(a);

2:vfile(a); **end**;

writeln('vvedite chislo w');

readln(w);

c1:=0;

c2:=0;

**for** i:=1 **to** n **do begin**

**if** (a[i]>w) **then** c1:=c1+a[i]

**else if (a[i]<>w) then** c2:=c2+a[i];

**end**;

writeln('summa bolshyx elementov:',c1);

writeln('summa menshyx elementov:',c2);

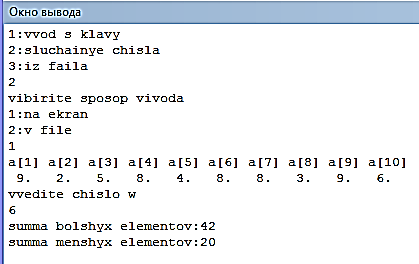
**end**.

Алгоритм выполнения.

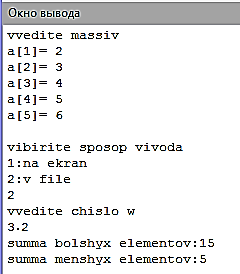
1. ввод числа W.
2. диапазон, при котором выполняется данное условие( от i=1 до n)
3. условие: (a[i]>w).
4. если удовлетворяет условию, то c1=c1+a[i], если нет, то c2=c2+a[i]
5. вывод на экран суммы больших элементов( с1 )
6. вывод на экран суммы меньших элементов ( с2 ).

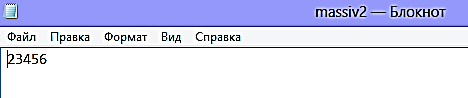
Результаты:

1. Матрица заполняется случайными числами, вывод на экран

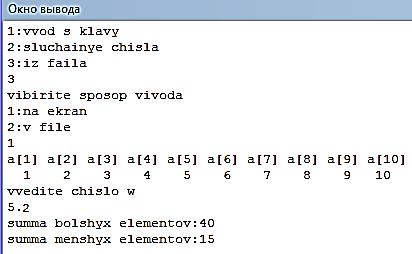


2. Матрица заполняется с клавиатуры, вывод в файл





3. Матрица вводится из файла, выводится на экран



**Тема 14. Обработка массивов переменной длины**

*ПРИМЕР1.* Изменить матрицу, прибавляя к каждому отрицательному элементу значение предыдущего элемента той же строки. Если отрицателен первый элемент строки, прибавить к нему значение последнего элемента строки.

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<time.h>

void main()

{ srand(time(NULL)); clrscr();

FILE \*ms,\*v; int a[3][4],b[3][4],i,j,c,n=3,m=4;

ms=fopen("D:\\student\\fileC.txt","r");

v=fopen("D:\\student\\fileV.txt","w");

metka:

printf("Viberete sposob zadaniya massiva: \n");

printf("1 - ruchnoi vvod, 2 - randomnie znacheniya(ot -50 do 50), 3 - Vvod s faila\n"); scanf("%d",&c);

if(c==1){ for(i=0;i<n;i++)

for(j=0;j<m;j++){printf("\na[%d][%d]= ",i,j);scanf("%d",a[i][j]);}

}

else if(c==2){ for(i=0;i<n;i++)

for(j=0;j<m;j++)a[i][j]=-50+rand()%100;

}

else if(c==3){ for(i=0;i<n;i++)

for(j=0;j<m;j++){ fscanf(ms,"%d",&a[i][j]);}}

else{ goto metka; }

for(i=0;i<n;i++){

for(j=0;j<m;j++){ printf("%d ",a[i][j]); b[i][j]=a[i][j]; }printf("\n"); }

printf("\n");

for(i=0;i<n;i++)

for(j=0;j<m;j++){if(j>0){if(a[i][j]<0)a[i][j]=a[i][j]+b[i][j-1];}

else if(j==0){if(a[i][j]<0)a[i][j]=a[i][j]+a[i][m-1];}

}

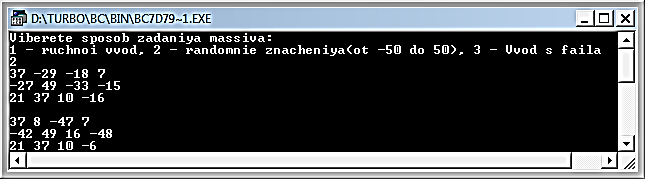
for(i=0;i<n;i++){

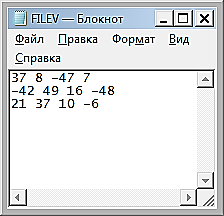
for(j=0;j<m;j++)printf("%d ",a[i][j]); printf("\n"); }

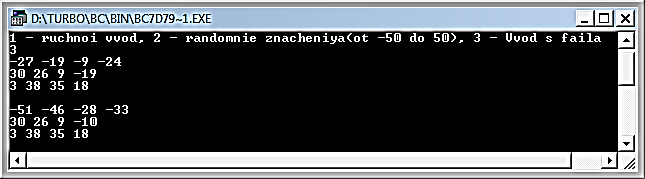
for(i=0;i<n;i++){

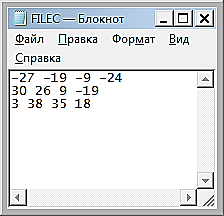
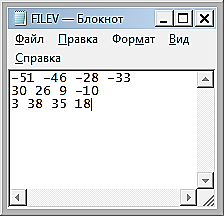
for(j=0;j<m;j++)fprintf(v,"%d ",a[i][j]); fprintf(v,"\n");}

}







ВВОД: ВЫВОД: 

*ПРИМЕР 2*. Дана матрица Р с двумя строками и десятью столбцами, каждым столбцом которой задана абсцисса и ордината одной из десяти точек плоскости. Если нет ни одной пары точек, расстояние между которыми меньше заданной величины R, заменить на нуль в матрице Р все отрицательные абсциссы точек, увеличив ординаты этих точек на R

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<time.h>

void main()

{ srand(time(NULL));

clrscr();

FILE \*ms,\*v; int a[2][10],i,R,j,c,n=2,m=10;

ms=fopen("D:\\student\\fileC1.txt","r");

v=fopen("D:\\student\\fileV1.txt","w");

metka:

printf("Viberete sposob zadaniya massiva: \n");

printf("1 - ruchnoi vvod, 2 - randomnie znacheniya(ot -50 do 50), 3 - Vvod s faila\n");

scanf("%d",&c);

if(c==1){ for(i=0;i<n;i++){

for(j=0;j<m;j++){printf(" a[%d][%d]= ",i,j);scanf("%d",a[i][j]);}printf("\n");

}}

else if(c==2){ for(i=0;i<n;i++)

for(j=0;j<m;j++)a[i][j]=-50+rand()%100;

}

else if(c==3){ for(i=0;i<n;i++)

for(j=0;j<m;j++){ fscanf(ms,"%d",&a[i][j]);}}

else{ goto metka; }

printf("Vvedite R: "); scanf("%d",&R);

for(j=0;j<m;j+=2)

for(i=0;i<n;i+=2){if(((a[i][j]+a[i][j+1])/2+(a[i+1][j]+a[i+1][j+1])/2)>R)goto metka2;}

for(i=0;i<(n-1);i++)

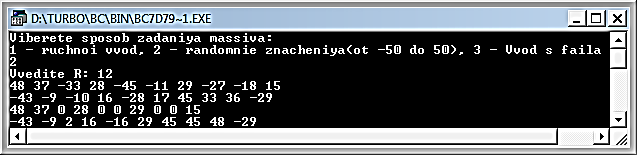
for(j=0;j<n;j++){if(a[i][j]<0){a[i][j]=0;a[i+1][j]+=R;}}

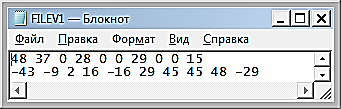
metka2:

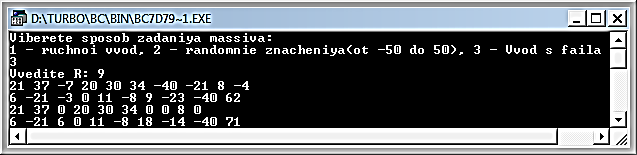
for(i=0;i<n;i++){

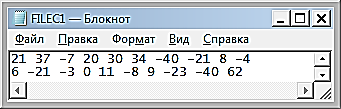
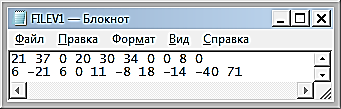
for(j=0;j<m;j++)fprintf(v,"%d ",a[i][j]); fprintf(v,"\n");}

}







ВВОД: 

*ПРИМЕР 3.* Дан массив С1,С2,…,Сn. Изменить матрицу A, увеличив каждый элемент, который меньше элемента главной диагонали, находящегося с ним в одной строке, на C[i]. Подсчитать общее число измененных элементов.

Алгоритм выполнения.

1. диапазон, при котором выполняется данное условие( от i=1 до n, от j:=1 до d)
2. условие: (a[i, j]<a[i, i])( сравниваем текущий элемент с элементом главной диагонали) . если удовлетворяет, то увеличиваем данный элемент в два раза (a[i, j]=a[i, j]+c[i]) и считаем количество таких элементов (kol=kol+1).
3. вывод на экран изменённого массива (a[i, j])
4. вывод на экран количества изменившихся элементов элементов ( kol).

