

Активизация познавательной деятельности учащихся на уроках информатики

"...никакая внешне предлагаемая информация не может быть перенесена внутрь его, если у школьника нет соответствующей мотивации и личностно значимых образовательных процессов".

А.В.Хуторский

Степанова М.М., учитель информатики МБОУ «Гимназия №52» Приволжского района г. Казани

Изменения, происходящие сегодня в современном обществе, в значительной степени определяют особенности и необходимость внесения изменений в деятельность педагога. Традиционные формы работы не всегда доказывают свою эффективность. Хочется, чтобы каждый урок был особенным, запоминающимся. Поддерживанию и развитию познавательного интереса способствует создание новизны на уроке, как в области содержания материала, так и в методах. Дети XXI века не могут просто получать знания. Им нужно научиться добывать информацию и применять ее в повседневной жизни; они хотят легко ориентироваться в постоянно меняющихся условиях.

В обучении информатике на уроках необходимо создавать атмосферу, помогающую школьнику как можно более раскрыть свои способности. Сочетание нескольких технологий, применяемых учителем на уроке, позволяет сделать каждый урок привлекательным и неповторимым. Использование элементов развивающего обучения существенно повышает уровень знаний по информатике, познавательную активность учащихся.

Удивление, желание узнать больше об изучаемом объекте, поделиться своими знаниями – характерные показатели познавательного интереса. И здесь много зависит от эмоционального настроя учителя, его умения импровизировать.

Человек есть творец. Природа наградила человека способностью открывать новое и позаботиться о богатстве чувств, возникающих при творческом озарении. Способность человека делать открытия – не случайное качество, а мощное генетически заложенное средство развития. Задача учителя создать творческую атмосферу, помочь учащимся самореализоваться. На своих уроках я использую различные творческие задания:

- в графическом редакторе Paint «создать» мозаику;
- создать визитную карточку;
- оформить титульный лист книги;
- создать БД «Ученик», содержащую информацию о ваших одноклассниках;
- придумать стихотворение (или рассказ), реализующее заданную алгоритмическую конструкцию;
- создать кроссворд;
- создать презентацию на заданную тему и т.д.

Вопросы активизации учения учащихся относятся к числу наиболее актуальных проблем современной педагогической науки и практики. Реализация принципа активности в обучении имеет определенное значение, т.к. обучение и развитие носят

деятельностный характер, и от качества учения как деятельности зависит результат обучения, развития и воспитания учащихся.

Ключевой проблемой в решении задачи повышения эффективности и качества учебного процесса является активизация учения учащихся. Ее особая значимость состоит в том, что учение, являясь отражательно преобразующей деятельностью, направлено не только на восприятие учебного материала, но и формирование отношения учащегося к самой познавательной деятельности. Преобразующий характер деятельности всегда связан с активностью субъекта. Знания, полученные в готовом виде, как правило, вызывают затруднения учащихся в их применении к объяснению наблюдаемых явлений и решению конкретных задач.

Актуальность данной темы состоит в том, что активные методы обучения позволяют использовать все уровни усвоения знаний: от воспроизводящей деятельности через преобразующую к главной цели – творческо-поисковой деятельности. Творческо-поисковая деятельность оказывается более эффективной, если ей предшествует воспроизводящая и преобразующая деятельность, в ходе которой учащиеся усваивают приемы учения.

Необходимость активного обучения заключается в том, что с помощью его форм, методов можно достаточно эффективно решать целый ряд задач, которые трудно достигаются в традиционном обучении:

- формировать не только познавательные, но и профессиональные мотивы и интересы, воспитывать системное мышление;
- учить коллективной мыслительной и практической работе, формировать социальные умения и навыки взаимодействия и общения, индивидуального и совместного принятия решения, воспитывать ответственное отношение к делу, социальным ценностям и установкам как коллектива, так и общества в целом.

За последние несколько лет изменились мотивы изучения предмета. Мотивом для изучения информатики, конечно, в первую очередь выступает интерес к компьютеру. Однако с каждым днем для большинства детей компьютер становится, фактически, бытовым прибором, а вместе с ним теряет и мотивационную силу. Появление очень большого количества программных продуктов снизило стремление учащихся к теоретической информатике. Учитывая, что мотивы учащихся формируются через их потребности и интересы (Потребность Интерес Мотив), все усилия учитель должен направить на развитие познавательных интересов учащихся.

Познавательный интерес выступает перед нами и как сильное средство обучения. Когда ребенок занимается из-под палки, он доставляет учителю массу хлопот и огорчений, когда же дети занимаются с охотой, то дело идет совсем по-другому. Активизация познавательной деятельности ученика без развития его познавательного интереса не только трудна, но практически и невозможна. Вот почему в процессе обучения необходимо систематически возбуждать, развивать и укреплять познавательный интерес учащихся и как важный мотив учения, и как стойкую черту личности, и как мощное средство воспитывающего обучения, повышения его качества.

