9 класс. 1 полугодие

Информационная безопасность в компьютерных сетях

Необходимость в защите информации:

- проведение финансовых операций с использованием Интернета;
- заказ товаров и услуг;
- использование кредитных карточек;
- доступ к закрытым информационным ресурсам;
- передача телефонных разговоров.

Проблемы, возникающие с безопасностью передачи информации:

1. перехват информации (целостность информации сохраняется, но ее конфиденциальность нарушена);

2. модификация информации (исходное сообщение изменяется либо полностью подменяется другим и отсылается адресату);

3. подмена авторства информации;

4. перехват сообщения с его изъятием.

Принципы безопасности:

1. Аутентификация - это процесс распознавания пользователя системы и предоставления ему определенных прав и полномочий. Каждый раз, когда заходит речь о степени или качестве аутентификации, под этим следует понимать степень защищенности системы от посягательств сторонних лиц на эти полномочия.

2. Целостность - состояние данных, при котором они сохраняют свое информационное содержание и однозначность интерпретации в условиях различных воздействий. В частности, в случае передачи данных под целостностью понимается идентичность отправленного и принятого.

3. Секретность - предотвращение несанкционированного доступа к информации. В случае передачи данных под этим термином обычно понимают предотвращение перехвата информации.

Криптография: Для обеспечения секретности применяется шифрование, или криптография, позволяющая трансформировать данные в зашифрованную форму, из которой извлечь исходную информацию можно только при наличии ключа.

В основе шифрования лежат два основных понятия: алгоритм и ключ. Алгоритм - это способ закодировать исходный текст, в результате чего получается зашифрованное послание. Зашифрованное послание может быть интерпретировано только с помощью ключа.

Электронная цифровая подпись (ЭЦП): При помощи электронной подписи получатель может убедиться в том, что полученное им сообщение послано не сторонним лицом, а имеющим определенные права отправителем. Электронные цифровые подписи создаются шифрованием контрольной суммы и дополнительной информации при помощи личного ключа отправителя. Таким образом, кто угодно может расшифровать подпись, используя открытый ключ, но корректно создать подпись может только владелец личного ключа. Для защиты от перехвата и повторного использования подпись включает в себя уникальное число - порядковый номер.

С 2012 года в Казахстане функционирует свой Национальный удостоверяющий центр, в котором граждане и организации Казахстана могут получить свои закрытые

ключи и программные пакеты для формирования своей цифровой подписи для ведения электронных юридических операций с ее использованием.

Сайты, использующие ЭЦП:

- «Электронное правительство»: http://egov.kz/;
- «Электронное лицензирование»: https://elicense.kz/.

Аутентификация: является одним из самых важных компонентов организации защиты информации в сети. Прежде чем пользователю будет предоставлено право получить тот или иной ресурс, необходимо убедиться, что он действительно тот, за кого себя выдает.

При получении запроса на использование ресурса от имени какого-либо пользователя сервер, предоставляющий данный ресурс, передает управление серверу аутентификации. После получения положительного ответа сервера аутентификации пользователю предоставляется запрашиваемый ресурс.

При аутентификации используется, как правило, принцип, получивший название "что он знает", - пользователь знает некоторое секретное слово, которое он посылает серверу аутентификации в ответ на его запрос.

Аутентификация используется при входе в личный кабинет.

Защита сетей: Брандмауэр - это система или комбинация систем, позволяющие разделить сеть на две или более частей и реализовать набор правил, определяющих условия прохождения пакетов из одной части в другую. Как правило, эта граница проводится между локальной сетью предприятия и Интернетом.

Открытие окна свойств и настройки брандмауэра: **Пуск – Панель управления – Сеть и подключения к Интернету – Брандмауэр Windows.**

Брандмауэр пропускает через себя весь трафик и для каждого проходящего пакета принимает решение - пропускать его или отбросить. Для того чтобы брандмауэр мог принимать эти решения, для него определяется набор правил.

Брандмауэр может быть реализован как аппаратными средствами (то есть как отдельное физическое устройство), так и в виде специальной программы, запущенной на компьютере.

Брандмауэр обычно состоит из нескольких различных компонентов, включая фильтры или экраны, которые блокируют передачу части трафика. Все брандмауэры можно разделить на два типа:

- пакетные фильтры, которые осуществляют фильтрацию IP-пакетов средствами фильтрующих маршрутизаторов;
- серверы прикладного уровня, которые блокируют доступ к определенным сервисам в сети.

Законодательство РК в сфере защиты информации

Закон «Об информатизации» от 11 января 2007 года N217.