**program** zadacha;

**const** n=10;

**type** s=**array**[1..n,1..n] **of** integer;

ms= **array**[1..n] **of** integer;

**var** i,j,v,k,r,kol:integer;

a:s;c:ms;

**procedure** klav(m:integer; **var** a:s;**var** c:ms);

**var** i,j:integer;

**begin**

writeln('vvedite matricy');

**for** i:=1 **to** m **do begin**

**for** j:=1 **to** m **do begin**

write('a[',i,',',j,']= ');

read(a[i,j]);

**end**;

**end**;

writeln('vvedite massiv');

**for** i:=1 **to** m **do begin**

write('c[',i,']= ');

read(c[i]);

**end**;

**end**;

**procedure** rand(m:integer; **var** a:s;**var** c:ms);

**var** i,j:integer;

**begin**

randomize;

**for** i:=1 **to** m **do**

**for** j:=1 **to** m **do**

a[i,j]:=random(10);

**for** i:=1 **to** m **do**

c[i]:=random(10);

**end**;

**procedure** fil(m:integer; **var** a:s;**var** c:ms);

**var** i,j,k:integer; ft1,ft2:text; t:integer;

**begin**

assign(ft1,'c:\cc\massiv1.txt');

assign(ft2,'c:\cc\massiv2.txt');

i:=1; j:=1;

reset(ft1);

**while not** eof(ft1) **do begin**

read(ft1,t);

a[i,j]:=t;

j:=j+1;

**if** j>n **then begin**

j:=1; i:=i+1;

**end**;

**end**;

k:=1;

**while not** eof(ft2) **do begin**

read(ft1,t);

c[i]:=t;

k:=k+1;

**end**;

close(ft1);

close(ft2);

**end**;

**procedure** ecran(a:s;c:ms);

**var** i,m,j,f:integer;

**begin**

m:=r;

writeln('matrica');

**for** i:=1 **to** m **do begin**

**for** j:=1 **to** m **do begin**

write(a[i,j]:3,' ');

**end**; writeln; **end**;

writeln('massiv');

**for** i:=1 **to** m **do begin**

write(c[i]:3,' ');

**end**;

writeln;

**end**;

**procedure** vfile(a:s);

**var** ft:**file of** real; t:integer;

i,m,j:integer;

**begin** m:=r; assign(ft,'c:\cc\massiv1.txt');

rewrite(ft);

**for** i:=1 **to** m **do begin**

**for** j:=1 **to** m **do begin**

t:=a[i,j];

write(ft,t);

**end**;

**end**;

close(ft);

**end**;

**begin**

v:=0;

writeln('razmernost');

read (r);

writeln('1:vvod s klavy');

writeln('2:sluchainye chisla');

writeln('3:iz faila');

read(v);

**case** v **of**

1: klav(r,a,c);

2: rand(r,a,c);

3: fil(r,a,c); **end**;

writeln('vibirite sposop vivoda');

writeln('1:na ekran');

writeln('2:v file');

read(k);

**case** k **of**

1:ecran(a,c);

2:vfile(a); **end**;

**for** i:=1 **to** r **do begin**

**for** j:=1 **to** r **do begin**

**if** (a[i,j]<a[i,i]) **then begin** a[i,j]:=a[i,j]+c[i];

kol:=kol+1;

**end**;

**end**;

**end**;

writeln('result');

**for** i:=1 **to** r **do begin**

**for** j:=1 **to** r **do**

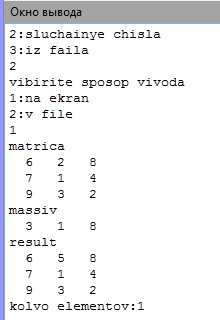
write(a[i,j]:3,' ');

writeln; **end**;

writeln('kolvo elementov:',kol);

readln;

**end**.



**Тема 15. Сложное сочетание данных разной структуры**

*ПРИМЕР 1*.Дана переменная В и матрица А, состоящая из m строк и m столбцов. Элементы С образуют одномерный массив из m элементов.

Требуется составить алгоритм и программу вычисления значений элементов одномерного массива Х1,Х2,..., X m (когда вычисляется Хi, Р равно Х i -1, для первого элемента массива Р=1). S — сумма элементов главной диагонали матрицы А, вычисляемая в алгоритме. Составить программу на Паскале и на Си.

type mas = array [1..100,1..100] of integer;

type vaz = array [1..100] of integer;

var a:mas; c:vaz; k,t,i,j,s,m,b: integer; z,d:string; p,q,n,v:real;

x:array[1..100] of real;

function randomom1(a: mas): mas;

var i:integer;

begin

for i:=1 to m do begin

for j:=1 to m do begin

a[i,j]:= random(20)-10;

end;

end;

randomom1:=a;

end;

function randomom2(c: vaz): vaz;

var i:integer;

begin

for i:=1 to m do begin

c[i]:= random(20)-10;

end;

randomom2:=c;

end;

function klava1(a:mas): mas;

var i:integer;

begin

for i:=1 to m do begin

for j:=1 to m do begin

readln(a[i,j]);

end;

end;

klava1:=a;

end;

function klava2(c:vaz): vaz;

var i:integer;

begin

for i:=1 to m do begin

readln(c[i]);

end;

klava2:=c;

end;

function fail1(a:mas; o:string): mas;

var t:text; i:integer;

begin

assign(t,o);

reset(t);

for i:=1 to m do begin

for j:=1 to m do begin

read(t,a[i,j]);

end;

end;

close(t);

fail1:=a;

end;

function fail2(c:vaz; o:string): vaz;

var t:text; i:integer;

begin

assign(t,o);

reset(t);

for i:=1 to m do begin

read(t,c[i]);

end;

close(t);

fail2:=c;

end;

begin

z:='text1.txt';

d:='text2.txt';

writeln('введите переменную B');

readln(b);

writeln('введите размер массива');

readln(m);

writeln('1 - ввод рандомом');

writeln('2 - ввод руками');

writeln('3 - ввод из файла');

readln(k);

case k of

1: begin a:= randomom1(a); c:= randomom2(c); end;

2: begin a:= klava1(a); writeln('введите массив2'); c:= klava2(c); end;

3: begin a:= fail1(a, z);c:= fail2(c, d);end;

end;

writeln('матрица А: ');

for i:=1 to m do begin

for j:=1 to m do begin

write(a[i,j],' ');

end;

writeln();

end;

writeln('массив С: ');

for i:=1 to m do begin

write(c[i],' ');

end;

writeln();

for i:=1 to m do begin

for j:=1 to m do begin

if (i=j) then

s:= a[i,j]+s;

end;

end;

writeln('полученный массив');

For i:=1 to m do begin

if (i=1) then p:=1

else begin

P:=x[i-1];

end;

n:=((s\*c[i])/p);

For k:=1 to m do begin

v:=1/((b\*a[i,k])+(c[i]\*c[i]));

q:=q+v;

end;

x[i]:=n\*q;

n:=0;

v:=0;

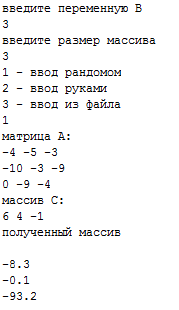
q:=0;

writeln();

write(x[i]:3:1,' ');

end;

end.



**Тема 16. Сложное условие завершения цикла при работе с матрицами**

*ПРИМЕР1* .Дана матрица А с 2 строками и 10 столбцами. Первый элемент каждого столбца рассматривается как диаметр внутренней, а второй – как диаметр внешней окружности кольца. В цикле печатать площади колец, начиная с кольца, представленного первым столбцом, заканчивая цикл, если(по ошибке) второй элемент какого-либо столбца оказывается меньше первого.

**program** lab16;

**var**

mas :**array**[1..2] **of array**[1..10] **of** integer;

menu:byte;

**procedure** vvodmassiv;

**var**

i,j:integer;

**begin**

writeln('Введите матрицу');

**for** j := 1 **to** 10 **do begin**

**for** i := 1 **to** 2 **do begin**

write('mas [',i,',',j,']:='); readln(mas[i,j]);

**end**;

**end**;

**end**;

**procedure** vivodekran;

**var** i,j:integer; vnut,vnesh:integer; s:real;

**begin**

**for** j := 1 **to** 10 **do begin**

vnut:=mas[1,j];

vnesh:=mas[2,j];

**if** vnesh<vnut **then break**;

s:=pi\*(vnesh-vnut);

writeln('s [',j,']:=',s:5:2);

**end**;

**end**;

**procedure** randommassiv;

**var** i,j:integer;

**begin**

**for** i := 1 **to** 2 **do begin**

**for** j := 1 **to** 10 **do begin**

mas[i,j]:=random(100);

write(mas[i,j]:3);

**end**;

writeln;

**end**;

**end**;

**begin**

randomize;

menu:=0;

writeln('1)заполнение random');

writeln('2)ввод массива с клавиатуры');

readln(menu);

**case** menu **of**

1: **begin**

randommassiv;

**end**;

2: **begin**

vvodmassiv;

**end**;

**else**

**begin**

writeln('неверный пункт меню');

readln;

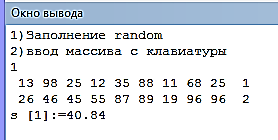
halt;

**end**;

**end**;

vivodekran;

**end**.



**Тема 17. Разработка алгоритмов и программ с выбором структуры данных**

*ПРИМЕР 1.* Задана матрица А с 2 столбцами и 5 строками. Определить номера тех строк, оба элемента которых имеют нулевые значения.