Работая над методической темой «Активизация познавательной деятельности учащихся на уроках информатики», пришла к некоторым выводам:

1. Необходимо обратить особое внимание именно на познавательную деятельность учащихся, т.к. активизация деятельности учеников на уроках информатики не представляет особого труда. Эта активность связана в основном с восприятием учащимися компьютера только как средства развлечения. И, соответственно, изучение компьютера, как вычислительного средства, инструмента для

поиска, обработки, передачи информации, т.е. как важнейшего орудия для осуществления информационных процессов, наконец, изучение устройства и принципов работы ЭВМ отходит у большинства обучающихся на второй план.

2. Активизируя познавательную деятельность учащихся средствами информатики (а точнее – информационных технологий), реализуя межпредметные связи в сочетании с современными мультимедийными возможностями и всем известной значимости урока информатики для школьников можно найти массу методов, приёмов и средств такой активизации.

Большинство детей, наверное, приходит на информатику с основной целью – развлечься посредством возможностей компьютерной техники. Средствами развлечения могут выступать игры, видео, музыка, изображения, то есть всё то, что привлекает визуально, позволяет интерактивно участвовать. Всё это называют одним словом – «мультимедиа». Поэтому Для активации познавательной деятельности обучающихся на уроке информатики необходимо, прежде всего, предоставить учебный материал в наиболее мультимедийном и интерактивном виде. Такой материал может быть представлен в виде:

- презентаций (с их помощью можно иллюстрировать материал, а можно предоставить учащимся возможность самостоятельно изучать, что более значимо);
- компьютерных игр (естественно, тех игр, которые содержат развивающий или познавательный материал);
- интерактивных программ, тестов (чем больше участия принимает ученик в процессе обучения, тем больше значимости обретают полученные знания, умения и навыки);
- графических демонстрационных материалов (это могут быть как обычные плакаты, стенды, раздаточные материалы, а лучше, если это будут изображения, которые школьник сам найдёт и просмотрит на ПК);
- видео или мультипликационных фильмов.

Лучше один раз увидеть, чем семь раз услышать – это понятно. Но ещё важнее – хотя бы один раз сделать. Тогда помимо знаний появляется умение. А если сделать несколько раз, развивается навык. Поэтому на уроках информатики должна иметь приоритет именно практическая направленность деятельности учащихся, через которую и происходит познание.

Другим средством для решения данной задачи может оказать помощь интеграция разных предметов при объяснении материала. Так при рассмотрении темы «Строки в Паскале», я обращаюсь к русскому языку. С этой целью **мою разработано методическое пособие**, которое было **апробировано** на уроках информатики на базе МБОУ «Гимназия №52» Приволжского района г. Казани получена **рецензия доктора педагогических наук, профессора, заведующего кафедрой института психологии и образования К(П)ФУ, действительного члена МПА и АПН В.Ф. Габдулхаков в 2015 году.** (Приложение)

Крайне важно чтобы практическая деятельность несла развивающий характер, поэтому в ней должно быть минимум инструкций, максимум самостоятельной исследовательской, поисковой, аналитической деятельности. Для активизации познавательной деятельности при изучении сложного или «скучного» материала, каким часто бывает программирование, порекомендую с самого начала продемонстрировать удивительные результаты, которые может предоставить тот или иной изучаемый материал. Например, можно показать работу небольших программ, созданных с помощью изучаемого языка программирования, выполняющих потрясающие действия:

небольшой конструктор, позволяющий собрать домик, снеговика и другие картинки, воспроизводящих мелодию и др. После такой демонстрации у части школьников возникает желание самим создать не только что-то подобное, а во много раз лучшее произведение программистского искусства.

Чаще всего познавательный интерес является доминирующим и при всех обстоятельствах имеет большую личную значимость для ученика. А раз так, то учителю очень важно не только его распознать, но и управлять им.

Рецензия

на методическое пособие
«Интеграция в обучении на уроках русского языка и информатики»,
составленное учителем информатики

МБОУ «Гимназия №52» Степановой Марией Михайловной

Методическое пособие предназначено для учащихся 9-х классов. Сегодня «метапредметное обучение» приобретает особую популярность. Это вполне объяснимо, ведь метапредметный подход заложен в основу новых стандартов. Программа курса ставит своей целью связать такие предметы, как русский язык и информатика.

Пособие представляет собой вводный курс по программированию, где все вопросы рассматриваются исключительно с точки зрения орфографии или преобразования слов.

Курс состоит из следующих разделов:

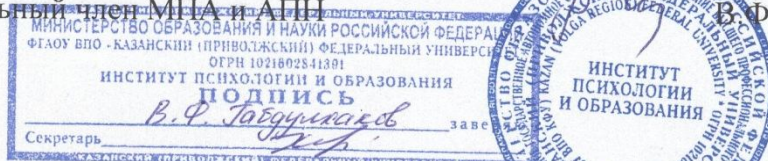
- строки,
- линейный алгоритм,
- разветвляющиеся алгоритмы,
- циклические алгоритмы

В каждом разделе приведены примеры заданий из русского языка, а также задания для самостоятельного изучения.

Актуальность данного курса обусловлена тем, что при изучении материала формируется межкультурные и межотраслевые знания, умения, способности, необходимые для адаптации и продуктивной деятельности в различных профессиональных сообществах.