Автоматизация - применение информационно-коммуникационных технологий для обработки, систематизации, хранения и передачи информации с целью облегчения и вытеснения форм человеческого труда, повышения производительности и прозрачности процессов.

Информатизация - организационный, социально-экономический и научно-технический процесс, направленный на формирование и развитие электронных информационных ресурсов, информационных систем на основе использования информационных технологий.

Информационная услуга - услуга по предоставлению пользователям информационных ресурсов.

Объектами информатизации являются: электронные информационные ресурсы, информационные системы, информационные работы и электронные услуги.

Субъектами информатизации являются: государственные органы, физические и юридические лица, осуществляющие деятельность или вступающие в правоотношения в сфере информатизации на территории Республики Казахстан.

Целями государственного регулирования в сфере информатизации являются развитие информационной инфраструктуры Республики Казахстан, в том числе формирование "электронного правительства", а также информационное обеспечение социального и экономического развития страны.

Государственное регулирование в сфере информатизации основано на следующих **принципах**:

1) неприкосновенность частной жизни граждан и строгое соблюдение конституционных прав и свобод граждан;

2) равенство прав физических и юридических лиц на участие в деятельности в сфере информатизации и использование ее результатов;

3) свободный доступ и обязательное предоставление электронных информационных ресурсов (презумпция открытости), содержащих информацию о деятельности государственных органов, кроме электронных информационных ресурсов, доступ к которым ограничен в соответствии с законодательством Республики Казахстан;

4) свобода поиска, производства и передачи любых электронных информационных ресурсов, доступ к которым не ограничивается в соответствии с законодательством Республики Казахстан;

5) обеспечение безопасности личности, общества и государства при использовании электронных информационных ресурсов и применении информационных технологий;

6) своевременность предоставления, объективность, полнота и достоверность электронных информационных ресурсов, в отношении которых законодательством Республики Казахстан установлен обязательный характер их публичного распространения либо предоставления государственным органам;

7) обеспечение добросовестной конкуренции;

8) обеспечение централизованного управления национальными электронными информационными ресурсами и национальными информационными системами;

9) обеспечение надежности и управляемости сферой информатизации на основе единых стандартов на территории Республики Казахстан;

10) соблюдение технологической нейтральности при установлении требований к применению информационных технологий;

11) ограничение сбора, обработки персональных данных целями, для которых они собираются.

Виды электронных информационных ресурсов.

В зависимости от права собственности электронные информационные ресурсы подразделяются на:

1) государственные;

2) негосударственные.

В зависимости от категории доступа электронные информационные ресурсы подразделяются на:

1) общедоступные;

2) ограниченного доступа.

Электронные информационные ресурсы, создаваемые, приобретаемые и накапливаемые за счет бюджетных средств, а также полученные государственными органами путем иных установленных законами Республики Казахстан способов, являются **государственными**.

Государственные электронные информационные ресурсы, имеющие важное стратегическое значение для экономики и безопасности государства, являются **национальными**.

Электронные информационные ресурсы, создаваемые и приобретаемые за счет средств физических и юридических лиц, а также полученные ими путем иных установленных законами Республики Казахстан способов, являются **негосударственными**.

Электронные информационные ресурсы, которые предоставляются или распространяются их собственником или владельцем без указания условий их использования, а также сведения, доступ к которым является свободным и не зависит от формы их представления и способа распространения, являются **общедоступными**.

Электронные информационные ресурсы, содержащие сведения, доступ к которым ограничен законами Республики Казахстан или их собственником, или владельцем в случаях, установленных законодательством Республики Казахстан, являются электронными информационными ресурсами **ограниченного доступа**.

Современное программное обеспечение и тенденции его развития

Программа – это последовательность инструкций, выполняя которые компьютер совершает определенные действия или решает поставленную задачу по обработке данных.

Программные средства (программное обеспечение, Software) – это набор программ различного предназначения.

Характеристики программ:

- **функциональность программы**, т.е. полнота удовлетворения ею потребностей пользователя;
- наглядный, удобный, интуитивно понятный и привычный пользователю **интерфейс**, т.е. способ взаимодействия программы с пользователем);
- **простота освоения программы** даже начинающими пользователями, для чего используются информативные подсказки, встроенные справочники и подробная документация;
- **надежность программы**, т.е. устойчивость ее к ошибкам пользователя, отказам оборудования и т.д., и разумные ее действия в этих ситуациях.