#include "stdafx.h"

#include <stdio.h>

#include <conio.h>

#include "stdlib.h"

#include <time.h>

int v,i,j,e,d,a[5][2],b[10];

double s;

void randomnue(int a[5][2])

{

for (i=0;i<5;i++){

for (j=0;j<2;j++){

a[i][j]=rand()%10;

printf("%d ",a[i][j]);

}

printf("\n");

}

}

void klav(int a[5][2])

{

for (i=0;i<5;i++)

for (j=0;j<2;j++)

scanf("%d ",&a[i][j]);

}

void randomn(int b[10])

{

for (i=0;i<10;i++){

b[i]=rand()%9;

printf("%d ",b[i]);

}

printf("\n");

}

void klava(int a[5])

{

for (i=0;i<5;i++)

scanf("%d ",&a[i]);

}

void rsm()

{int fl=0;

puts("Variant vvoda:\n");

puts("1)Randomnue");

puts("2)S klavuatyru");

puts("Vvedute nomer:");

scanf("%d",&v);

printf("Massiv a:");

printf("\n");

switch(v)

{

case 1:randomnue(a);break;

case 2:klav(a);break;

}

for(i=0;i<5;i++)

{

if ((a[i][0]==0) && (a[i][1]==0))

{printf("%d ",i+1);fl++;}

}

if (!fl) printf("net nulei ");

}

void rso()

{int fl=0;

puts("Variant vvoda:\n");

puts("1)Randomnue");

puts("2)S klavuatyru");

puts("Vvedute nomer:");

scanf("%d",&v);

printf("Massiv b:");

printf("\n");

switch(v)

{

case 1:randomn(b);break;

case 2:klava(b);break;

}

for(i=0;i<5;i++)

{

if ((b[i]==0) && (b[i+5]==0))

{printf("nomer %d ",i);fl++;} else fl=0;

}

if (!fl) printf("net nulei ");

}

void v3()

{

printf("chisla \n\n");

for(i=0;i<5;i++)

{

d=rand()%49;

e=rand()%49;

if(d>e)

printf("%4d",d);

}

}

void main()

{

int c,fl=0;

srand(time(NULL));

puts("Menu:\n");

puts("1)Rabota s matruzey");

puts("2)Rabota s massivamu");

puts("3)Rabota s peremennymi");

puts("Vvedute nomer:");

scanf("%d",&v);

printf("\n");

switch(v)

{

case 1:rsm();break;

case 2:rso();break;

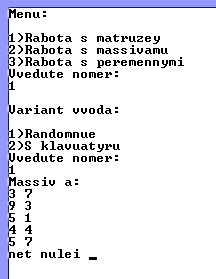
case 3:v3();break;

}

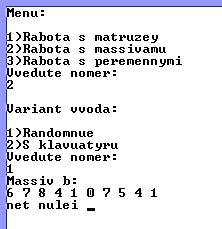
getch();

}

Работа с матрицей случайных чисел

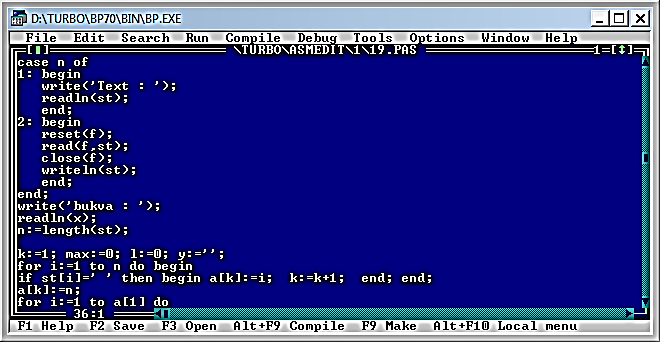


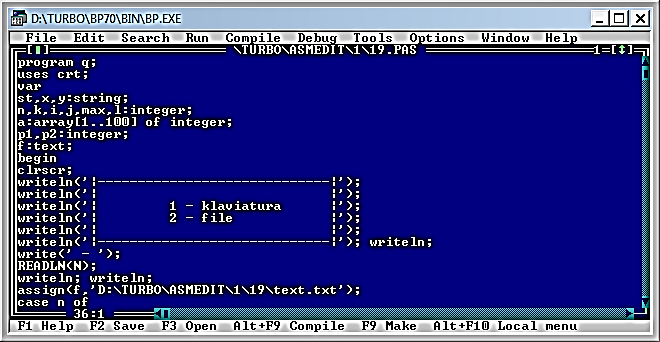
Работа с одномерным массивом случайных чисел

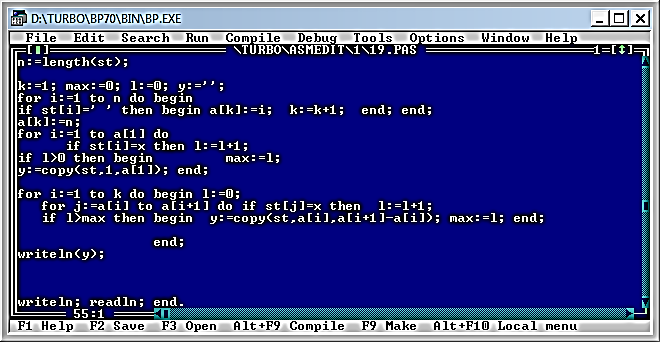


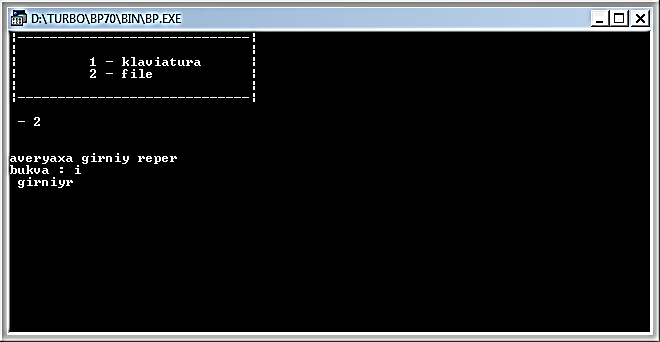
**Тема 18. Обработка символьной информации**

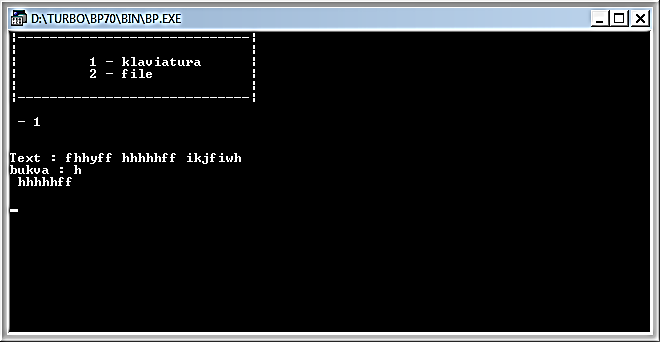
*ПРИМЕР 1.* В заданном тексте определить слово с наибольшим вхождением заданного символа.











**Тема 19. Программирование с использованием комбинированного типа данных. Построение модели, создание типизированного файла**

*ПРИМЕР 1.* Сведения о компьютерах некоторого учреждения состоят из марки компьютера, даты приобретения, номера комнаты, где находится компьютер, и фамилии ответственного лица. Вывести сведения о компьютерах PentiumIV, приобретенных в 2000 году.

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<math.h>

#include<time.h>

FILE \*v,\*vi; int i,n;

struct { char mark[20];

char data[10];

char nom[5];

char fam[15]; } tab[20];

int input()

{

printf("Vvedite kolichestvo strok v ishodnom file - "); scanf("%d",&n);

return n;

}

void head()

{

printf("|------------------------------------------------------------------------------|\n");

printf("|------Marka-------|------God------|------Nomer-------|---------Familia--------|\n");

printf("|------------------|---------------|------------------|------------------------|\n");

}

int input\_file(int n)

{

for(i=0;i<n;i++){ fscanf(v,"%s",tab[i].mark); fscanf(v,"%s",tab[i].data); fscanf(v,"%s",tab[i].nom); fscanf(v,"%s",tab[i].fam); }

fclose(v);

}

int output\_file(int n)

{

for(i=0;i<n;i++){ if(!strcmp(tab[i].data,"2000")){

fprintf(vi,"%s ",tab[i].mark); fprintf(vi,"%s ",tab[i].data); fprintf(vi,"%s ",tab[i].nom);

fprintf(vi,"%s ",tab[i].fam); fprintf(vi,"\n");}}

fclose(vi);

}

int output(int n)

{

for(i=0;i<n;i++){ if(!strcmp(tab[i].data,"2000"))

printf("|-----%s-----|------%s-----|--------%s-------|--------%s---------|\n",tab[i].mark,tab[i].data,tab[i].nom,tab[i].fam);}

}

int main()

{ v=fopen("D:\\student\\21prak\_vvod.txt","r");

vi=fopen("D:\\student\\21prak\_vivod.txt","w");

input();

