# Методика изучения темы «Алгоритмизация и программирование».

- Основные понятия, которые с которыми учащиеся знакомятся в курсе изучаемого раздела это - алгоритм, исполнитель алгоритма, система команд исполнителя, способы записи алгоритма, формальное исполнение алгоритма, алгоритмический язык, блок схема, линейный, разветвляющийся, циклический, и вспомогательный алгоритмы, системы программирования.
- Алгоритмическое мышление является необходимой частью научного взгляда на мир. В то же время оно включает и некоторые общие мыслительные навыки, полезные и в более широком контексте. К таким относится, например, разбиение задачи на подзадачи.
- Обучение школьника основам алгоритмического мышления базируется на понятии исполнителя. Основой для введения исполнителей служат задачи. Исполнители, используемые в курсе, традиционны. Единожды введенные исполнители в дальнейшем активно используются на протяжении всего курса. Общая схема подачи материала в курсе следующая: от частного к общему, от примера к понятию. Подача материала допускает шесть форм - стадий:

- манипуляция с физическими предметами;
- театрализация;
- манипуляция с объектами на экране компьютера;
- командный режим управления экранными объектами;
- управление экранными объектами с помощью линейных про грамм;
- продвинутое программирование с использованием процедур и
- других универсальных конструкций.

Теоретический и практический объем знаний и умений, который должен приобрести ученик в процессе изучения темы «Понятие алгоритма. Программирование» настолько велик, что требует большой подготовки учителя, наличия теоретического и методического материала. Для того чтобы ученик действительно научился программировать, он должен:

- уметь приводить примеры алгоритмов, перечислять свойства алгоритмов;
- уметь определять возможность применения исполнителя для решения конкретной задачи по системе его команд;
- знать основные алгоритмические конструкции и уметь использовать их для построения алгоритмов;
- уметь строить и исполнять алгоритмы для учебных исполнителей;
- уметь использовать стандартные алгоритмы для решения учебных задач;
- уметь записать на учебном алгоритмическом языке (или языке программирования) алгоритм решения простой задачи;
- уметь составлять простейшие алгоритмы и записывать их различными способами;
- знать один из языков программирования, основные алгоритмические конструкции языка и соответствующие им операторы языка программирования, подпрограммы: функции, процедуры, рекурсии;
- знать переменные величины: тип, имя, значение, уметь их описывать;
- знать структурированные типы данных: массивы, записи, файлы;

- уметь решать основные учебные задачи:
  - упорядочивание массива;
  - поиск минимального и максимального элементов массива с указанием их местоположения;
  - определение количества одинаковых и разных букв в тексте, количества слов в тексте;
- уметь работать с записями и файлами;
- уметь разработать программу методом последовательной детализации (сверху вниз) и сборочным методом (снизу вверх);
- знать машинную графику. Уметь построить график функции, создать движущиеся изображения, моделировать простейшие физические процессы;
- уметь применять численные методы, создавать диалоговые программы. Знать различные технологии программирования;
- знать объектно-ориентированное программирование: объект, свойства объекта, операции над объектом.

# Основные школьные языки программирования. Элементы программирования в базовом курсе информатики

- Когда-то наиболее популярными языками программирования в школах мира были Бейсик и Паскаль. Бейсик всегда считался самым простым языком программирования, а Паскаль — самым подходящим языком для обучения программированию. Но теперь это не так. Да, Бейсик прост. Но он создавался во времена, когда человечество не имело никакого опыта создания компьютерных систем, и основан на устаревших и не оправдавших себя принципах. Собственно, никакой фундаментальной целостной идеи в основе Бейсика не лежит. Сегодня есть простые и при этом более наглядные и идейно замкнутые языки программирования, нежели Бейсик. Паскаль удобен в учебных целях; ведь именно для них он и создавался. А если нужно создать настоящий программный продукт, Паскаль оказывается неудобен.