Виды программного обеспечения:

- **системные программы** (управление работой всех устройств компьютера и его ресурсов по обеспечению обработки данных);
- **прикладные программы** (обработка данных);
- **системы программирования** (обработка других программ).

Системные программы:

1. Операционная система – это комплекс взаимосвязанных системных программ, назначение которого - организовать взаимодействие пользователя с компьютером и выполнение всех других программ.

2. Драйверы – специальные системные программы для управления внешними устройствами компьютера.

3. Оболочки – это программы, созданные для упрощения работы со сложными программными системами, такими, например, как DOS. Они преобразуют неудобный командный пользовательский интерфейс в дружественный графический интерфейс или интерфейс типа "меню". Оболочки предоставляют пользователю удобный доступ к файлам и обширные сервисные услуги (Norton Commander).

4. Операционные оболочки, в отличие от обычных программ-оболочек, не только дают пользователю более наглядные средства для выполнения часто используемых действий, но и предоставляют новые возможности для запускаемых программ (MS-Windows 3.x).

5. Программы вспомогательного назначения, служебные программы, утилиты (лат. utilitas — польза).

Прикладные программы:

табличные процессоры (Lotus -1-2-3, Quattro Pro, Microsoft Excel, Super Calc и др.), системы управления базами данных /СУБД/, графические редакторы, системы деловой и научной графики (Microsoft Chart, Boeing, Graph и др., Harvard Graphics),системы автоматизированного проектирования /САПР/ (Auto Cad), интегрированные системы, бухгалтерские программы.

Системы программирования:

Turbo C, Turbo Pascal, Basic, Borland Delphi, Microsoft Visual Basic, Borland C++.

Служебные программы



Программы вспомогательного назначения, служебные программы, утилиты (лат. utilitas — польза):

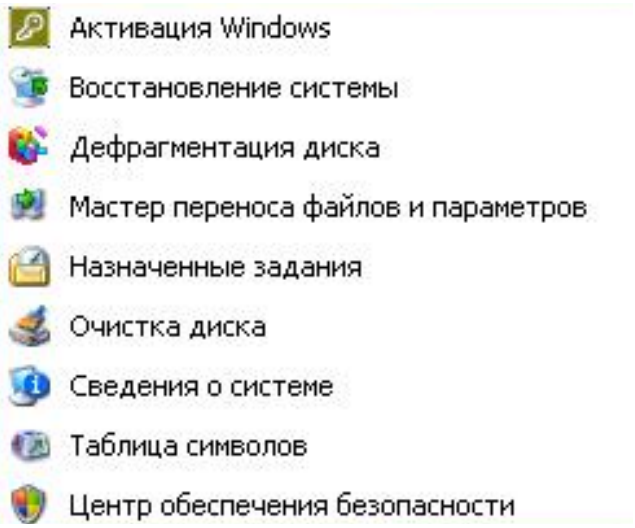
1. расширяют и дополняют соответствующие возможности операционной системы,
2. решают самостоятельные важные задачи.

Вспомогательные программы: программы-упаковщики, программы для создания резервных копий, антивирусные программы, коммуникационные программы,

программы для диагностики компьютера, программы-кэши для диска, программы для оптимизации дисков, программы динамического сжатия дисков, программы для автономной печати (спулеры), программы для управления памятью, программы для печати экрана, программы управления локальной сетью.

Служебное программное обеспечение – это набор программ общего пользования, которые необходимы для настройки и улучшения работы компонентов компьютера.

Утилиты (служебные программы) – это программы, которые служат для выполнения вспомогательной обработки данных, а также расширяют возможности оборудования, ограниченные стандартными функциями. Утилиты расширяют параметры и настройки программ.



Сведения о системе: Пуск – Программы-Стандартные – Служебные- Сведения о системе

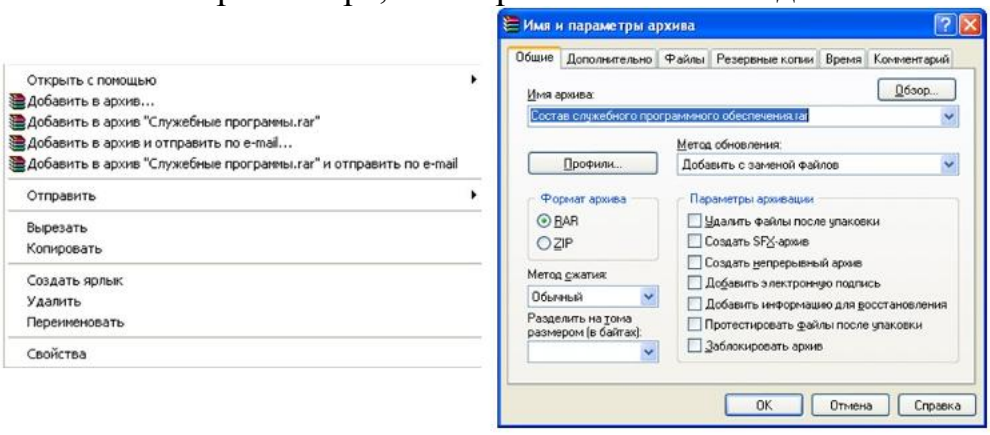
Проверка диска: Пуск – Программы – Стандартные – Служебные – Проверка диска.