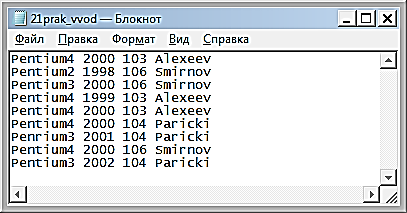
head();

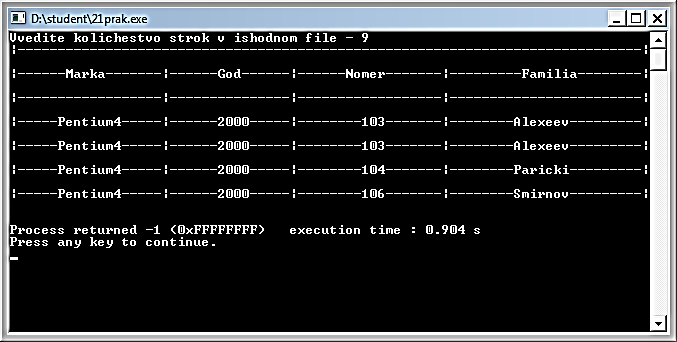
input\_file(n);

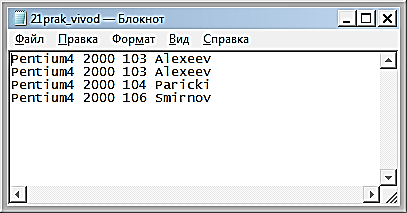
output(n);

output\_file(n);

}







*ПРИМЕР 2.* Багаж авиапассажира характеризуется фамилией пассажира, количеством вещей и общим весом вещей. Вывести фамилии пассажиров багаж составляет больше двух вещей и вычислить средний вес багажа всех пассажиров.

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <math.h>

int v,i,j,e,d,a[5][2],b[10];

double s;

void randomnue(int a[5][2])

{

for (i=0;i<5;i++){

for (j=0;j<2;j++){

a[i][j]=rand()%9+1;

printf("%d ",a[i][j]);

}

printf("\n");

}

}

void klav(int a[5][2])

{

for (i=0;i<5;i++)

for (j=0;j<2;j++)

scanf("%d ",&a[i][j]);

}

void randomn(int b[10])

{

for (i=0;i<10;i++){

b[i]=rand()%9+1;

printf("%d ",b[i]);

}

printf("\n");

}

void klava(int a[5])

{

for (i=0;i<5;i++)

scanf("%d ",&a[i]);

}

void rsm()

{

puts("Variant vvoda:\n");

puts("1)Randomnue");

puts("2)S klavuatyru");

puts("Vvedute nomer:");

scanf("%d",&v);

printf("Massiv a:");

printf("\n");

switch(v)

{

case 1:randomnue(a);break;

case 2:klav(a);break;

}

}

void rso()

{

puts("Variant vvoda:\n");

puts("1)Randomnue");

puts("2)S klavuatyru");

puts("Vvedute nomer:");

scanf("%d",&v);

printf("Massiv b:");

printf("\n");

switch(v)

{

case 1:randomn(b);randomn(b);randomn(b);break;

case 2:klava(b);klava(b);klava(b);break;

}

}

void v3()

{

printf("chisla \n\n");

for(i=0;i<5;i++)

{

d=rand()%49+1;

e=rand()%49+1;

if(d>e)

printf("%4d",d);

}

}

void main()

{

clrscr();

int c;

srand(time(NULL));

puts("Menu:\n");

puts("1)Rabota s matruzey");

puts("2)Rabota s massivamu");

puts("3)Rabota s peremennymi");

puts("Vvedute nomer:");

scanf("%d",&v);

printf("\n");

switch(v)

{

case 1:rsm();break;

case 2:rso();break;

//case 3:rsp();break;

case 3:v3();break;

}

{

for(i=0;i<5;i++)

{

if ((a[i][0]==0) && (a[i][1]==0))

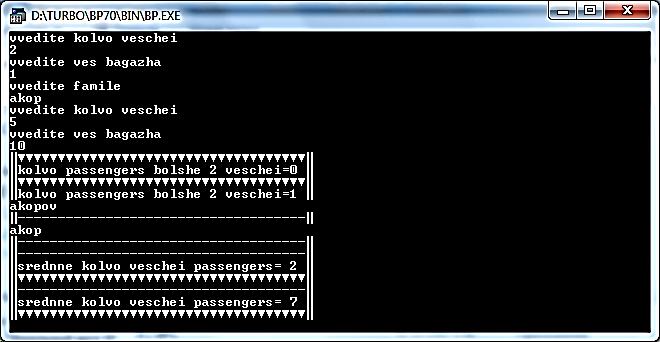
printf("%d ",i);

}

}

getch();

}



**Тема 20. Построение графика функции**

*ПРИМЕР 1.*Построить графики функций y1(x)=cos(x-1)+abs(x), y2(x)=sin(x)-abs(x+1) на отрезке [-5; 5]

a) каждую функцию в отдельной области экрана

б) обе функции в одной области экрана

program graph;

uses crt,graphABC;

var y:real; var x,b:integer;

begin

moveto(50,0);

lineto(50,100);

moveto(0,50);

lineto(100,50);

moveto(250,0);

lineto(250,100);

moveto(200,50);

lineto(300,50);

moveto(150,200);

lineto(150,300);

moveto(100,250);

lineto(200,250);

x:=-5;

y:=-cos(x-1)-abs(x);

b:=round(y);

moveto(3\*x+50,6\*b+50);

for x:=-5 to 5 do begin

y:=-cos(x-1)-abs(x); b:=round(y);

setpencolor(clred);

lineto(3\*x+50,6\*b+50);

end;

x:=-5; y:=-cos(x-1)-abs(x); b:=round(y); moveto(3\*x+150,6\*b+250);

for x:=-5 to 5 do begin

y:=-cos(x-1)-abs(x); b:=round(y);

setpencolor(clred);

lineto(3\*x+150,6\*b+250);

end;

x:=-5; y:=-sin(x)+abs(x+1); b:=round(y); moveto(3\*x+250,6\*b+50);

for x:=-5 to 5 do begin

y:=-sin(x)+abs(x+1); b:=round(y);

setpencolor(clblue);

lineto(3\*x+250,6\*b+50);

end;

x:=-5; y:=-sin(x)+abs(x+1); b:=round(y); moveto(3\*x+150,6\*b+250);

for x:=-5 to 5 do begin

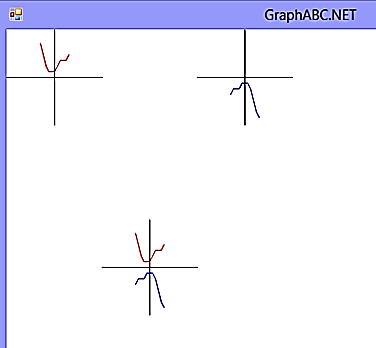
y:=-sin(x)+abs(x+1); b:=round(y);

setpencolor(clblue);

lineto(3\*x+150,6\*b+250);

end;

end.

****

**Тема21. Динамические структуры данных**

*ПРИМЕР 1*. Дан массив b[15]. Требуется данные из массива записать в стек. При выводе стека на экран предусмотреть вывод на экран только положительных элементов.

#include "stdafx.h"

#include <stdlib.h>

#include <iomanip>

#include <conio.h>

#include "locale.h"

struct Node

{

int d;

Node \*p;

};

Node \*first(int d);

void push(Node \*\*top, int d);

int pop(Node \*\*top);

int main()

{

setlocale(LC\_ALL,"Russian");

int k,i, b[15];

printf( "\n\nВвод массива а:\n");

for (i=0; i<15; i++)

{

printf( "Введите элемент массива b [%d]=",i);

scanf("%d", &b[i]);

}

Node \*top = first(b[0]);

for (i=1; i<15; i++)

{

k=b[i];

push(&top, k);

}

printf("\n");

printf( "Исходный массив b:\n");

for (i=0; i<15; i++)

{

printf("%5d", b[i]);

}

printf("\n");

printf( " Полученный стек:\n");

while (top)

pop(&top);

getch();

return 0;

}

// начальное формирование стека

Node \*first (int d)

{

Node \*pv=new Node;

pv->d=d;

pv->p=0;

return pv;

}

// занесение в стек

void push (Node \*\*top, int d)

{

Node \*pv = new Node;

pv->d=d;

pv->p= \*top;

\*top=pv;

}

// выборка из стека

int pop (Node \*\*top)

{

int temp = (\*top)->d;

Node \*pv=\*top;

\*top=(\*top)->p;

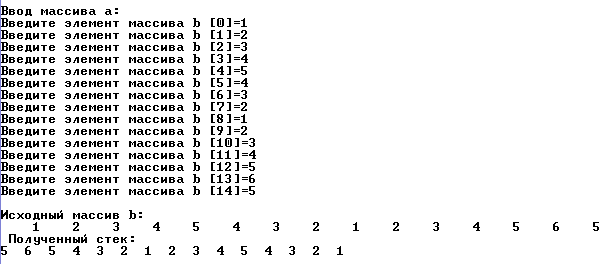
delete pv;

if ( temp>0)

printf("%d ", temp);

return temp;

}

Результат: 

* 1. **Материалы справочного характера**

***Процедуры и функции графического модуля Паскаля***

**Окружность, эллипс, дуга, сектор**

*Процедура* **Circle** (x,y: integer; r: word) вычерчивает окружность радиуса r с центром в точке с координатами (x, y).