Доктор педагогических наук,
профессор, заведующий кафедрой
института психологии и образования К (П)ФУ,
действительный член МПА и АПН

В.Ф.Габдулхаков



Муниципальное бюджетное общеобразовательное учреждение
«Гимназия №52» Приволжского района г. Казани

Интеграция в обучении на уроках
русского языка и информатики

Составитель: Степанова М.М.

Казань – 2015г.

Пояснительная записка

Предисловие

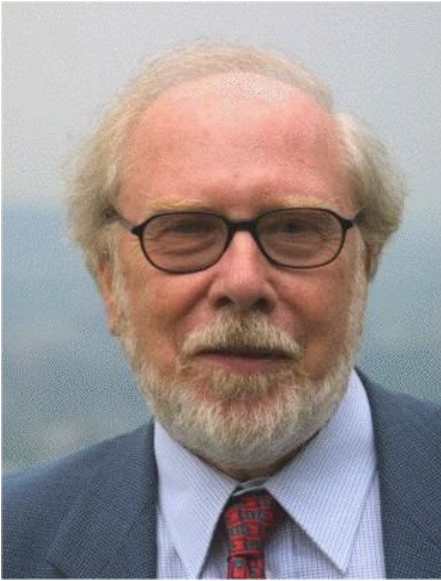
Считается, что программирование является приложением в большей степени физико-математических наук. Так в большинстве своем на уроках информатики разбираются задачи типа «Найти корень уравнения» или «Установить взаимосвязь между силой тока и сопротивлением». Но существуют ряд задач, которые способствуют развитию у учащихся не вычислительных навыков, а на «Проверка орфографии» или «Как получить из одного слова другого». Именно о таких заданиях пойдет речь в этом сборнике.

Пособие представляет собой вводный курс по программированию и рассчитан для учащихся 9-х классов. В данном пособии последовательно излагаются вопросы, касающиеся основ программирования на языке Паскаль. Программы на языке Pascal отличаются строгой структурой. Программирование на нем приучает к аккуратности, продуманности. Кажущаяся ненужной строгость в описании типов данных, процедур и функции оборачивается изящными и понятными конструкциями, а также высокой производительностью программ и экономным использованием памяти.

Пособие содержит условия задач и возможный вариант решения на языке. Также в данном пособии представлены ряд задач для самостоятельного выполнения. Все задачи классифицированы по основным разделам: линейные алгоритмы, разветвляющиеся алгоритмы и циклические алгоритмы..

Среда программирования

Язык назван в честь выдающегося французского математика, физика, литератора и философа Блеза Паскаля, который создал первую в мире механическую машину, складывающую два числа.



Язык Паскаль был разработан Никлаусом Виртом в 1970 г. как язык со строгой типизацией и интуитивно понятным синтаксисом. В 80-е годы наиболее известной реализацией стал компилятор Turbo Pascal фирмы Borland, в 90-е ему на смену пришла среда программирования Delphi, которая стала одной из лучших сред для быстрого создания приложений под Windows. Delphi ввела в язык Паскаль ряд удачных объектно-ориентированных расширений, обновленный язык получил название Object Pascal. Из альтернативных реализаций Object Pascal следует отметить многоплатформенный open source компилятор Free Pascal

Преимущества PascalABC.NET

Язык **PascalABC.NET** позволяет использовать большинство средств, предоставляемых платформой .NET: единая система типов, классы, интерфейсы, исключения, делегаты, перегрузка операций, обобщенные типы (generics), методы расширения, обширные.NET-библиотеки. Добавлен ряд языковых конструкций: описание метода в теле класса, множества произвольных типов, операторы **foreach** и **lock**, внутриблочные переменные. С другой стороны, **PascalABC.NET** имеет структуру языка Delphi (Object Pascal): внешние процедуры и функции, модули.

Алфавит языка

Алфавит - это совокупность допустимых в языке символов. Алфавит включает следующий набор основных символов:

- строчные и прописные латинские буквы:
 - ✓ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
 - ✓ a b c d e f g h i j k l m n o p q r s t u v w x y z
- пробел
- подчеркивание: _
- арабские цифры: 0 1 2 3 4 5 6 7 8 9
- знаки операций: + - * / = < > <= >= := @
- ограничители: . , ' () [] (. .) { } (* *) .. : ;
- спецификаторы: ^ # \$

Структура программы

Программа на языке **PascalABC.NET** имеет следующий вид:

```
program имя программы;  
    раздел uses  
    раздел описаний  
begin  
    операторы  
end.
```

Первая строка называется **заголовком программы** и не является обязательной.

Раздел **uses** начинается с ключевого слова **uses**, за которым следует список имен модулей и пространств имен .NET, перечисляемых через запятую.

Раздел описаний может включать разделы описания переменных, констант, меток, типов, процедур и функций, которые следуют друг за другом в произвольном порядке.

Далее следует блок **begin/end**, внутри которого находятся операторы, отделяемые один от другого символом "точка с запятой".

Раздел **uses** и раздел описаний могут отсутствовать.

Идентификаторы

Идентификаторы служат в качестве имен программ, модулей, процедур, функций, типов, переменных и констант. Идентификатором считается любая последовательность латинских букв или цифр, начинающаяся с буквы. Буквой считается также символ подчеркивания "_".