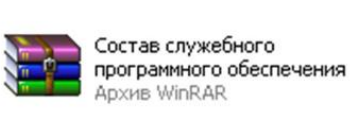
Служебное программное обеспечение предназначено для выполнения определенных функций:

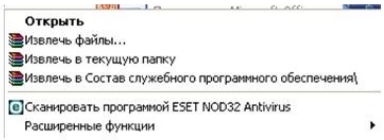
- ввод-вывод информации
- планирование и управление задачами
- проверка работоспособности устройств компьютера
- вывод справочной информации
- перемещение (копирование) файлов, папок, документов
- диагностика и профилактика компьютера

Архиваторы – программы, необходимые для сжатия одного или нескольких файлов в архив или несколько архивов. В архиве содержится основная информация о файлах – имя и размер, для того, чтобы при распаковке было понятно, что это за файл или папка.

Архив – это файл, содержащий в себе один или несколько файлов в сжатом (упакованном виде). Имена файлов архивов могут иметь расширения .zip, arj, rar, arc и др., в зависимости от типа архиватора, в котором они были созданы.







Антивирусные программы – программы, необходимые для защиты системы компьютера от вирусов. Антивирусные программы являются необходимыми при наличии имеющегося интернет – подключения. Они блокируют нежелательные, вредные или подозрительные ссылки, содержащие вирус, и отправляют их в карантин.

Программы технического обслуживания и диагностики представляют собой совокупность программ для аппаратной части компьютера. Системы технического обслуживания необходимы для проверки работы компьютера в целом, а также его отдельных частей. Являются незаменимым инструментом для технического ремонта ПК.

Инсталляция программ – это установка новых программ на ваш компьютер.

Этапы инсталляции:

1. Выбрать *Пуск – Настройка – Панель управления – Установка и удаление программ;*
2. Нажать кнопку *Установить;*
3. Следовать указаниям *Мастера установки.*

В процессе инсталляции обратите внимание на:

- Условия лицензионного соглашения;
- Наличие необходимого свободного места на диске;
- Каталог, в который будет установлен программный продукт;
- Выбор устанавливаемых компонентов.

Файлы и их обработка (2 часа)

Известно, что файлы используют для долговременного хранения информации на внешних запоминающих устройствах и доступа к этой информации.

В языке Pascal под **файлом** понимается область памяти на диске, способная хранить некоторое количество однородной информации. В эту область внешней памяти можно как поместить данные (ввод), так и извлечь из нее (вывод).

Особенности файлов в Pascal:

- число компонент файла не фиксируется и не ограничивается;
- доступ к компонентам файла возможен только путем только их последовательного перебора от начала файла;
- компоненты файла создаются только процедурой вывода и доступны только процедуре ввода.

Иными словами, **файл** - это список неопределенного количества данных одного типа, расположенный на некотором внешнем запоминающем устройстве.

Классифицируют файлы **по методу доступа**:

- последовательный (чтобы считать последние даны, необходимо сначала прочитать все предыдущие);
- прямой (обращение к любому элементу, независимо от его расположения в файле);

и **по типу (логической структуре)**:

- типизированный;
- нетипизированный;
- текстовый.

Файловый тип определяется так:

FILE OF тип_элементов

Например:

```
Var
F1: File Of Char; {типизированный текстовый файл, т.е. файл, состоящий из символов}
MyDocument: Text; {текстовый файл}
DataBase: File Of Student; {файл из записей типа Student}
```

С понятием файла связано также понятие указателя на текущий элемент файла. Все элементы файла считаются пронумерованными; начальный элемент имеет номер 0.