*Процедура* **Arc** (x, y, ugol\_begin, ugol\_end, r: integer) вычерчивает дугу окружности радиуса r с центром в точке с координатами (x, y). Параметры ugol\_begin и ugol\_end задают угловые координаты начала и конца дуги. Отсчет углов ведется против часовой стрелки. Значения угловых координат задается в градусах.

*Процедура* **Ellipse** (x, y: integer; ugol\_begin, ugol\_end, rx, ry: word) вычерчивает эллипс или дугу эллипса с центром в точке с координатами   
(x, y). Параметры ugol\_begin и ugol\_end задают угловые координаты начала и конца дуги. Параметры rx и ry определяют горизонтальный и вертикальный радиусы эллипса.

*Процедура* **PieSlice** (x, y: integer; ugol\_begin, ugol\_end, r: word) вычерчивает сектор окружности радиуса r с центром в точке с координатами (x, y). Параметры ugol\_begin и ugol\_end задают угловые координаты начала и конца сектора.

Сектор может быть закрашен в соответствии со стилем, заданным процедурой SetFillStyle (о ней чуть позже).

*Процедура* **Sector** (x, y: integer; ugol\_begin, ugol\_end, rx, ry: word) вычерчивает сектор эллипса с центром в точке с координатами (x, y) и горизонтальным радиусом rx, вертикальным - ry. Параметры ugol\_begin и ugol\_end задают угловые координаты начала и конца сектора.

Сектор может быть закрашен в соответствии со стилем, заданным процедурой SetFillStyle.

**Прямоугольник, закрашенный прямоугольник, параллелепипед**

*Процедура* **Rectangle** (x1, y1, x2, y2: integer) вычерчивает контур прямоугольника. Параметры x1, y1 задают положение левого верхнего угла, x2, y2 – правого нижнего.

*Процедура* **Bar** (x1, y1, x2, y2: integer) вычерчивает закрашенный прямоугольник. Параметры x1, y1 задают положение левого верхнего угла, x2, y2 – правого нижнего. Стиль и цвет заливки определяется процедурой SetFillStyle.

*Процедура* **Bar3D** (x1,y1,x2,y2: integer; глубина: word; граница: boolean) вычерчивает параллелепипед. Параметры x1, y1 задают положение левого верхнего угла, x2, y2 – правого нижнего угла ближней грани. Параметр глубина задает расстояние между передней и задней гранями в пикселях. Параметр граница определяет, нужно ли вычерчивать верхнюю границу задней грани параллелепипеда. Стиль и цвет заливки ближней грани определяется процедурой SetFillStyle.

**Вывод текста в графическом режиме**

*Процедура* **OutText** (text: string) выводит строку символов text от текущей позиции указателя вывода и перемещает указатель в точку, расположенную за последним выведенным символом.

*Процедура* **OutTextXY** (x, y: integer; text: string) выводит строку символов text, начиная с точки с координатами (x, y), при этом указатель своего положения не меняет, т.е. остается в точке (x, y).

**Стиль вычерчиваемых линий, контуров**

*Процедура* **SetLineStyle** (type, pattern, thick: word) устанавливает стиль вычерчиваемых линий. Здесь type, pattern, thick – соответственно тип, образец и толщина линии.

Тип линии может быть задан с помощью одной из следующих констант:

SolidLn=0 {сплошная линия}

DottedLn=1 {точечная линия}

CenterLn=2 {штрих-пунктирная линия}

DashedLn=3 {пунктирная линия}

UserBitLn=4 {узор определяет пользователь}

Параметр Pattern учитывается только для линий, вид которых определяется пользователем, т.е. если type=4. Во всех остальных случаях можно поставить любое значение типа word (но обязательно поставить, все-таки это параметр процедуры, значит должен быть).

Каким образом можно задать пользовательский тип линии? Под тип линии отводится переменная типа word, т.е. два байта. Эти два байта и определяют образец линии: каждый установленный в единицу бит этого слова соответствует светящейся точке, нулевой бит - несветящийся пиксель. Таким образом, задается отрезок линии длиной в 16 пикселей. Этот образец периодически повторяется по всей длине линии.

Параметр thick может принимать одно из двух значений:

NormWidth=1 {толщина в 1 пиксель}

ThickWidth=3 {толщина в 3 пикселя}

**Стиль и цвет заливки**

*Процедура* **SetFillStyle** (style, color: word) устанавливает стиль и цвет заливки (закрашивания) областей (Bar, Bar3D, Sector и др.). В качестве параметра style используют одну из констант:

EmptyFill=0 {заливка цветом фона}

SolidFill=1 {сплошная заливка текущим цветом}

LineFill=2 {горизонтальная штриховка}

LtSlashFill=3 {штриховка под углом 45° влево тонкими линиями}

SlashFill=4 {штриховка под углом 45° влево утолщенными линиями}

BkSlashFill=5 {штриховка под углом 45° вправо утолщенная}

LtBkSlashFill=6 {штриховка под углом 45° вправо тонкая}

HatchFill=7 {заполнение +++++}

XHatchFill=8 {заполнение под углом 45° редкой косой клеткой}

InterleaveFill=9 {заполнение под углом 45° частой косой клеткой}

WideDotFill=10 {заполнение редкими точками}

CloseDotFill=11 {заполнение частыми точками}

UserFill=12 {узор определяется пользователем}

**Стиль вывода текста**

*Процедура* **SetTextStyle** (font, orient, size: word) устанавливает шрифт font, ориентацию orient и размер size текста, выводимого на экран. Параметр font может принимать одну из констант:

DefaultFont=0 {стандартный, каждый символ размером 8\*8 пикселей}

TriplexFont=1 {Triplex шрифт}

SmallFont=2 {мелкий}

SansSerifFont=3 {SansSerif шрифт}

GothicFont=4 {готический}

В 7.0 версии Паскаля набор шрифтов значительно расширен, но для новых шрифтов не придуманы мнемонические константы, поэтому можно использовать такие номера шрифтов:

1. - «рукописный» шрифт (scri.chr);
2. - одноштриховой шрифт типа Courier (simp.chr);
3. - наклонный шрифт типа Times Italic (tscr.chr);
4. - шрифт типа Times Rovan (lcom.chr);
5. - шрифт типа Courier увеличенного размера (euro.chr);
6. - крупный двухштриховой шрифт (bold.chr).

**Замечание:** все шрифты, кроме стандартного (матричного), являются векторными, что позволяет изменять их размеры без ухудшения качества. Каждый из этих шрифтов размещается в отдельном файле. Для использования этих шрифтов необходимо разместить соответствующий файл в рабочем каталоге, в противном случае вызов этого шрифта игнорируется и подключается стандартный шрифт.

Параметр orient задает ориентацию выводимого текста:

HorizDir=1 {слева направо}

VertDir=2{снизу вверх}

Каждый шрифт способен десятикратно изменять свои размеры. Размер шрифта задается параметром size, который может иметь значения от 1 до 10 (точечный или матричный шрифт – в диапазоне от 1 до 32).

**Заполнение (закрашивание) произвольной замкнутой фигуры**

*Процедура* **FloodFill** (x, y: integer; border: word) заполняет произвольную замкнутую фигуру, используя текущий стиль и цвет заполнения. Координаты точки (x, y) указывают, начиная с какой точки будет производиться заливка. Если точка находится внутри замкнутой фигуры, то будет закрашена внутренняя область. Если фигура не замкнута, то заливка разольется по всему экрану. Параметр border указывает цвет граничной линии.

**Очистка графического экрана**

*Процедура* **ClearDevice** очищает графический экран, устанавливает указатель в левый верхний угол.

**Таблица1. Константы цветов**

|  |  |  |
| --- | --- | --- |
| Имя константы | Номер цвета | Цвет |
| Black | 0 | Черный |
| Blue | 1 | Темно-синий |
| Green | 2 | Темно-зеленый |
| Cyan | 3 | Бирюзовый |
| Red | 4 | Красный |
| Magenta | 5 | Фиолетовый |
| Brown | 6 | Коричневый |
| LightGray | 7 | Светло-серый |
| DarkGray | 8 | Темно-серый |
| LightBlue | 9 | Синий |
| LightGreen | 10 | Светло-зеленый |
| LightCyan | 11 | Светло-бирюзовый |
| LightRed | 12 | Розовый |
| LightMagenta | 13 | Малиновый |
| Yellow | 14 | Желтый |
| White | 15 | Белый |

