Лекция 1. Знакомство со средой.

Введение

Программа – список команд. В нашем случае это инструкций компьютера, потому что мы будем рассматривать именно компьютерные программы:

Отобразить что-нибудь на экране

Вычислить что-нибудь

Записать что-нибудь в файл

И т. д.

В наших компьютерах, телефонах, планшетах, да даже телевизорах много самых различных программ. Браузер, плеер, Word, игры, возможно даже вирусы, это все программы, созданные и используемые с какой-то конкретной целью.

Есть различные способы записать программу. Например мы можем написать команды на листочке, вопрос как это исполнять уже не наша проблема. Можно рассказать программисту что именно ты хочешь чтобы программа делала и он сам запишет за тебя все команды. Но так как мы учимся программировать, мы запишем наш список команд на специальном языке, который после определенных ***волшебных*** преобразований будет понятен нашему устройству.

Такой специальный язык называется язык программирования – это формальный язык, предназначенный для записи компьютерных программ. Язык программирования определяет набор лексических, синтаксических и семантических правил, определяющих внешний вид программы и действия, которые выполнит исполнитель (обычно — ЭВМ) под её управлением.

Языки программирования бывают разные, для примера мы возьмём Pascal, Python, JavaScript, C++. Их можно разделить на несколько категорий по различным признакам:

1. Тип исполнения – компилируемый и интерпретируемый.
2. Типизация переменных – строгая типизация и не строгая типизация.
3. Стиль написания - императивный, декларативный, объектноориентируемый.

Второй пункт мы затронем когда сегодня доберемся до переменных, а вот третий пункт пока выглядит как набор страшных слов и перестанет таким быть еще не скоро, так что пока про него немного забудем. Вернемся же к первому пункту и попытаемся понять что за страшно умные слова тут написаны.

По сути своей наша программа это обычный текст, да он написан немного иначе нежели просто сказать ***напиши на экран Привет***, но это все еще обычный текст. И если бы компьютер мог понимать обычные слова, то языки программирования были бы не нужны как явление, выучил тот же условный английский (исключительно для унификации, может и на русском бы работало) и сиди себе пиши обычными словами что ты хочешь от компьютера. Но это все мечты и далекое будущее распознавание текста и речевых конструкций, а сейчас наши устройства могут понимать только язык из 0 и 1. В связи с этим фактом надо как-то перевести нашу речь на 0 и 1. Для этого у каждого языка программирования есть свои средства и глобально именно эти «Переводчики» и бывают двух описанных выше видов.

Разница между ними в способе «перевода». Компилятор читает весь код, анализирует его, и отмечая все, даже самые маленькие и не заметные ошибки, потом сначала превращает его в так называемый промежуточный код, а после создает из него набор инструкций на языке который понимает компьютер и собирает это в исполняемый файл. Интерпретатор же в свою очередь смотрит на код немного иначе. Он так же смотрит на ошибки, но теперь уже на более глобальные, в частности на ошибки в написании стандартных структур, скобки забытые например или название функции написали не верно, более мелки ошибки он не рассматривает, после чего начинает по строчное выполнение кода, и если там находится та самая маленькая ошибка, по типу деление на ноль (если мы конечно не читаем данные из вне, там и компилятор ничего не сделает), то в языке использующей интерпретатор мы узнаем об этом только тогда когда будет выполнена та самая строка. Более подробно в это тему мы пока погружаться не будем, а перейдем дальше.

А именно перейдем к вопросу а где писать? Ответов на этот вопрос может быть много, я например скажу Spyder или VS-Code, блогеры программисты на ютубе скажут что PyCharm или SublineText лучшее что они использовали, но если быть честным и правдивым, то можно писать код даже используя обычный блокнот, не инструменты и программы делают из человека бездумно набирающего буквы настоящего программиста, а знания, умения и навыки. В связи с этим все примеры в виде скриншотов будут использовать стандартную IDLE, которая будет установлена на ваш компьютер вместе с установкой Python. Когда мы непосредственно дойдем до запуска программ я покажу какие для этого есть способы, так что если IDLE вам не нравится выбрать и установить понравившуюся IDE на ваше усмотрение, лишь бы вам было удобно.