Общая схема чтения из файла:

```
reset (f1);
…
read (f1, <имя переменной>);
…
close (f1);
```

Общая схема записи в файл:

```
rewrite (f2);
…
write (f2, <имя переменной>);
…
close (f2);
```

С переменными файлового типа применимы следующие стандартные процедуры и функции:

Assign (FileName, DosFullName) - связывает файловую переменную FileName с конкретным файлом DOS, полное имя которого DosFullName;

Reset (FileName) - открыть файл на чтение, установить указатель на первый элемент в файле; если файла не существует, то возвращается ошибка;

Rewrite (FileName) - открыть файл на запись, установить указатель на первый элемент в файле; если файла не существует - открыть новый файл;

Close (FileName) - завершает действия с файлом и разрывает связь с файловой переменной FileName;

Read (FileName, список) - чтение значений начиная с текущей позиции указателя в список ввода; после каждого чтения указатель смещается на следующую позицию, при достижении конца файла возвращает ошибку;

Write (FileName, список) - запись списка выражений в файл; продвижение указателя на количество элементов списка;

Seek (FileName, Number) - явная установка указателя на элемент файла с номером Number;

Truncate (FileName) - отсечение от файла его хвостовой части, начинающейся с текущей позиции указателя включительно;

FileSize (FileName) - возвращает число элементов файла;

FilePos (FileName) - возвращает номер элемента, на который установлен текущий указатель;

EoF (FileName) - возвращает логическое значение True или False, говрящее о достижении конца файла.

Структурированные данные (2 часа)

Паскаль позволяет работать с более сложными по своей конструкции типами данных. Они называются типами данных пользователя и относятся к **структурированным типам** (от слова структура). Отличаются они тем, что переменные структурного типа состоят не из одного элемента (как все стандартные типы — целые, вещественные, символьные и логические), а из нескольких элементов с одним общим именем.

Можно сказать, что **структурные типы данных** определяют некоторый способ образования новых типов из уже имеющихся. Таким образом, **Паскаль** допускает образование структур данных произвольной сложности, позволяя тем самым достичь адекватного представления в программе тех данных, с которыми вы оперируете.

Существует несколько методов структурирования, каждый из которых отличается способом обращения к отдельным компонентам, и способом обозначения компонентов, входящим в структурные данные. По способу организации и типу компонентов в сложных типах данных выделяют следующие разновидности:

- регулярный тип (массивы);
- комбинированный тип (записи);
- файловый тип (файлы);
- множественный тип (множества);
- строковый тип (строки).

Все структурированные типы данных требуют отдельного рассмотрения и будут подробно изучены нами в дальнейшем, а сегодня мы только их определим.

Массив (array). Он представляет собой заранее известное количество однотипных элементов, снабженных индексами. Массив может быть одномерным или многомерным.

Запись (record). Она включает в себя несколько полей, тип которых может отличаться друг от друга. Например, товар на складе описывается следующими величинами: наименование, количество, цена, наличие сертификата качества и т.д. В этом примере наименование – величина типа string, количество – integer, цена – real, наличие сертификата – boolean.

Запись представляет собой наиболее общий и гибкий структурированный тип данных, так как она может быть образована из неоднотипных компонентов и в ней явным образом выражена связь между элементами данных, характеризующими реальный объект.

Строка (string) – последовательность символов кодовой таблицы персонального компьютера. Количество символов в строке может изменяться от 0 до 255.

Множество (set) - это набор взаимосвязанных по какому-либо признаку или группе признаков элементов. Каждый элемент во множестве называется ***элементом множества***. Множество должно состоять из порядковых элементов, и их число не должно превышать 255.

Файл (file) - последовательность однотипных компонентов, записанных на внешнем носителе под определенным именем. Тип этих компонентов может быть любой, за исключением типа - файла. Размер файла не объявляется.

Вложенные циклы (2 часа)

При оформлении вложенных циклов обязательно учитывать **2 условия**:

1. внутренний цикл должен полностью входить во внешний;

2. нельзя использовать одну и ту же переменную в качестве параметра внутреннего и внешнего циклов.

Вложенные циклы используются при составлении программ работы с массивами, создания рисунков (графических файлов).

Работа программы с вложенными циклами (при создании графических файлов) останавливается оператором **until keypressed**.

Пример:

```
for i:=1 to 4 do
for j:=1 to 4 do
     begin
       write('Введите элемент ',i,' ',j,' ');
       readln matr[i,j];
     end;
```

Понятие о массивах (2 часа)

1. ПОНЯТИЕ МАССИВА.

В практической деятельности человека часто используются таблицы и списки.

Например:

- Список учеников в журнале;
- Список среднесуточной температуры месяца;
- Таблица умножения.

Элементы списков и номера строк и столбцов всегда нумеруются.