**2.5 Тест**

**Тесты на основные алгоритмические конструкции, массивы данных.**

Использованы материалы сайта <http://kpolyakov.spb.ru>

|  |  |  |
| --- | --- | --- |
| № п/п | задание | ответ |
| 1 | Значения двух массивов A[1..100] и B[1..100] задаются с помощью следующего фрагмента программы:  **for n:=1 to 100 do**  **A[n] := n - 10;**  **for n:=1 to 100 do**  **B[n] := A[n]\*n;**  Сколько элементов массива B будут иметь положительные значения?  1) 10 2) 50 3) 90 4) 100 |  |
| 2 | Все элементы двумерного массива A размером 10х10 элементов первоначально были равны 0. Затем значения элементов меняются с помощью вложенного оператора цикла в представленном фрагменте программы:  **for n:=1 to 4 do**  **for k:=n to 4 do begin**  **A[n,k] := A[n,k] + 1;**  **A[k,n] := A[k,n] + 1;**  **end;**  Сколько элементов массива в результате будут равны 1?  1) 0 2) 16 3) 12 4) 4 |  |
| 3 | Значения двумерного массива задаются с помощью вложенного оператора цикла в представленном фрагменте программы:  **for n:=1 to 5 do**  **for k:=1 to 5 do**  **B[n,k] := n + k;**  Чему будет равно значение B[2,4]?  1) 9 2) 8 3) 7 4) 6 |  |
| 4 | Дан фрагмент:  **for n:=l to 6 do**  **for m:=l to 5 do begin**  **C[n,m]:=C[n,m]+(2\*n-m);**  **end;**  Чему будет равно значение С[4,3], если перед этими командами значение С[4,3]=10?  1) 5 2) 10 3) 15 4) 25 |  |
| 5 | Значения элементов двух массивов А и В размером 1 х 100 задаются с помощью следующего фрагмента программы:  **for i:=1 tо 100 do**  **A[i] := 50 – i;**  **for i:=1 tо 100 do**  **B[i] := A[i] + 49;**  Сколько элементов массива В будут иметь отрицательные значения?  1) 1 2) 10 3) 50 4) 100 |  |
| 6 | Значения элементов двумерного массива А были равны 0. Затем значения некоторых элементов были изменены:  **n := 0;**  **for i:=1 tо 5 do**  **for j:=1 tо 6-i do begin**  **n := n + 1;**  **A[i,j] := n;**  **end;**  Какой элемент массива будет иметь в результате максимальное значение?  1) A[1,1] 2) A[1,5] 3) A[5,1] 4) A[5,5] |  |
| 7 | Значения элементов двумерного массива А размером 5x5 задаются с помощью вложенного цикла в представленном фрагменте программы:  **for i:=1 tо 5 do**  **for j:=1 tо 5 do begin**  **A[i,j] := i\*j;**  **end;**  Сколько элементов массива будут иметь значения больше 10?  1) 12 2) 8 3) 10 4) 4 |  |
| 8 | Значения элементов двумерного массива А размером 5x5 задаются с помощью вложенного цикла в представленном фрагменте программы:  **for i:=1 tо 5 do**  **for j:=1 tо 5 do begin**  **A[i,j] := i + j;**  **end;**  Сколько элементов массива будут иметь значения больше 5?  1) 5 2) 20 3) 10 4) 15 |  |
| 9 | Дан фрагмент программы:  **for n:=1 tо 5 do**  **for m:=1 tо 5 do**  **C[n,m] := (m – n)\*(m – n);**  Сколько элементов массива С будут равны 1?  1) 5 2) 2 3) 8 4) 14 |  |
| 10 | В программе описан одномерный целочисленный массив с индексами от 0 до 10. В приведенном ниже фрагменте программы массив сначала заполняется, а потом изменяется:  **for i:=0 to 10 do**  **A[i]:= i + 1;**  **for i:=1 to 10 do**  **A[i]:= A[i-1];**  Как изменяются элементы этого массива?  1) все элементы, кроме последнего, сдвигаются на 1 элемент вправо  2) все элементы, кроме первого, сдвигаются на 1 элемент влево  3) все элементы окажутся равны 1  4) все элементы окажутся равны своим индексам |  |
| 11 | Определите значение целочисленных переменных a и b после выполнения фрагмента программы:  **a := 3 + 8\*4;**  **b := (a div 10) + 14;**  **a := (b mod 10) + 2;** |  |
| 12 | Определите значение целочисленных переменных a и b после выполнения фрагмента программы:  **a := 1819;**  **b := (a div 100)\*10+9;**  **a := (10\*b–a) mod 100;** |  |
| 13 | Определите значение целочисленных переменных a и b после выполнения фрагмента программы:  **a := 42;**  **b := 14;**  **a := a div b;**  **b := a\*b;**  **a := b div a;** |  |
| 14 | В результате выполнения фрагмента программы  **while n <> 0 do begin**  **write ( 2\*(n mod 5 + 3) );**  **n := n div 10;**  **end;**  на экран выведено число 10614. Какое число хранилось до этого в переменной **n**, если известно, что все цифры в нем нечетные? |  |
| 15 | Определите значение переменной **b** после выполнения следующего фрагмента программы, где **a** и **b** – вещественные (действительные) переменные:  **a := -5;**  **b := 5 + 7 \* a;**  **b := b / 2 \* a;** |  |
| 16 | Определите значение переменной **c** после выполнения следующего фрагмента программы:  **a := 100;**  **b := 30;**  **a := a – b\*3;**  **if a > b then**  **c := a – b else c := b – a;** |  |
| 17 | Определите значение переменной **c** после выполнения следующего фрагмента программы:  **a := 6;**  **b := 15;**  **a := b – a\*2;**  **if a > b then**  **c := a + b else c := b – a;** |  |
| 18 | Определите значение переменной **c** после выполнения следующего фрагмента программы:  **a := -5;**  **b := 14;**  **b := b + a\*2;**  **if a > b then**  **c := a + b else c := b – a;** |  |
| 19 | Определите значение переменной **c** после выполнения следующего фрагмента программы:  **a := -5;**  **b := 3;**  **a := a - b\*2;**  **if a > b then**  **c := b - a**  **else c := a – b;** |  |
| 20 | Определите значение переменной **S** после выполнения следующего фрагмента программы:  **S:=1; i:=1;**  **while i < 5 do begin**  **S := S + i\*(i+1);**  **i := i + 1;**  **end;** |  |
| 21 | Определите значение переменной **с** после выполнения следующего фрагмента программы:  **x:= 2.5E+02;**  **x:= x + 0.5E+02;**  **y:= -x;**  **c:= -2\*y - x;** |  |
| 22 | Определите значение переменной **с** после выполнения следующего фрагмента программы:  **a := 30;**  **b := 6;**  **a := a \* 3 / b;**  **if a < b then**  **c := 3\*a - 5 \*(b+3)**  **else c := 3\*a + 5 \* (b+3);** |  |
| 23 | Определите, что будет напечатано в результате работы следующего фрагмента программы:  **var k, s: integer;**  **begin**  **s:=0;**  **k:=0;**  **while k < 30 do begin**  **k:=k+3;**  **s:=s+k;**  **end;**  **write(s);**  **end.** |  |
| 24 | Определите, что будет напечатано в результате работы следующего фрагмента программы:  **var k, s: integer;**  **begin**  **s:=3;**  **k:=1;**  **while k < 25 do begin**  **s:=s+k;**  **k:=k+2;**  **end;**  **write(s);**  **end.** |  |
| 25 | Определите, что будет напечатано в результате работы следующего фрагмента программы:  **var k, s: integer;**  **begin**  **s:=0;**  **k:=1;**  **while s < 66 do begin**  **k:=k+3;**  **s:=s+k;**  **end;**  **write(k);**  **end.** |  |
| 26 | Определите, что будет напечатано в результате работы следующего фрагмента программы:  **var k, s: integer;**  **begin**  **s:=1;**  **k:=0;**  **while k < 13 do begin**  **s:=s+2\*k;**  **k:=k+4;**  **end;**  **write(s+k);**  **end.** |  |
| 27 | Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:  **F(1) = 1**  **F(n) = F(n–1) \* (n + 1), при n > 1**  Чему равно значение функции F(5)? В ответе запишите только натуральное число. |  |
| 28 | Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:  **F(1) = 1**  **F(n) = F(n–1) \* (n + 2), при n > 1**  Чему равно значение функции F(5)? В ответе запишите только натуральное число. |  |
| 29 | Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:  **F(1) = 1**  **F(n) = F(n–1) \* (2\*n + 1), при n > 1**  Чему равно значение функции F(4)? В ответе запишите только натуральное число. |  |
| 30 | Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:  **F(1) = 1**  **F(n) = F(n–1) \* (2\*n - 1), при n > 1**  Чему равно значение функции F(5)? В ответе запишите только натуральное число. |  |
| 31 | Ниже записана программа. Получив на вход число , эта программа печатает два числа,  и . Укажите наибольшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 7.  **var x, L, M: integer;**  **begin**  **readln(x);**  **L:=0; M:=0;**  **while x > 0 do begin**  **L:= L + 1;**  **M:= M + x mod 10;**  **x:= x div 10;**  **end;**  **writeln(L); write(M);**  **end.** |  |
| 32 | Ниже записана программа. Получив на вход число , эта программа печатает два числа,  и . Укажите наибольшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 8.  **var x, L, M: integer;**  **begin**  **readln(x);**  **L:=0; M:=0;**  **while x > 0 do begin**  **L:= L + 1;**  **if x mod 2 = 0 then**  **M:= M + x mod 10;**  **x:= x div 10;**  **end;**  **writeln(L); write(M);**  **end.** |  |
| 33 | Ниже записана программа. Получив на вход число , эта программа печатает два числа,  и . Укажите наибольшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 0.  **var x, L, M: integer;**  **begin**  **readln(x);**  **L:=0; M:=0;**  **while x > 0 do begin**  **L:= L + 1;**  **if x mod 2 = 0 then**  **M:= M + x mod 10;**  **x:= x div 10;**  **end;**  **writeln(L); write(M);**  **end.** |  |
| 34 | Ниже записана программа. Получив на вход число , эта программа печатает два числа,  и . Укажите наибольшее из таких чисел , при вводе которых алгоритм печатает сначала 3, а потом 8.  **var x, L, M: integer;**  **begin**  **readln(x);**  **L:=0; M:=0;**  **while x > 0 do begin**  **L:= L + 1;**  **if x mod 2 = 1 then**  **M:= M + x mod 10;**  **x:= x div 10;**  **end;**  **writeln(L); write(M);**  **end.** |  |
| 35 | Определите, какое число будет напечатано в результате выполнения следующего алгоритма:  **Var a,b,t,M,R:integer;**  **Function F(x:integer):integer;**  **begin**  **F:=(5-x)\*(x+3);**  **end;**  **BEGIN**  **a:=-20; b:=20;**  **M:=a; R:=F(a);**  **for t:=a to b do begin**  **if (F(t)>R)then begin**  **M:=t;**  **R:=F(t);**  **end;**  **end;**  **write(R);**  **END.** |  |
| 36 | Определите, какое число будет напечатано в результате выполнения следующего алгоритма:  **Var a,b,t,M,R:integer;**  **Function F(x:integer):integer;**  **begin**  **F:=(x-5)\*(x+3);**  **end;**  **BEGIN**  **a:=-5; b:=5;**  **M:=a; R:=F(a);**  **for t:=a to b do begin**  **if (F(t)>R)then begin**  **M:=t;**  **R:=F(t);**  **end;**  **end;**  **write(M);**  **END.** |  |
| 37 | Определите, какое число будет напечатано в результате выполнения следующего алгоритма:  **Var a,b,t,M,R:integer;**  **Function F(x:integer):integer;**  **begin**  **F:=(x+7)\*(1-x);**  **end;**  **BEGIN**  **a:=-5; b:=5;**  **M:=a; R:=F(a);**  **for t:=a to b do begin**  **if (F(t)< R)then begin**  **M:=t;**  **R:=F(t);**  **end;**  **end;**  **write(M);**  **END.** |  |
| 38 | Определите, какое число будет напечатано в результате выполнения следующего алгоритма:  **Var a,b,t,M,R:integer;**  **Function F(x:integer):integer;**  **begin**  **F:= 2\*x\*x + 8\*x + 10;**  **end;**  **BEGIN**  **a:=-10; b:=10;**  **M:=a; R:=F(a);**  **for t:=a to b do begin**  **if (F(t)< R)then begin**  **M:=t;**  **R:=F(t);**  **end;**  **end;**  **write(M);**  **END.** |  |
| 39 | Определите, какое число будет напечатано в результате выполнения следующего алгоритма:  **Var a,b,t,M,R:integer;**  **Function F(x:integer):integer;**  **begin**  **F:= x\*x + 6\*x + 10;**  **end;**  **BEGIN**  **a:=-10; b:=10;**  **M:=a; R:=F(a);**  **for t:=a to b do begin**  **if (F(t)> R)then begin**  **M:=t;**  **R:=F(t);**  **end;**  **end;**  **write(M);**  **END.** |  |
| 40 | Определите, какое число будет напечатано в результате выполнения следующего алгоритма:  **Var a,b,t,M,R:integer;**  **Function F(x:integer):integer;**  **begin**  **F:= x\*x - 8\*x + 10;**  **end;**  **BEGIN**  **a:=-5; b:=5;**  **M:=a; R:=F(a);**  **for t:=a to b do begin**  **if (F(t)> R)then begin**  **M:=t;**  **R:=F(t);**  **end;**  **end;**  **write(R);**  **END.** |  |
| 41 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  for (i = 0; i < 5; i = i + 2)  {  printf("%i", i);  i = i + 1;  } |  |
| 42 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  for (i = 0; i < 5; i = i + 2)  {  printf("%i", i);  i = i – 1;  } |  |
| 43 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  for (i = 0; i < 5; i ++)  {  if (i % 2 == 0)  printf("A");  else  continue;  printf("%i", i);  } |  |
| 44 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  do  {  printf("%i", i);  i = i + 1;  } while (i % 4 != 0); |  |
| 45 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  while (1)  {  for (j = 0; j < 3; j ++)  {  printf("%i%i", i, j);  i = i + 1;  if (i > 2)  break;  }  } |  |
| 46 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  for (i = 0; i < 5; i ++)  if (i % 3 == 0)  if (i % 2 == 0)  printf("A");  else  printf("B"); |  |
| 47 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  for (i = 0; i < 10; i ++)  switch (i)  {  case 1:  printf("A");  case 3:  printf("B");  default:  printf("C");  } |  |
| 48 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  for (i = 0; i < 10; i ++);  if (i % 5 == 0)  printf("%i", i); |  |
| 49 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  i = 0;  for (i < 3)  {  printf("%i", i \* 2);  i ++;  } |  |
| 50 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  while (i < 100)  {  if (i == 5)  break;  else  continue;  printf("%i", i);  }  printf("%i", i); |  |
| 51 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  do  printf("%i", i);  while (i < 10);  i ++;  while (i < 20);  i ++; |  |
| 52 | Переменным *i* и *j* (типа *int*) присвоены нулевые значения. Что будет выведено в результате работы следующего фрагмента:  while (i < 10)  while (j < 10)  {  while (i + j < 10)  {  printf("%i", i);  if (i + j == 8)  break;  else  i ++;  }  } |  |
| 53 | Чему равно значение выражения:  1 & 3 && 5 |  |
| 54 | Чему равно значение выражения:  sizeof 3.5 \* 2 |  |
| 55 | Чему равно значение выражения:  1 == 2 < 3 + 5 > 3 |  |
| 56 | Чему равно значение выражения:  (int) 1 / 2.0 + (float) 1 / 2 |  |
| 57 | Все используемые переменные описаны и имеют тип *int*. Что будет выведено в результате работы следующих фрагментов:  i = 2;  printf("%i", i ? 3 : 2); |  |
| 58 | Все используемые переменные описаны и имеют тип *int*. Что будет выведено в результате работы следующих фрагментов:  i = 1;  if (!i)  printf("A");  if (!!i)  printf("B");  else  printf("C"); |  |
| 59 | Все используемые переменные описаны и имеют тип *int*. Что будет выведено в результате работы следующих фрагментов:  i = 10;  for (; i --; )  printf("%d", i); |  |
| 60 | Все используемые переменные описаны и имеют тип *int*. Что будет выведено в результате работы следующих фрагментов:  i = !0;  while i<5  printf("%i", i ++); |  |