Например, a1, _h, b123 - идентификаторы, а 1a, ф2 - нет.

Описание переменных

- Переменные могут быть описаны в разделе описаний, а также непосредственно внутри любого блока **begin/end**.
- Раздел описания переменных начинается со служебного слова **var**, после которого следуют элементы описания вида
- *список имен: тип;*
- или
- *имя: тип := выражение;*
- или
- *имя := выражение;*

Имена в списке перечисляются через запятую.

Пример описания переменных

- **var**
a,b,c: integer;
d: real := 3.7;
s := 'Pascal forever';
al := new ArrayList;
p1 := 1;

Описание констант

- Раздел описания именованных констант начинается со служебного слова **const**, после которого следуют элементы описания вида
- *имя константы = значение;*
- или
- *имя константы : тип = значение;*

Пример описания констант

```
const  
Pi = 3.14;  
Count = 10;  
Name = 'Mike';  
DigitsSet = ['0'..'9'];  
Arr: array [1..5] of integer = (1,3,5,7,9);
```

Rec: **record** name: **string**; age: integer **end** = (name: 'Иванов'; age: 23);
Arr2: **array** [1..2,1..2] **of** real = ((1,2),(3,4));

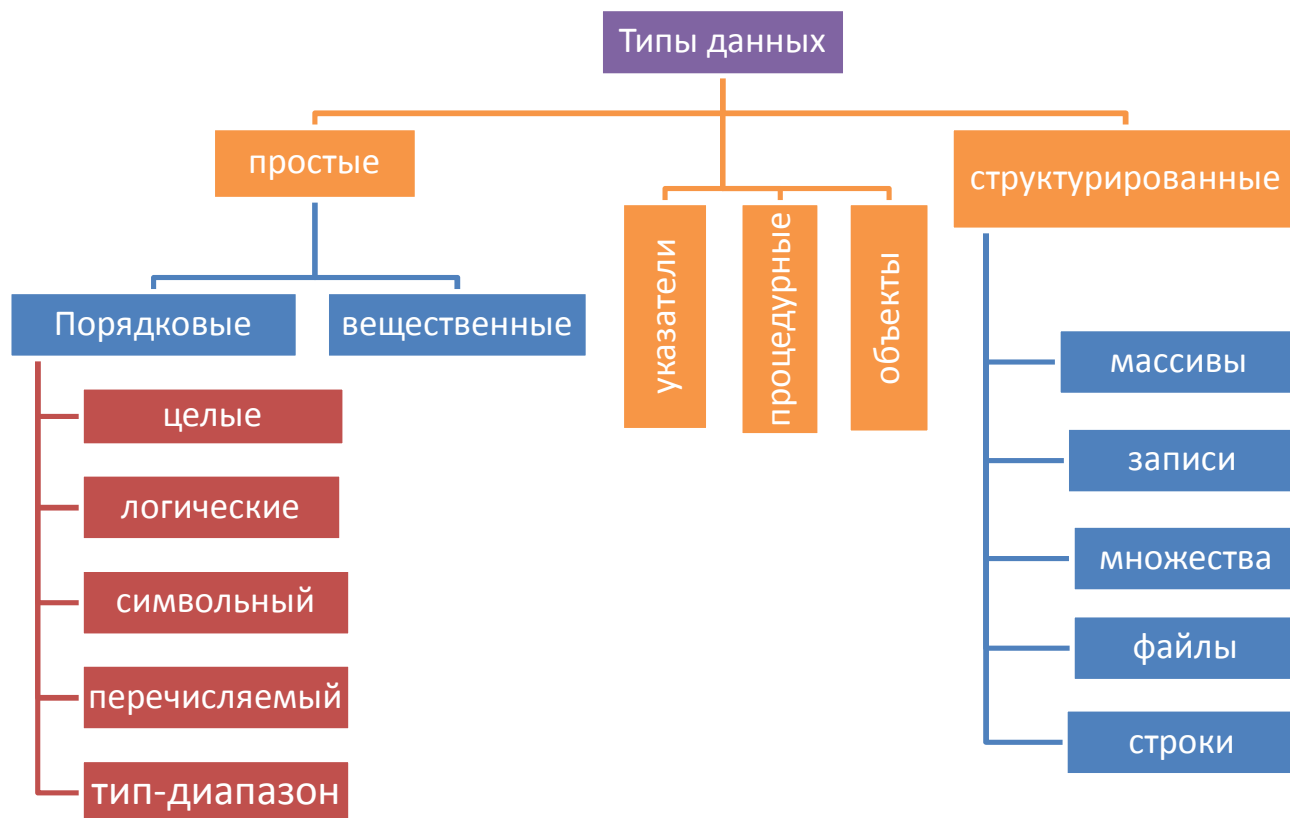
Описание меток

Раздел описания меток начинается с зарезервированного слова **label**, после которого следует список меток, перечисляемых через запятую. В качестве меток могут быть использованы идентификаторы и положительные целые числа:

label a1,l2,777777;

Описание типов

- *Раздел описания типов* начинается со служебного слова **type**, после которого следуют строки вида
- *имя типа* = *тип*;
- Например,
- **type**
 myint = integer;
 arr10 = **array** [1..10] **of** integer;
 pinteger = ^integer;
 A = **class**
 i: integer;
 constructor Create(ii: integer);
 begin
 i:=ii;
 end;
end;



Простые (порядковые) типы

Тип	Идентификатор	Размер байтах	в	Диапазон значений
логический	boolean	1		true, false
символьный	char	1		все символы кода ASCII

Интервальный тип (тип-диапазон) определяется пользователем и формируется только из порядковых типов. Представляет собой подмножество значений в конкретном диапазоне.