В языках программирования подобные типы данных описываются с помощью массивов.

Массив - это набор однотипных данных, к которым можно обратиться с помощью единственного имени.

Каждое значение в массиве называется ***элементом.***

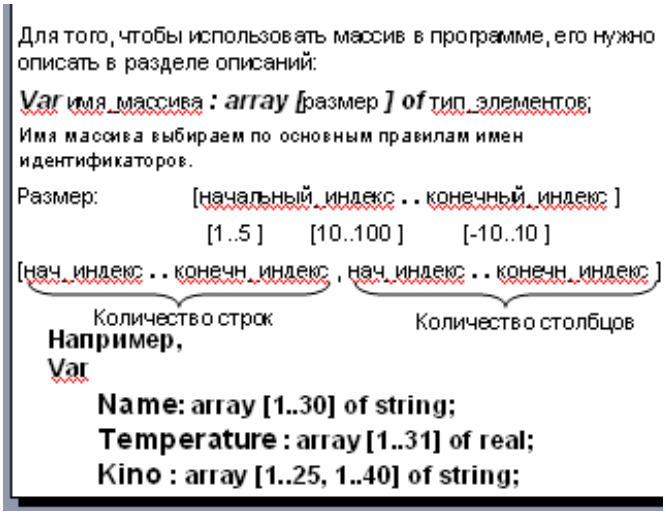
Номер элемента в списке называется ***индексом.***

Элементы представляют собой символьные или числовые значения, которые можно использовать в выражениях ЯП ПАСКАЛЬ





2. ОПИСАНИЕ МАССИВА.





Первый способ:

Const MaxM = ...; {Максимальные значения количества элементов}
Type <имя типа> = Array[1..MaxM] **Of <тип элементов>;**
Var A1, B2, C3: <имя типа> ;
Например:
Const MaxM = 10;
Type A = Array[1..MaxM] **Of integer;**
Var M: A;

Второй способ:

Const MaxM = ...; {Максимальные значения количества элементов}
Var <имя массива>:Array[1..MaxN] **Of <тип элементов>;**
Например:
Const MaxM = 10;
Var A: Array[1..MaxM] **Of integer;**

3. ЭТАПЫ РАБОТЫ С МАССИВАМИ.

А) ВВОД ДАННЫХ

Способы ввода данных в массив:

- с клавиатуры

```
Program vvod;
  Const  maxM=10;
   Var tab1: array[1..maxM] of integer;
  i: integer;
Begin
  For i:=1 to maxM do
        begin
           writeln('Введите элементы массива ');
            readln(tab[i]);
      end;
End.
```

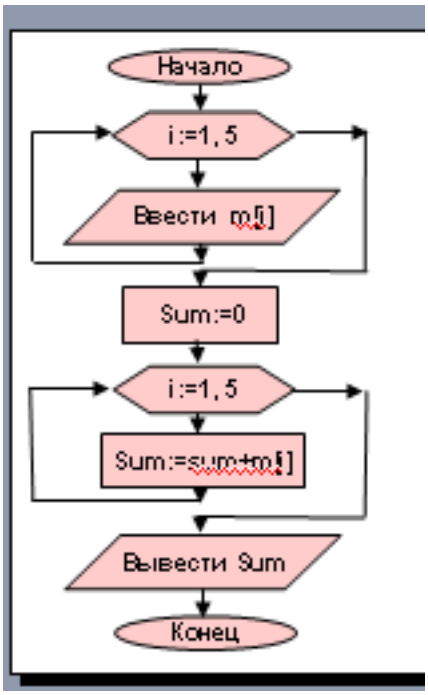
- с помощью генератора случайных чисел;
- с помощью вычисления элементов.

Б) ПРЕОБРАЗОВАНИЕ МАССИВА (ДЕЙСТВИЯ НАД ЭЛЕМЕНТАМИ МАССИВА)

Рассмотрим работу с массивом на примерах:

Задача 1.

Написать программу, которая сначала должна обеспечить ввод 5 чисел в одномерный массив, затем вычислить их сумму и вывести ее на экран.




```
Program Sum_mas;
uses Crt;
Var i: integer;
    Sum : real;
    M : array [1..5] of Rea
begin
Clrscr;
For i:=1 to 5 do
    begin
        write ('m[',i,']=');
        readln(m[i]);
    end;
sum:=0;
For i:=1 to 5 do
    Sum:=sum+m[i];
Writeln ('Sum=',sum:6:2);
readln;
end.
```

В) ВЫВОД МАССИВА

```
Program vivod;
  Const  maxM=10;
     Var tab1: array[1..maxM] of
integer;
       i: integer;
Begin
  For i:=1 to maxM do
      begin
          writeln('Вывод элементов
массива');
          writeln (tab[i]);
      end;
End.
```

Итоги:

1. Массив – это структура данных, представляющая собой совокупность элементов одного типа.
2. Массив должен быть объявлен в разделе объявления переменных.
3. Доступ к элементу массива осуществляется путем указания имени массива и индекса (номера) элемента.
4. Для ввода, вывода и обработки массивов удобно применять команды цикла.