**2.6 Контрольная работа по теме «Массивы данных»**

Вариант 1

1. Значения элементов двухмерного массива A[1..10,1..10] сначала равны 4. Затем выполняется следующий фрагмент программы:

**for (i=0;i<6;i++)**

**for (j=0;j<5;j++)**

**{A[i][j]=A[i][j]+6;**

**A[j][i]=A[j][i]+6;}**

Сколько элементов массива будут равны 10?

1. В программе описан одномерный целочисленный массив с индексами от 0 до 10 и целочисленная переменная **i**. В приведенном ниже фрагменте программы массив сначала заполняется, а потом изменяется:

**for i:=0 to 10 do A[i]:=i;**

**for i:=0 to 9 do begin**

**A[i]:=A[i+1];**

**end;**

Чему будут равны элементы этого массива?

1. В программе описан одномерный целочисленный массив A с индексами от 0 до 10. Ниже представлен фрагмент этой программы, в котором значения элементов массива сначала задаются, а затем меняются.

**for i:=0 to 10 do**

**A[i]:=3\*i;**

**for i:=1 to 10 do**

**A[i]:=A[i] mod 3;**

Чему будут равны элементы этого массива?

1) Все элементы будут равны 3.

2) Все элементы будут равны 1.

3) Все элементы будут равны 0.

4) Все элементы будут равны своим индексам.

1. Написать программу на одном из языков программирования: Считать из текстового файла Beg.txt две последовательности C1,C2, ..., С7 и Р1,P2,..., Р7. Если каждый элемент первой последовательности меньше суммы элементов второй последовательности, найти при каких значениях i максимально значение выражения Сi/(Pi+Ci2 ). результат записать в файл Rez.txt.

Вариант 2

1. Значения элементов двухмерного массива A[1..10,1..10] сначала равны 4. Затем выполняется следующий фрагмент программы:

**for i:=1 to 4 do**

**for j:=1 to 5 do begin**

**A[i,j]:=A[i,j]+4;**

**A[j,i]:=A[j,i]+5;**

**end;**

Сколько элементов массива будут равны 9?

1. В программе описан одномерный целочисленный массив с индексами от 0 до 10 и целочисленные переменные **k**, **i**. В приведенном ниже фрагменте программы массив сначала заполняется, а потом изменяется:

**for(i=0;i<11;i++) A[i]=i;**

**for (i=0;i<5;i++)**

**{ k=A[i];**

**A[i]=A[i+5];**

**A[i+5]=k; }**

Чему будут равны элементы этого массива?