Можно создать собственный тип данных простым перечислением значений, которые может принимать переменная данного типа. Это так называемый **перечисляемый тип данных**.

Целочисленные типы

Тип	Диапазон	Формат	Размер байтах
Byte	0..255	Беззнаковый	1
ShortInt	−128..127	Знаковый	1
SmallInt	−32768..32767	Знаковый	2
Word	0..65535	Беззнаковый	2
Integer	−32768..32767	Знаковый	2
LongWord	0..4294967295	Беззнаковый	4
LongInt	−2147483648..2147483647	Знаковый	4
QWord	0..18446744073709551615	Беззнаковый	8

Вещественные типы

Идентификатор	Длина (байт)	Диапазон значений
real	6	$2,9 \times 10^{-39} - 1,7 \times 10^{38}$
single	4	$1,5 \times 10^{-45} - 3,4 \times 10^{38}$
double	8	$5 \times 10^{-324} - 1,7 \times 10^{308}$
extended	10	$3,4 \times 10^{-4932} - 1,1 \times 10^{4932}$

Структурированные типы

- Массив – это структура, занимающая в памяти единую область и состоящая из фиксированного числа компонентов одного типа.
- Строки представляет собой последовательность символов. Причем количество этих символов не может быть больше 255 включительно. Такое ограничение характерная черта Pascal.
- Запись – это структура, состоящая из фиксированного числа компонент, называемых полями. В разных полях данные могут иметь разный тип.

- Множества представляют собой совокупность любого числа элементов, но одного и того же перечисляемого типа.
- Файлы для Pascal представляют собой последовательности однотипных данных, которые хранятся на устройствах внешней памяти (кстати, жесткий диск – это тоже внешняя память).

Выражения

Выражение задает правило вычисления некоторого значения. Выражение состоит из констант, переменных, указателей функций, знаков операций и скобок.

Математические операции

Символ операции	Название операции	Пример
*	умножение	2*3 (результат: 6)
/	деление	30/2 (результат: 1.5E+01)
+	сложение	2+3 (результат: 5)
-	вычитание	5-3 (результат: 2)
div	целочисленное деление	5 div 2 (результат: 2)
mod	остаток от деления	5 mod 2 (результат: 1)

Логические операции

Над логическими аргументами в Паскале определены следующие операции:

NOT - логическое отрицание («НЕ»)

AND - логическое умножение ("И")

OR - логическое сложение («ИЛИ»)

XOR - логическое «Исключающее ИЛИ»

A	B	not A	A and B	A or B	A xor B
true	true	false	true	true	false
true	false		false	true	true
false	true	true	false	true	true
false	false		false	false	false

Операции отношения

> - больше < - меньше = - равно <> - не равно

>= - больше или равно <= - меньше или равно

В операциях отношения могут принимать участие не только числа, но и символы, строки, множества и указатели.

Приоритет операций

- унарная операция not, унарный минус -, взятие адреса @
- операции типа умножения: * / div mod and
- операции типа сложения: + - or xor
- операции отношения: = <> < > <= >= in

Порядок выполнения операций переопределить можно с помощью скобок.

Например 2*5+10 равно 20, но 2*(5+10) равно 30.

Стандартные функции Pascal

Обращение	Тип аргумента	Тип результата	Примечание
Abs(x)	Целый, вещественный	Целый, вещественный	Модуль аргумента
Sqr(x)	Целый, вещественный	Целый, вещественный	Квадрат аргумента
Sqrt(x)	Целый, вещественный	вещественный	Корень квадратный
Sin(x)	Целый, вещественный	вещественный	Синус, угол в радианах
Cos(x)	Целый, вещественный	вещественный	Косинус, угол в радианах
ArcTan(x)	Целый, вещественный	вещественный	Арктангенс (значение в радианах)
Exp(x)	Целый, вещественный	вещественный	Экспонента
Frac(x)	вещественный	вещественный	Дробная часть числа
Int(x)	Целый, вещественный	вещественный	Целая часть числа
Ln(x)	Целый, вещественный	вещественный	Логарифм натуральный
Pi	-	вещественный	3,141592653
Random	-	вещественный	Псевдослучайное число в интервале [0, 1]
Round(x)	вещественный	целый	Округление до ближайшего целого
Trunc(x)	вещественный	целый	Отбрасывание дробной части числа

Возведение числа в степень

x^y $exp(y*ln(x))$

Оператор присваивания

<имя_переменной>:=<выражение>

Примеры:

Математическая запись	На языке программирования
$y = \sqrt{x}$	<code>y:=sqrt(x);</code>
$y = x - 5 $	<code>y:=abs(x-5);</code>
$y = \sin^2 x$	<code>y:=sqr(sin(x));</code>
$y = \frac{x + 5}{2}$	<code>y:=(x+5)/2;</code>

Ввод и вывод данных

- **Ввод данных**

- [read](#)(<список ввода>);
- [readln](#)(<список ввода>);

Примеры:

- `read(a,b,c);` {где a,b,c - переменные. Ввод данных осуществляется через пробел}
- `readln(a,b,c);` {где a,b,c - переменные. Ввод данных осуществляется через enter}
- Список вывода может содержать константы, переменные, выражения, формат вывода. Выражения в списке вывода разделяются запятыми.