ДВУМЕРНЫЙ МАССИВ

Двумерный массив можно представить себе в виде таблицы, в которой все строки и столбцы пронумерованы.

Каждый элемент такого массива имеет два индекса:

Первый индекс – это номер строки;

Второй индекс – номер столбца.

A[1,1]	A[1,2]	A[1,3]	A[1,4]	A[1,5]
A[2,1]	A[2,2]	A[2,3]	A[2,4]	A[2,5]
A[3,1]	A[3,2]	A[3,3]	A[3,4]	A[3,5]
A[4,1]	A[4,2]	A[4,3]	A[4,4]	A[4,5]

Описание двумерных массивов:

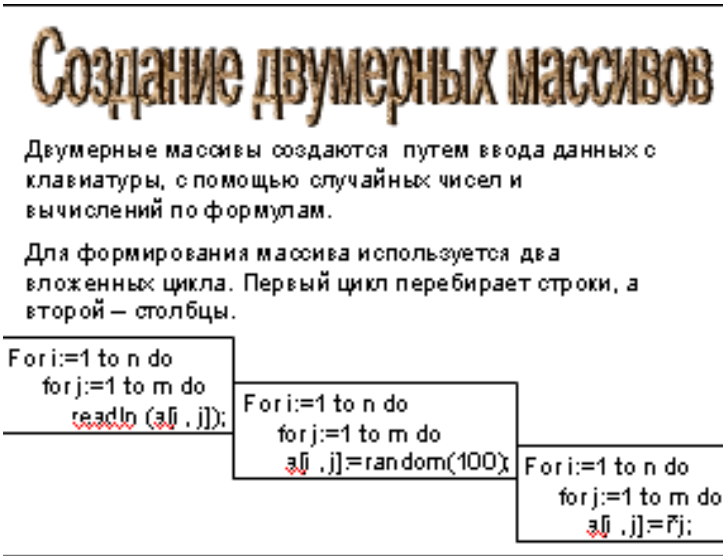
```
Const n=4;
      m=5;
Var  A :array [1..n, 1..m] of integer;
```

Строки Столбцы

A [2,4]

A [4,2]

A[1,1]	A[1,2]	A[1,3]	A[1,4]	A[1,5]
A[2,1]	A[2,2]	A[2,3]	A[2,4]	A[2,5]
A[3,1]	A[3,2]	A[3,3]	A[3,4]	A[3,5]
A[4,1]	A[4,2]	A[4,3]	A[4,4]	A[4,5]



Ввод элементов массива

```
Program vvod;
  Var tab1: array[1..10, 1..10] of integer;
  I, j: integer;
Begin
  For i:=1 to 10 do
     for j:=1 to 10 do
     begin
       writeln('Введите элемент массива ');
       readln (a[i , j]);
     end;
End.
```

Вывод массивов в виде таблицы

```
For i:=1 to n do
     begin
       for j:=1 to m do
       write (a[i , j]);
       writeln;
     end;
```

Составление программ с использованием двумерных массивов

Задача 1. Создать программу для формирования таблицы умножения (таблицы Пифагора).

```
Program Piphagor;
 Uses crt;
 var i,j:byte;
 a: array [1..9, 1..9] of integer;
begin
 clrscr;
 {шапка таблицы Пифагора}
 for i:=1 to 9 do write(i:3,' |'); writeln;
 for i:=1 to 9 do write('----+'); writeln;      {сама таблица}
  for i:=2 to 9 do
   begin
    for j:=1 to 9 do
      begin
        a[i,j]:=i*j;
        write(a[i,j]:3,' |');
      end;
    writeln;
   end;
 readln;
end.
```

Стандартные алгоритмы обработки массивов (3 часа)

Работа с **одномерными массивами**:

1. Добавить к каждому элементу массива число 25.

```
For i:=1 To n Do A[i]:=A[i]+25;
```