1. В программе обрабатывается двумерный целочисленный массив A [0..n,0..n]. Первый индекс элемента обозначает номер строки, а второй – номер столбца. Дан фрагмент программы:

**for i:=0 to n do begin**

**c:=A[i,n-i];**

**A[i,n-i]:=A[1,i];**

**A[1,i]:=c;**

**end;**

Что меняет этот фрагмент программы?

1) два столбца в таблице

2) строку и столбец в таблице

3) элементы диагонали и строки в таблице

4) элементы диагонали и столбца в таблице

1. Написать программу на одном из языков программирования: Считать из текстового файла Beg txt две последовательности C1, C2, ..., С10 и P1, P2,..., P10. Если исходная последовательность C1,C2, ..., C10 не содержит ни одного элемента, значение которого совпадает со значением какого-либо элемента второй последовательности, задать значения элементам Х1,X2,..., X10 по правилу Xi=max(Ci, Рi). Результат записать в файл Rez.txt.

Вариант 3

1. Значения элементов двухмерного массива A[1..10,1..10] сначала равны 4. Затем выполняется следующий фрагмент программы:

**for i:=1 to 6 do**

**for j:=1 to 5 do begin**

**A[i,j]:=A[i,j]+6;**

**A[j,i]:=A[j,i]+6;**

**end;**

Сколько элементов массива будут равны 10?

1. В программе описан одномерный целочисленный массив с индексами от 0 до 10 и целочисленные переменные **k**, **i**. В приведенном ниже фрагменте программы массив сначала заполняется, а потом изменяется:

**for(i=0;i<11;i++) A[i]=i;**

**k=A[10];**

**for(i=0;i<10;i++)**

**A[i]=A[i+1];**

**A[0]=k;**

Чему будут равны элементы этого массива?

1. Дан фрагмент программы, обрабатывающий двухмерный массив A размером n×n.

**for(i=0;i<n-1;i++)**

**for (j=0;j<n;j++)**

**if A[i][0] < A[j][0]**

**{k=A[i][0];**

**A[i][0]=A[j][0];**

**A[j][0]=k;}**

В этом фрагменте:

1) упорядочивается первая строка массива по убыванию

2) упорядочивается первый столбец массива по убыванию

3) заменяются элементы k-ого столбца таблицы

4) заменяются элементы k-ой строки таблицы

1. Написать программу на одном из языков программирования: Считать из текстового файла Beg.txt две последовательности С1,С2, ..., C10 и P1,P2,…, P10. Если наибольший элемент первой последовательности меньше наименьшего элемента второй последовательности, задать значения элементам Х1,Х2,..., X10 новой последовательности значения С1,C2, ..., C10, а элементам X11,Х12, ..., X20 значения Р1,Р2, ...P10. Результат записать в файл Rez.txt.

Вариант 4

1. Значения элементов двухмерного массива A[1..10,1..10] сначала равны 0. Затем выполняется следующий фрагмент программы:

**for (i=0;i<4;i++)**

**for (j=1;j<6;j++)**

**{A[i][j]=A[i][j]+4;**

**A[j][i]=A[j][i]+5;}**

Сколько элементов массива будут равны 9?

1. В программе описан одномерный целочисленный массив с индексами от 0 до 10 и целочисленные переменные **k**, **i**. В приведенном ниже фрагменте программы массив сначала заполняется, а потом изменяется:

**for i:=0 to 10 do A[i]:=i;**

**for i:=0 to 4 do begin**

**k:=A[2\*i];**

**A[2\*i]:=A[2\*i+1];**

**A[2\*i+1]:=k;**

**end;**

Чему будут равны элементы этого массива?

1. Значения двух массивов A и B с индексами от 1 до 100 задаются при помощи следующего фрагмента программы:

**for i:=1 to n do**

**A[i]:=(i-75)\*(i-75);**

**for i:=1 to n do**

**B[101-i]:=A[i];**

Какой элемент массива B будет наибольшим?

1) B[1] 2) B[26] 3) B[75] 4) B[100]

1. Написать программу на одном из языков программирования: Считать из текстового файла Beg.txt матрицу Р(2,10). Элементы столбцов соответственно абсцисса и ордината одной из десяти точек плоскости. Если нет ни одной пары точек, расстояние между которыми меньше заданной величины R, заменить на нуль в матрице Р все отрицательные абсциссы точек, увеличив ординаты этих точек на R. Результат записать в файл Rez.txt.

ПРИМЕР РЕШЕНИЯ ЗАДАЧИ КОНТРОЛЬНОЙ РАБОТЫ

Считать целочисленный массив A(m) из текстового файла Beg.txt и выполнить сортировку элементов с третьего по пятнадцатый методом обмена. Отсортированную последовательность записать в файл результата Rez.txt . Определить, есть ли в этой последовательности элемент со значением b,заданным с клавиатуры.

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <math.h>

const m=10;

FILE \*file;

int a[m],i,j,k,f=0,n,b;

void vvod()

{

file=fopen("beg.txt","r");

for(i=0;i<m;i++)

fscanf(file,"%d",&a[i]);

fclose(file);

}

void izfaila()

{

file=fopen("rez.txt","w");

for(i=0;i<m;i++)

fscanf(file,"%d",&a[i]);

fclose(file);

}

void sort()

{

for (i=1;i<m;i++)

for (j=0;j<m-i;j++)

{

if (a[j]<a[j+1])

{

k=a[j]; a[j]=a[j+1]; a[j+1]=k;

}

}

}

void poisk()

{

i=0;

j=m;

while ((i<j) && (f!=1))

{

n=(i+j)/2;

if (a[n]==b) f=1;

else if (a[n]<b) i=n+1;

else j=n;

}

}

void main()

{

clrscr();

printf("Числа которые в файле\n");

vvod();

for(i=0;i<m;i++)

printf("%4d",a[i]);

sort();

vvod();

izfaila();

printf("\nМассив после сортировки\n");

for(i=0;i<m-2;i++)

printf("%4d",a[i]);

printf("\nВведите число которое ищете\n");

scanf("%d",&b);

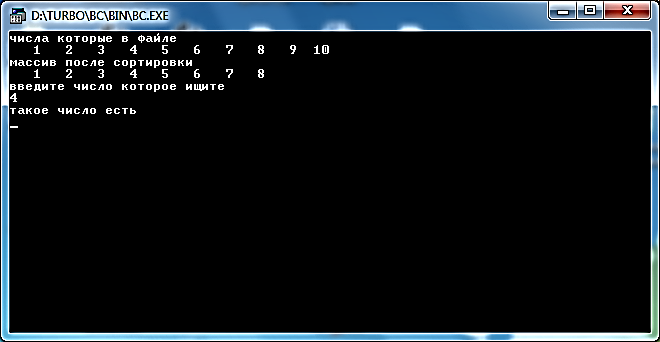
poisk();

if (f==1) printf("Такое число есть\n");

else printf("такого числа нет\n");

getch();

}



**ЗАКЛЮЧЕНИЕ**

Итак, мы завершили трудный путь освоения премудростей структурного программирования на языках Паскаль и Си. Если вы подошли к изучению предмета достаточно серьезно и заинтересованно, разобрали все примеры и решили все предложенные задачи, вы теперь несомненно в состоянии самостоятельно составить алгоритм и написать программу, решающую задачу средней сложности. Вы также знаете, как отладить и выполнить вашу программу и готовы перейти к следующему этапу – объектно-ориентированному программированию.

Мы живем в быстро изменяющемся мире, а программирование является в настоящее время одной из наиболее динамичных областей человеческой деятельности — новые идеи, новые языки, версии, новые системы программирования появляются с устрашающей скоростью. Освоиться в этом потоке информации и остаться на плаву для программиста-профессионала невозможно без настроя на постоянную, ежедневную учебу. Не испытывая удовольствия от учебы, едва ли возможно выдержать это в течение достаточно длительного срока.

Успехов всем!

**РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА**

1. Голицына О.Л., Партыка Т.Л., Попов И.И. Языки программирования. – М: Форум, 2013
2. Демидов Д.В. Основы программирования в примерах на языке Паскаль. – М.: НИЯУ МИФИ, 2014
3. Зубков С.В. Assembler для DOS, Windows и Unix – М: ДМК,2011
4. Культин Н.Б. С/С++ в задачах и примерах. - Спб: БХВ-Петербург, 2013
5. Окулов С.М. Основы программирования - М.: БИНОМ, 2012
6. Павловская Т.А. С/С++. Программирование на языке высокого уровня.: учебник – СПб:Питер, 2014
7. Потопахин В.В. Язык С. Освой на примерах - Спб: БХВ-Петербург, 2011
8. Семакин И.Г., Шестаков А.П. Основы алгоритмизации и программирования. - М:ОИЦ «Академия», 2012
9. Эпштейн М.С. Программирование на языке С:учебник для студ. средн. проф. образования – М:ОИЦ «Академия», 2013
10. Эпштейн М.С. Практикум по программированию на языке С. – М:ОИЦ «Академия», 2012
11. <http://ipg.h1.ru/lessons/ci>
12. <http://www.intuit.ru/studies/courses/975/311/info>
13. <http://www.intuit.ru/studies/courses/564/420/info>
14. <http://www.intuit.ru/studies/courses/16740/1301/info>
15. <http://www.intuit.ru/studies/courses/4453/686/info>
16. <http://window.edu.ru>