- **Вывод данных**

- [write](#)(<список вывода>);
- [writeln](#)(<список вывода>);

Примеры:

- `write(a,b,c);` {где a,b,c - переменные. После вывода данных на экран, курсор останется на последнем символе}
- `writeln(a,b,c);` {где a,b,c - переменные. После вывода данных на экран, курсор перейдет на новую строку}
- Окончание `ln` в имени процедуры означает, что курсор автоматически будет переведен в начало следующей строки экрана.

Строки

Строковый тип данных

Для обработки строковой информации в Pascal введен строковый тип данных. Строкой в Pascal называется последовательность из определенного количества символов. Количество символов последовательности называется длиной строки. Синтаксис:

```
var s: string[n];  
    s1: string;
```

n - максимально возможная длина строки - целое число в диапазоне 1..255. Если этот параметр опущен, то по умолчанию он принимается равным 255.

Строковые константы записываются как последовательности символов, ограниченные апострофами.

Пример:

```
'Русский язык'
```

Процедуры и функции для работы со строками

1. Length(s); Функция вычисляет длину строки

Пример.

```
n := length('Pascal'); {n будет равно 6}
```

2. Слияние нескольких строк в одну строку:

а). Concat(s1,s2,...,sn); Функция выполняет слияние нескольких строк в одну в порядке их следования

б). Операция слияния нескольких строк в одну также возможна с помощью знака +.

Пример:

```
a := 'Русский '  
b := 'язык';  
c := a + b;
```

3. Copy(s,n,k); Функция копирования из исходной строки s, начиная с позиции n, длиной k символов,.

Пример:

```
s := 'Литература';  
s2 := copy(s, 1, 6); {s2 будет равно 'Литера'}
```

4. Delete(s,n,k); Процедура удаляет из строки s, начиная с позиции n, длиной k символов,

Пример:

```
s := 'Родной язык';  
delete(s,1,7); {s будет равно 'язык'}
```

5. Insert(s1,s,n); Процедура вставляет подстроку s1 в строку s, начиная с позиции n.

Пример:

```
s1 := 'y';  
s := 'дочка';  
insert(s1,s,2); {s будет равно 'удочка'}
```

6. Pos(s1,s); Функция производит поиск в строке s подстроки s1.

Результатом функции является номер первой позиции подстроки в исходной строке. Если подстрока не найдена, то функция возвращает 0.

Пример:

```
s := 'Автомат';  
s1 := 'том';  
s2:= 'Атом';  
x1 := pos(s1, s); {x1 будет равно 3}  
x2 := pos(s2, s); {x2 будет равно 0}
```

7. Chr(n); Функция возвращает символ по коду, равному значению выражения n.

8. Ord(c); В данном случае функция возвращает код символа c.

9. UpCase(c); Если c - строчная латинская буква, то функция возвращает соответствующую прописную латинскую букву, в противном случае символ c возвращается без изменения.

10. Str(X, s); Процедура преобразует численное выражение X в его строковое представление и помещает результат в s.

11. Val(s,x,c); Процедура преобразует строковую запись числа, содержащуюся в s, в числовое представление, помещая результат в x. Если в s встречается недопустимый символ, то преобразование не происходит, а в s записывается позиция первого недопустимого символа. Если после выполнения процедуры c равно 0, то это свидетельствует об успешно произошедшем преобразовании.

Алгоритм — описание последовательности действий (план), строгое исполнение которых приводит к решению поставленной задачи за конечное число шагов.

Виды алгоритмов:

- **Линейный** – это такой алгоритм, в котором все действия выполняются последовательно друг за другом.
- **Разветвляющийся (ветвление)** – при выполнении алгоритма действия исполнителя определяются результатами проверки некоторых условий.
- **Циклический (цикл)** – при исполнении алгоритма отдельные команды или группы команд повторяются многократно.
- **Вспомогательный** – алгоритм, по которому решается некоторая подзадача из основной задачи и который, как правило, выполняется многократно.

Вспомогательный алгоритм, записанный на языке программирования, называется подпрограммой или процедурой.