2. Если элемент четный, то добавить к нему первый, а иначе последний элемент массива.

```
For i:=1 To n Do
   If  A[i] mod 2 =0
    then A[i]:=A[i] + A[1]
     else A[i]:=A[i] +A[n];
```

3. Получить третий массив как сумму соответствующих элементов 2 заданных.

```
   For i:=1 To n Do
   C[i]:=A[i]+B[i];
```

4. Получить элементы третьего массива как максимальные из 2 заданных массивов.

```
   For i:=1 To n Do
   If A[i]>B[i] then C[i]:=A[i] else C[i]:=B[i];
```

5. Найти значение максимального по модулю элемента массива.

```
Maks:=Abs(A[1]);
     For i:=2 To n Do
If Abs(A[i])> Maks then Maks:=Abs(A[i])
```

6. Найти среднее арифметическое положительных элементов массива.

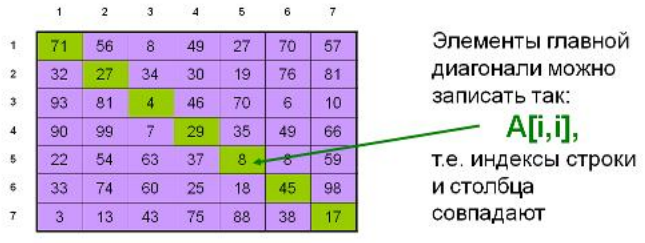
```
s:=0;  k:=0
     For i:=1 To n Do
If  A[i]>0 then
 begin
   s:=s+(A[i]); k:=k+1;
 end;
Average:=s/k
```

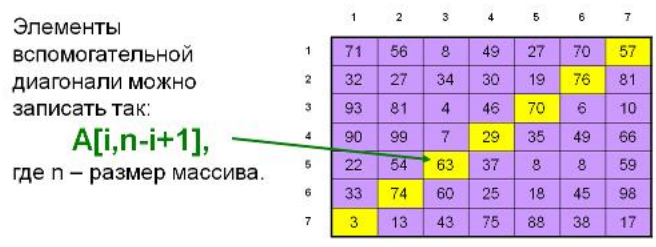
7. Изменить знак у максимального по модулю элемента массива.

```
maks:= 1;
     For i:=2 To n Do
If Abs(A[i])>Abs(A[maks]) then maks:=i;
A[maks]:=-A[maks];
```

Работа с **двумерными массивами**:

«Квадратные» массивы - это такие массивы, в которых количество столбцов равно количеству строк.





1. Заполнение нулями главной диагонали.

```
 for i:=1 to n do
      a[i,i]:=0;
```

2. Особое значение при работе с квадратными массивами занимает задача **«транспонирования»**, т.е переворачивания массива относительно главной диагонали.

Для выполнения этой задачи нужно поменять местами элементы **A[i,j] и A[j,i]**

Для перестановки двух элементов в массиве необходимо воспользоваться вспомогательной переменной:

```
Temp:=A[2,5]
A[2,5]:=A[5,2];
A[5,2]:=Temp;
```

Для переворачивания всего массива используется двойной цикл.

3. Заполнение нулями элементов выше вспомогательной диагонали.

```
for i:=1 to n do
     for j:=1 to n-i do
      a[i,j]:=0;
```

4. Заполнение нулями элементов по периметру массива.

```
 for i:=1 to n do
      begin
        a[i,1]:=0;
        a[i,n]:=0;
        a[1,i]:=0;
        a[n,i]:=0;
      end;
```

5. В большинстве случаев обработка двумерных массивов аналогична обработке одномерных массивов. Разница состоит лишь в том, что при обработке двумерных массивов используются вложенные циклы:

Если элемент четный, то добавить к нему первый, а иначе последний элемент массива.

```
For i:=1 To 10 Do
     For j:=1 To 10 Do
   If  A[i, j] mod 2 =0 then A[i, j]:=A[i, j] + A[1,1] else A[i, j]:=A[i, j] +A[10,10];
```

Пример программы:

```
Program zadacha1;
  Var tab1: array[1..10, 1..10] of integer;
       i, j: integer;
Begin
  For i:=1 to 10 do
    for j:=1 to 10 do
    begin
      writeln('Введите элемент массива ');
      readln (a[i , j]);
    end;
For i:=1 To 10 Do
     For j:=1 To 10 Do
   If  A[i, j] mod 2 =0 then A[i, j]:=A[i, j] + A[1,1] else A[i, j]:=A[i, j] +A[10,10];
For i:=1 to 10 do
    begin
      for j:=1 to 10 do
      write (a[i , j]);
      writeln;
    end;
end.
```