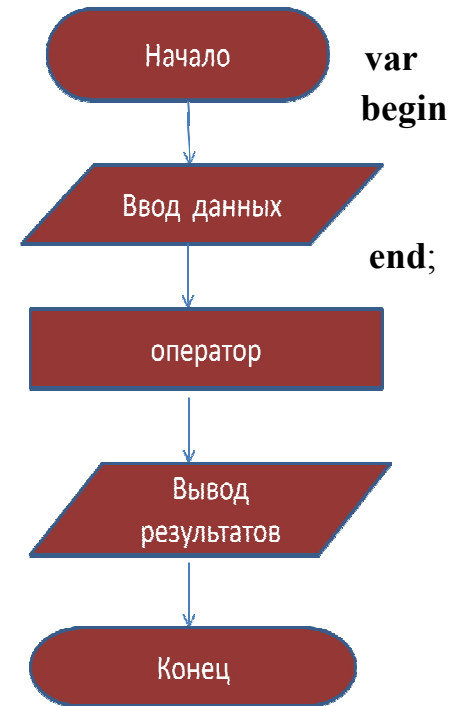
Линейный алгоритм

Program имя_программы;
 {описание данных}
 readln(ввод данных);
оператор
writeln(вывод результатов);

Пример:

Напишите программу знакомства.

```
Program Dialog;  
var c, c1:string;  
begin  
  write('Привет! Как тебя зовут?');  
  readln(c);  
  write(c, ', как твои дела?');  
  readln(c1);  
  write('До свидания,', c);  
end.
```



Задачи для самостоятельного выполнения

1. Вычислите длину слова в предложении.
2. Используя средства обработки строк, исправьте слово «ВЫЛЫСЫПЫСТЫ».
3. Напишите программу, подсчитывающее количество вхождений заданной пользователем буквы в строке.
4. Составьте программу, определяющую, является ли введенное слово числом.
5. Введите 2 целых числа. Преобразуйте числа в строки, объедините их в одну строку и выведите на экран результат.
6. Напишите программу, как получить из слова «БЕРЕГ» получить слово «БЕРЕТ»
7. Составьте программу, которая исправляет ошибки в тексте «РУСКИЙ ЯЗЫК».

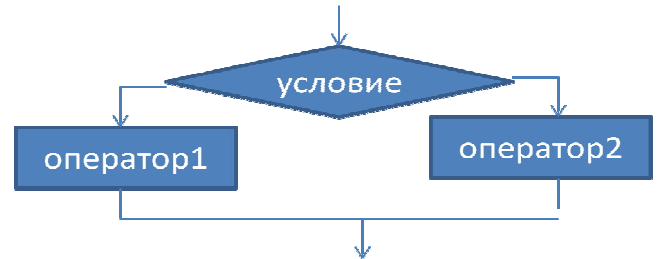
Разветвляющийся алгоритм (ветвление)

В Pascal ветвление реализуется с помощью условного оператора.

1. Полный условный оператор

- **IF** *условие* **THEN** *оператор1* **ELSE** *оператор2*;
- **IF** *условие* **THEN**

```
BEGIN  
    оператор1_1;  
    оператор1_2;  
END  
ELSE  
BEGIN  
    оператор2_1;  
    оператор2_2;  
END;
```

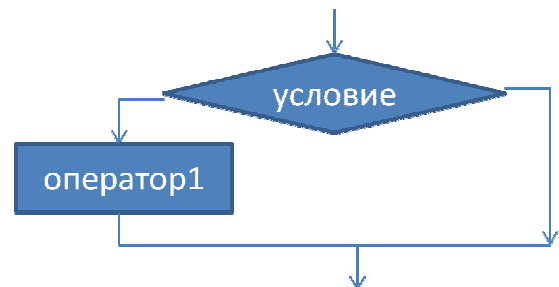


Перед **ELSE** точка с запятой никогда не ставится!!!

Неполный условный оператор

- **IF** *условие* **THEN** *оператор1* ;
- **IF** *условие* **THEN**

```
BEGIN  
    оператор1_1;  
    оператор1_2;  
END;
```



Пример:

Напишите программу определяющую, какое слово длиннее.

```
Program Slovo;  
var S1,S2:string;  
    n1,n2:integer;  
begin  
    write('введите первое слово ');  
    readln(s1);  
    write('введите второе слово ');  
    readln(s2);  
    n1:=length(s1);  
    n2:=length(s2);  
    if n1>n2 then writeln(s1, ' длиннее ',s2);  
    if n2>n1 then writeln(s2, ' длиннее ',s1);  
    if n1=n2 then writeln(s1, '=',s2);  
end.
```

Задачи для самостоятельного выполнения

1. Вычислите длину самого короткого слова в предложении из трех слов, разделенных одним пробелом.
2. Выясните, какая из букв первая или последняя встречается в заданном слове чаще.
3. Сколько букв «У» в слове стоит на четных местах?
4. Замените в заданном слове все буквы «О» пробелами.
5. В тексте подсчитайте количество гласных букв.
6. Даны два слова. Поменяйте местами буквы этих слов, занимающие одинаковые позиции.
7. Вычеркните из заданного слова все буквы, совпадающие с его последней буквой.
8. Сложное слово состоит из двух частей одинаковой длины и соединительной гласной. Найдите обе части этого слова.

Циклический алгоритм

Циклы в Pascal бывают трех видов:

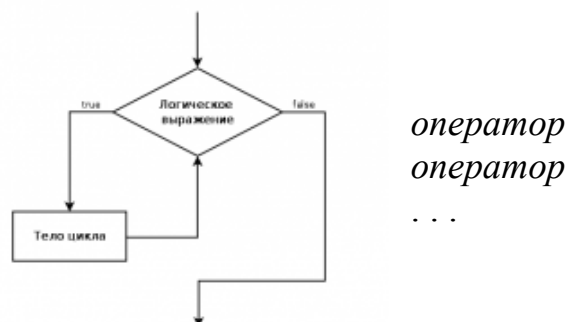
- Цикл типа «ПОКА»
- Цикл типа «ДО»
- Цикл типа «ПЕРЕСЧЁТ»

Группа операторов, называемая «телом цикла».

Цикл «ПОКА» (цикл с предусловием)

Цикл **while** является циклом с предусловием. В заголовке цикла находится логическое выражение. Если оно возвращает **true**, то тело цикла выполняется, если **false** – то нет.

```
While условие Do  
Begin  
  оператор 1;  
  2;  
  3;  
End;
```



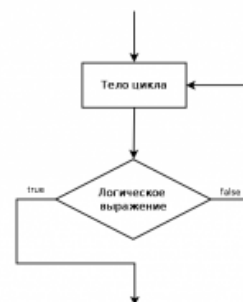
*оператор
оператор
...*

Цикл **while** может не выполниться ни разу, если логическое выражение в заголовке сразу вернуло **false**.

Цикл «ДО» (цикл с постусловием)

Бывает, что тело цикла должно выполниться хотя бы один раз, не зависимо оттого, что вернет логическое выражение. В таком случае используется цикл **repeat** – цикл с постусловием.

```
REPEAT  
  оператор1;  
  оператор2;  
  оператор3  
UNTIL выражение;
```



Цикл «ПЕРЕСЧЁТ» (циклом со счетчиком)

Часто цикл **for** называют циклом со счетчиком. Этот цикл используется, когда число повторений не связано с тем, что происходит в теле цикла. Т.е. количество повторений может быть вычислено заранее. В Pascal тело цикла не должно содержать выражений, изменяющих счетчик. *Счетчик* – это переменная любого из перечисляемых типов (целого, булевого, символьного, диапазонного, перечисления).

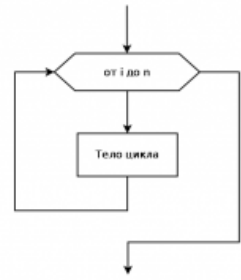
Начальные и конечные значения могут быть представлены не только значениями, но и выражениями, возвращающими совместимые с типом счетчика типы данных.

Прямой пересчет:

for счетчик:=значение **to** конечное_значение **do**
 тело_цикла;

Обратный пересчет:

for счетчик:=значение **downto** конечное_значение **do**
 тело_цикла;



Пример:

Написать программу, которая определяла бы является ли число, введенное с клавиатуры палиндромом или нет.

Program Polindrom;

var

 s: string;

 i, f: byte;

begin

 write('Введите слово, которое хотите проверить: ');

 readln(s);

 f := 0;

for i := 1 **to** length(s) **div** 2 **do**

if s[i] <> s[length(s)-i+1] **then begin**

 writeln('Не палиндром');

 f := 1;

break

end;

if f = 0 **then**

 write('Палиндром');

end.

Задачи для самостоятельного выполнения

1. Составьте программу подсчета сколько раз в тексте встречается заданный фрагмент (цепочка символов). Например, в тексте «БАНАН УПАЛ НА БАРАБАН» фрагмент «БА» встречается 3 раза.
2. Составьте программу перевода строки строчных русских букв в прописные.
3. Составьте программу, вычеркивающую каждую третью букву заданного слова.
4. Составьте программу дешифрования текстового сообщения, зашифрованного программой из задачи 3.
5. Удвойте каждую букву в данном слове «ЛИТЕРАТУРА».
6. Проверьте, можно ли вычеркиванием букв из одного слова получить другое.
7. Дана строка-предложение на русском языке. Зашифровать ее, выполнив циклическую замену каждой буквы на следующую за ней в алфавите и сохранив при этом регистр букв («А» перейдет в «Б», «а» — в «б», «Б» — в «В», «я» — в «а» и

- т. д.). Букву «ё» в алфавите не учитывать («е» должна переходить в «ж»). Знаки препинания и пробелы не изменять.
8. Дана строка-предложение на русском языке и число K ($0 < K < 10$). Зашифровать строку, выполнив циклическую замену каждой буквы на букву того же регистра, расположенную в алфавите на K -й позиции после шифруемой буквы (например, для $K = 2$ «А» перейдет в «В», «а» — в «в», «Б» — в «Г», «я» — в «б» и т. д.). Букву «ё» в алфавите не учитывать, знаки препинания и пробелы не изменять.

Литература

1. Кузнецов А.А., Апатова Н.В. «Основы информатики 8-9 классы» – М.: Дрофа, 2001. - 175с.
2. Попов Б.В. TURBO PASCAL для школьников. Версия 7.0. — М.: Финансы и статистика, 1996.
3. Фаронов В.В. Программирование на персональных ЭВМ в среде Турбо-Паскаль. - М.: Изд-во МГТУ, 1990. - 442 с
4. Изучение языков программирования в школе. - К.: Рад.школа, 1988. - 272 с.
5. Язык компьютера. - М.: Мир, 1989. - 240 с.